THÈSE

présentée par

Nabil LAYAÏDA

pour obtenir le titre de

Docteur de l'Université
Joseph Fourier − Grenoble 1
(arrêté ministériel du 5 juillet 1984 et du 30 mars 1992)

Spécialité : Informatique

Madeus : système d'édition et de présentation de documents structurés multimédia

Date de soutenance le : Jeudi 12 Juin 1997

Composition du jury :

Président :	Jean−Pierre Verjus
Rapporteurs :	Malik Ghallab
	Alain Derycke
Examinateurs :	Sacha Krakowiak
	Michel Adiba
Directeur de thèse :	Vincent Quint

Thèse préparée dans le cadre du projet Opéra,
Institut National de Recherche en Informatique et en Automatique, UR INRIA Rhônes−Alpes.

À Catherine

Remerciements

Je commence par les chefs, de toutes façon je n'ai pas le choix :)

J'ai eu l'immense privilège d'avoir un super chef durant ces trois années, Vincent Quint directeur de recherches à l'INRIA et directeur de la branche européenne du consortium World Wide Web, de m'avoir accueilli au sein de l'équipe qu'il dirige, et de m'avoir donné la possibilité de mener à bien cette thèse.

Je tiens à remercier les membres du jury

Jean−Pierre Verjus directeur de l'INRIA Rhône Alpes et professeur à l'INPG de m'avoir fait l'honneur de présider mon jury de thèse.

Malik Ghallab directeur de recherches au CNRS et Alain Derycke professeur à l'université de Lille 1 d'avoir accepté de juger ce travail.

Sacha Krakowiak professeur à l'université Joseph Fourier qui a accepté de m'initier à la recherche pendant le DEA et Michel Adiba directeur de l'école doctorale et professeur à l'université Joseph Fourier d'avoir accepté de faire partie de mon jury.

Je remercie Cécile Roisin qui a accepté la tâche ingrate de la lecture des premières versions du manuscrit et m'a permis de l'améliorer. Les pages qui suivent doivent beaucoup à ses critiques.

Je voudrais aussi remercier mes collègues du projet Opéra et du consortium W3C qui, par leurs conseil, leur encouragements ou leur aide ont contribué à l'aboutissement de ce travail. Je remercie tout particulièrement Loay, Muriel, Irène, Daniel, Ramzi, Laurent et Laurent, Stéphane, Manuel, Dominique, Maria, Frédéric, Jean−Yves, Vladimir, Roland, Xavier, Jacques, ...

Enfin et surtout, je remercie mes parents et ma nombreuse famille pour la super ambiance qui à toujours régné chez nous : Brahim, Farida, Djahid, Abdenour, Redouane, (moi−même :), Karim, Nawel, Adel, Oussama, Khaled, Zoubir et la dernière Afaf : longue vie à ma mère.

Chapitre I

Introduction

Chapitre II

Introduction aux systèmes et à l'édition multimédia

Chapitre III

Modèles temporels pour les documents multimédia

Chapitre IV

Architecture et représentation des documents dans Madeus

Chapitre V

Représentation et gestion des contraintes temporelles dans Madeus

Chapitre VI

Système de présentation de Madeus

Chapitre VII

Conclusion

Chapitre I

Introduction

I.1 Introduction

Les percées technologiques récentes en matière de multimédia ont permis d'accroître les possibilités d'interaction entre l'homme et la machine. La manipulation digitale, de graphiques, de son et d'images animées sur des stations de travail ou des ordinateurs personnels a changé la nature d'un grand nombre d'applications.

En particulier, les applications de traitement de documents électroniques, habituellement dédiées à la création et à la présentation de données textuelles et graphiques, trouvent dans le multimédia des possibilités nouvelles. L'information qu'elles manipulent est plus riche, puisqu'elles intègrent dans ces documents du son et des images animées. Ces nouveaux types de documents électroniques sont communément appelés *documents multimédia*. Le développement des réseaux de communication rapides permet par ailleurs de relier ces documents pour constituer ainsi des réseaux de *documents hypermédia* à l'intérieur desquels on peut envisager de naviguer à la manière du *World Wide Web*.

Les documents ont jusqu'ici été abordés principalement sous l'angle de leur structure logique (organisation en chapitres, sections, paragraphes, etc.), de leur structure spatiale (présentation graphique et mise en page) et de leur structure sémantique (hypertexte). Un nouveau type de structure est maintenant considéré, la structure temporelle qui décrit l'enchaînement des éléments dans le temps. L'intégration de cette nouvelle dimension dans la structure globale d'un document, ainsi que l'introduction d'éléments de base qui ont eux–mêmes une dimension temporelle (vidéo, audio, interactions de l'utilisateur...), constitue l'objet de cette thèse. Les documents étudiés ici sont donc multimédia, temporisés et interactifs[(1)].

Les standards et les modèles employés pour représenter les documents classiques, comme SGML et ODA, sont devenus inadaptés pour représenter de tels documents. De

---

(1) Dans la suite de ce mémoire, on désignera par l'expression « document multimédia » les documents temporisés, interactifs et contenant des éléments de natures diverses (texte, image, audio, vidéo, etc.).

ce fait, de nouveaux standards comme *HyTime*, *MHEG* et *HyperODA*, émergent pour les compléter. Mais les standards ne suffisent pas et il existe très peu d'outils pour la création, la modification et la présentation de documents multimédia complexes.

Le travail présenté dans cette thèse a pour objectif de contribuer au domaine de l'édition et de la présentation des documents multimédia, en considérant de façon prioritaire les besoins des auteurs.

I.2 Motivations et objectifs

Dans tous les systèmes d'édition de documents multimédia proposés actuellement, la construction d'une présentation se fait au moyen de langages de programmation, selon une approche impérative, pour définir des enchaînements temporels complexes. Cette approche comporte plusieurs inconvénients majeurs, comme la faible portabilité des documents, l'inadaptation de la programmation à la nature incrémentale du processus d'édition, la difficulté de la maintenance des documents ainsi produits, et enfin, les problèmes qu'ont les auteurs non–informaticien pour les maîtriser.

En considérant la façon dont les autres dimensions des documents (spatiale, logique, hypermédia) sont représentées et traitées dans les systèmes actuels, on s'aperçoit qu'une grande partie des inconvénients cités ci–dessus a disparu grâce à la définition de formats de documents déclaratifs. Il parait donc intéressant d'étudier comment une approche déclarative peut s'appliquer à la dimension temporelle.

Un des premiers objectifs de tout système d'édition est de répondre aux besoins de l'auteur. Dans un contexte multimédia celui–ci doit pouvoir exprimer les informations temporelles qu'il désire et uniquement celles–ci. De plus, le système doit lui offrir le moyen de contrôler de façon incrémentale la validité des informations qu'il a introduites.

On ne peut apporter une réponse pertinente au problème de la spécification de la dimension temporelle d'un document multimédia qu'en ayant une connaissance précise de tous les éléments qui constituent un système d'édition / présentation multimédia. C'est pourquoi nous avons fait le choix de prendre une approche verticale du domaine de l'édition / présentation de documents multimédia, en concevant et en mettant en œuvre un système réel.

En effet, la compréhension de la nature des traitements effectués par le système de présentation sur les objets multimédia est nécessaire pour déterminer quel type d'information temporelle l'auteur peut vouloir formuler dans les documents multimédia. Par exemple, comme il est difficile de contrôler la durée de présentation d'une vidéo, à

cause de la charge du réseau et/ou de la machine, il est important de permettre à l'auteur de spécifier des durées indéterministes pour ces types d'éléments. Un autre cas typique de source d'indéterminisme dans les présentations multimédia vient de la possibilité de spécifier des interactions utilisateur dans les documents.

Ainsi, nous pensons qu'un environnement d'édition / présentation de documents multimédia doit prendre en compte à la fois les contraintes temporelles exprimées par l'auteur et celles inhérentes à la présentation de média différents. Une des principales difficultés de cette approche est que les contraintes considérées peuvent être précises, imprécises ou incertaines.

Ces motivations nous ont conduit à aborder cette thèse selon trois axes complémentaires :

- L'intégration de la dimension temporelle dans les modèles de documents au travers d'un langage déclaratif qui s'appuie sur un modèle temporel à base d'intervalles.
- La définition de mécanismes d'analyse et de synthèse permettant d'offrir aux auteurs le moyen de contrôler leurs documentss et ce dans un contexte d'édition interactive.
- La spécification et la mise en œuvre d'un système d'édition / présentation de documents multimédia, appelé Madeus.

Dans tout travail de recherche effectué dans un cadre applicatif (ici l'édition multimédia), il est fondamental de confronter les propositions théoriques avec la réalité de l'application. C'est pourquoi, nous avons tenté de mener tout au long de cette thèse une activité équilibrée entre théorie et application. Deux types de résultats sont donc attendus :

1. Des résultats théoriques, sous la forme d'un langage de spécification qui prend en compte l'aspect quantitatif du temps, l'indéterminisme de la durée de certains objets de base et le besoin d'opérateurs d'interruption. Ce langage est associé à des mécanismes d'analyse (vérification de la cohérence d'une spécification) et de synthèse (recherche d'une solution de présentation).
2. Des résultats pratiques, sous la forme du prototype d'édition et de présentation de documents multimédia Madeus qui permet de créer et de présenter des documents d'une complexité significative.

I.3 Cadre de travail

Ce travail de thèse s'est déroulé au sein du projet Opéra de l'Inria Rhône–Alpes. Le projet Opéra s'intéresse aux documents électroniques : documents structurés, hypertextes et multimédia. Il étudie des modèles de documents qui rendent compte à la fois de leur organisation logique ou abstraite, de leur présentation graphique, de leur contenu et de leur aspect temporel. Il met également au point des techniques d'édition qui s'appuient sur ces modèles.

La plupart des actions de recherche du projet Opéra trouvent leur application dans les logiciels expérimentaux développés dans le projet :

- Thot, un système général et paramétrable pour l'édition interactive de documents et d'objets structurés ;
- Amaya, un outil d'édition/navigation pour le Web ;
- Alliance, un système d'édition coopérative ;

Le travail réalisé au cours de ma thèse a permis d'introduire au sein du projet Opéra un nouveau domaine, celui des documents multimédia, et un nouveau logiciel expérimental, Madeus.

I.4 Plan de la thèse

Ce mémoire de thèse est organisé en deux grandes parties : les deux premiers chapitres fixent le contexte de notre travail en faisant l'analyse des besoins et des applications existantes pris en fixant le cadre théorique des modèles temporels ; les trois chapitres suivants constituent notre réponse aux trois problèmes soulevés plus haut : quel langage temporel, comment assurer l'analyse et la synthèse de scénarios et quel système de présentation de documents multimédia offrir aux auteurs/lecteurs. Nous détaillons ci–dessous le contenu de chacun de ces chapitres.

Chapitre II

Le chapitre II introduit les notions de base sur les systèmes multimédia. Cette étude aborde dans une première partie les problèmes liés à la synchronisation des objets multimédia, ainsi que les différents niveaux où elle doit être considérée. La deuxième partie est consacrée à l'édition de documents multimédia. En particulier, nous présentons la décomposition de l'architecture d'un document en structures logique, spatiale, temporelle et hypermédia. Une description et une évaluation des approches existantes concernant l'édition, ainsi que celles des standards *HyTime* et *MHEG* sont proposées en fin de ce chapitre.

Chapitre III

Le chapitre III est consacré à l'étude des modèles temporels pour les documents multimédia. L'étude consiste d'abord à identifier quel type de langage on veut offrir aux auteurs pour construire des scénarios temporel.
Nous présentons ensuite différentes techniques d'analyse et de synthèse des scénarios temporels. Cette étude est effectuée à partir des travaux d'autres domaines de l'informatique (la planification en intelligence artificielle ou la recherche opérationnelle) qui sont eux aussi confrontés à des problèmes d'ordonnancement temporel.

Chapitre IV

Dans le chapitre IV, nous décrivons le système d'édition et de présentation Madeus développé dans le cadre de cette thèse. Cette description porte sur son principe de conception, son architecture générale et son interface utilisateur. Le format de document utilisé est ensuite présenté. Ce format permet de décrire les documents selon leur quatre dimensions (logique, spatiale, hypermédia et temporelle).

Chapitre V

Dans ce chapitre, nous abordons le problème de la représentation et de la gestion des contraintes au sein de l'éditeur Madeus. Une attention particulière est accordée à l'incrémentalité des mises à jour du document, aux mécanismes de vérification de la cohérence et au formatage temporel car ils constituent l'un des aspects originaux de ce travail. Cette étude nous a permis de réaliser un gestionnaire temporel qui est au cœur de l'architecture de Madeus.

Chapitre VI

Le chapitre VI décrit le système de présentation de Madeus. Son rôle est de fournir le support nécessaire pour l'ordonnancement de la présentation et la restitution de l'information multimédia à travers les dispositifs de sortie comme l'écran graphique et le haut−parleur. Il permet en plus la supervision de la présentation en prenant en compte les interactions de l'utilisateur et l'indéterminisme temporel.

Chapitre VII

La conclusion résume l'apport essentiel de ce travail et propose une vision prospective de ce nouveau domaine des systèmes de documents multimédia à

forte composante temporisée. Nous tirons aussi le bilan de la réalisation de Madeus ainsi que les perspectives de recherche suggérées par ce travail.

Chapitre II

Introduction aux systèmes et à l'édition multimédia

II.1 Introduction

Savoir manipuler de l'information multimédia est de nos jours une qualité de plus en plus recherchée. A un point tel, que le terme « multimédia » est devenu une notion courante et employée de façon très intuitive et très générale. De façon très informelle, les systèmes multimédia sont caractérisés par le traitement informatique intégré d'information exprimée dans divers média (son, image, vidéo, texte, etc.). Les éléments clés permettant cette intégration sont la puissance et les possibilités croissantes de la nouvelle génération d'ordinateurs personnels qui apparaissent sur le marché. Ces ordinateurs se distinguent de la génération précédente par leur capacité à manipuler l'ensemble de ces média, leur stockage ainsi que leur échange sous une forme entièrement digitale. Il devient ainsi possible de restituer de l'information aussi bien sous forme audio et vidéo que textuelle et graphique à un ou plusieurs utilisateurs connectés à travers le réseau. Ces traitements sont devenus possibles grâce à la banalisation dans ces machines de configurations aussi bien matérielles (cartes vidéo, audio haute qualité et lecteurs de CD) que logicielles. La puissance de traitement ainsi obtenue devient bien souvent équivalente à celle des stations de travail.

Cette augmentation du potentiel des machines crée le besoin de nouveaux types d'applications tirant profit de ces nouvelles possibilités. Les applications de type visio–conférence permettant de retransmettre presque en temps réel l'image et la parole, à partir d'un lieu distant, en sont de très bons exemples. De plus en plus, ce phénomène s'étend vers des applications existantes comme les jeux, le courrier électronique, les systèmes d'édition de documents et les systèmes hypermédia. Les premières utilisations des systèmes multimédia ont essentiellement concerné les domaines suivants :

- Les systèmes de démonstration de nouveaux outils ou produits,
- Les applications de type kiosque (point d'information, guides touristiques, billetterie, etc.),

- Les encyclopédies électroniques (actuellement plus répandues sur des supports de type CD–ROM),
- Les manuels de référence techniques (comme les procédures de réparation complexes d'avions),
- L'enseignement assisté par ordinateur et le télé–enseignement.

L'objet de ce chapitre est d'abord d'introduire les notions de base sur les systèmes multimédia avec un intérêt particulier pour le problème de la synchronisation. Cette étude couvre la nature des problèmes liés à la synchronisation des objets multimédia, les applications de cette synchronisation ainsi que les différents niveaux où elle doit être considérée. La deuxième partie est consacrée à l'édition de documents multimédia. En particulier, nous présentons la décomposition de l'architecture d'un **document multimédia** en structures logique, spatiale, temporelle et hypermédia. Une description des approches existantes concernant l'édition ainsi que celle des standards de documents multimédia est proposée en fin de ce chapitre.

II.2 Systèmes multimédia

Dans le domaine du multimédia, l'emploi du terme synchronisation fait implicitement référence au temps. Cette relation avec la dimension temporelle, qui est au centre de cette étude, est la source principale de la difficulté. Le problème a, en effet, tendance à toucher directement ou indirectement un grand nombre de disciplines en informatique (l'hypermédia, les systèmes d'exploitation, les réseaux, le domaine du temps réel, etc.). Deux problèmes supplémentaires viennent s'ajouter à ces difficultés : d'une part, le multimédia étant un domaine de recherche récent, les concepts qui s'y rapportent ne sont pas encore très mûrs, et d'autre part, les problèmes posés par la synchronisation, ainsi que leurs solutions, recouvrent en partie ceux qu'on trouve dans les autres domaines. Ce dernier aspect donne au multimédia un caractère pluri–disciplinaire souvent difficile à appréhender, mais le transforme aussi en un sujet soumis à beaucoup de controverses.

Dans cette section, nous commençons d'abord par préciser le sens que nous accordons à l'expression « système multimédia ». Ensuite, nous présentons les notions de base de la synchronisation avant de passer en revue les différents problèmes qui sont posés dans ce domaine.

II.2.1 Classification des systèmes multimédia

Plusieurs définitions ont été proposées dans la littérature pour définir le terme d'application multimédia ou système multimédia. Dans cette section, nous reprenons la classification proposée par Blakowski [10]. Bien qu'elle ne montre pas tous les aspects liés à la synchronisation, elle permet déjà de dégager une échelle de mesure par rapport à cette notion. Blakowski s'appuie sur trois critères qui permettent de considérer les différentes applications par rapport au qualifiant multimédia. Une fois combinés, ces critères permettent de dire si telle application est « plus » multimédia qu'une autre. Les critères sur lesquels repose cette classification et que nous détaillons par la suite sont au nombre de trois :

- Le nombre de média manipulés dans l'application, comme l'audio, la vidéo, le texte, les interactions de l'utilisateur,
- La nature temporelle des média supportés (continus comme la vidéo, statique comme le texte, etc.),
- Le niveau d'intégration de ces différents média au sein de l'application.

Le critère le plus simple et le plus intuitif est bien entendu le nombre de média manipulés. Mais si on s'appuie uniquement sur ce critère, un système d'édition de documents qui intègre du texte et des graphiques peut être qualifié d'application multimédia. C'est le cas, par exemple, des applications de P.A.O (Publication Assistée par Ordinateur) qui possèdent généralement des environnements graphiques très riches. A lui seul, ce critère n'est donc pas suffisant pour caractériser une application multimédia.

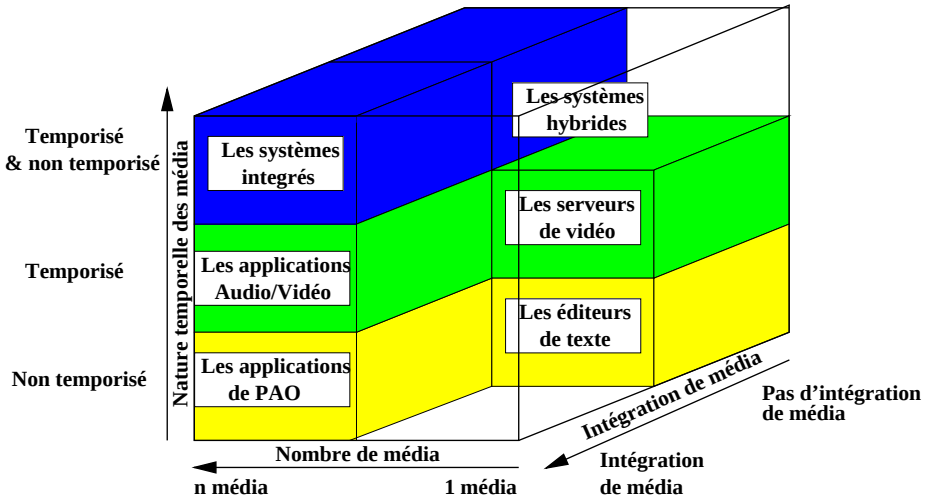


Fig. 2.1 : Classification des systèmes multimédia

La nature temporelle des média manipulés par l'application est un second critère qu'il faut prendre en compte. Selon ce critère, les média sont classés en média dépendants du temps (temporisés) et de média indépendants du temps (non temporisés). Les média temporisés comme la vidéo ou l'audio sont dépendants du temps car leur restitution nécessite plusieurs actions élémentaires de présentation espacées par des quanta réguliers de temps (un quantum pour chaque image d'une séquence vidéo). Par contre, les éléments textuels correspondent à la seconde catégorie car leur présentation se traduit par une seule action de présentation (l'affichage à l'écran), cette dernière n'étant sujette à aucune contrainte par rapport au temps du monde réel. Plusieurs auteurs [32] [71] retiennent ce dernier aspect et définissent un système multimédia comme une application capable de gérer simultanément ces deux types d'éléments. Ce dernier critère même combiné au premier reste insuffisant, car si on considère une application d'édition de documents statiques (comprenant texte et images), l'adjonction d'un simple mécanisme d'annotation audio la rendrait multimédia. Intuitivement, cet exemple montre la faiblesse des deux critères à qualifier le réel niveau de difficultés pressenti pour la construction d'un système multimédia plus complexe.

Le dernier critère retenu par Blakowski tente de compléter l'insuffisance des deux critères précédents. Il concerne le niveau d'intégration des différents média. Le sens attribué au terme intégration est pris ici dans un sens large. Il traduit la capacité d'une application à rendre homogène le traitement de média de natures différentes. Ce traitement porte en général sur la modification et la mise à jour de ces données, leur stockage et leur présentation cohérente au sein de l'application.

En combinant ces trois critères, on obtient alors la définition suivante [10] : un système ou une application est qualifié de multimédia si il/elle supporte le traitement intégré de plusieurs média dont au moins un est de nature temporisée. La Fig. 2.1 donne quelques exemples d'applications classées selon ces trois critères.

II.2.2 Unités de présentation

Les éléments multimédia (vidéo, audio, animation, interactions utilisateurs, images virtuelles, etc.) sont généralement composés d'une séquence d'unités élémentaires de grain plus fin. Ces unités appelées *Logical Data Units* (LDU) [10] ou encore *unités d'information* dans [96] se retrouvent à plusieurs niveaux dans une application multimédia. En particulier, les éléments multimédia temporisés sont formés par une séquence d'unités de présentation qui sont soumises à des contraintes de synchronisation. Si on prend l'exemple des éléments de type séquence vidéo, ils sont généralement composés d'une suite de scènes chacune correspondant à un passage particulier de la séquence (scène ou apparaît à l'écran telle personne ou telle information). Ces scènes peuvent à leur

tour être composées de sous–scènes, elles–même composées par une succession d'images fixes.

Dans les formats standard de représentation des données multimédia comme Mpeg [66], Jpeg [114] H.261 [106], les contraintes liées au volume physique occupé par ces données est le principal aspect pris en compte. Par exemple, la succession d'images dans une séquence vidéo est exploitée afin de réduire l'information nécessaire pour reconstituer l'ensemble de la séquence : c'est le principe de la compression vidéo Mpeg [66]. Le format Mpeg tire profit des similarités entre les images d'une séquence vidéo, par prédiction et interpolation (voir Fig. 2.2), afin de réduire la redondance de l'information contenue dans ces séquences. La compression crée ainsi de nouvelles dépendances entre les différentes unités de présentation liées uniquement à leur encodage. Si on considère une image particulière du format Mpeg, on retrouve encore de telles dépendances entre des blocs de pixels compressés composant l'image [114]. Celles–ci sont très importantes à considérer pour le traitement de la synchronisation dans une application multimédia, car même si à un niveau haut on peut être amené à s'intéresser plus particulièrement aux contraintes temporelles d'une succession d'images, le format de codage peut avoir une grande influence sur la synchronisation. En effet, avec le format Mpeg la présentation d'une image dépend fortement du temps lié à sa décompression, mais en plus le contenu de certaines images (images de type B ou P dans le codage) ne peut être reconstitué que si leur image de référence (image de type I dans le codage) a été préalablement traitée (voir Fig. 2.2).

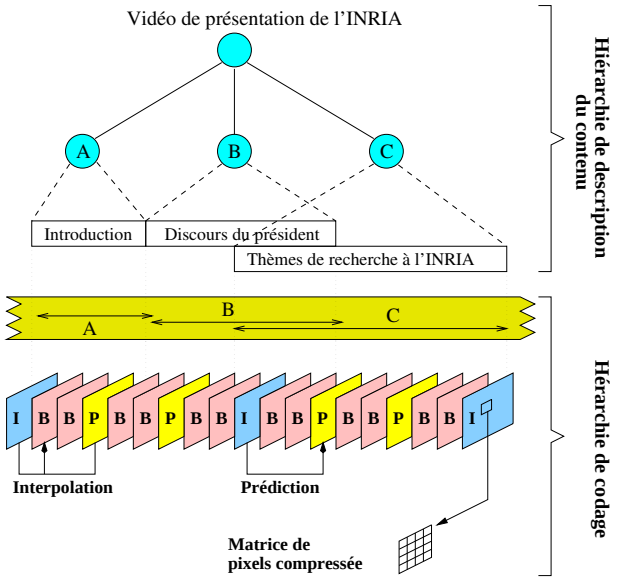


Fig. 2.2 : Hiérarchies décrivant une séquence vidéo Mpeg

Ces différents grains ainsi que les dépendances de présentation et de représentation des éléments multimédia montrent que leur organisation est le plus souvent hiérarchique. En fonction de l'application multimédia visée, l'intérêt du concepteur peut porter sur un niveau d'abstraction impliquant une partie de cette hiérarchie. Cet intérêt dépend, en premier lieu, de la nature des traitements qu'il souhaite fournir à l'utilisateur et qui peuvent concerner l'une de ces deux hiérarchies :

- la hiérarchie décrivant le **contenu** des éléments multimédia,
- et la hiérarchie liée à leur **codage**.

Le premier type de hiérarchies est naturellement plus convoité par les applications multimédia comme l'édition de documents [43], celles concernant la recherche d'information, l'indexation [116] et le stockage, comme les bases de données multimédia [1]. L'objectif de ces applications consiste à fournir à l'utilisateur des opérations de manipulation transparentes au codage des différents éléments. Cette transparence, obtenue par une description symbolique du contenu des éléments, traduit le grand besoin de ces applications en services de désignation [62]. Les aspects liés au codage sont délégués à un niveau plus bas, dit de support [23]. Notons que dans les formats les plus répandus, la hiérarchie décrivant le contenu des éléments n'est souvent pas associée à celle du codage. Il reste donc à la charge des applications d'en permettre la description, la manipulation ainsi que la mise à jour.

II.2.3 Notions de synchronisation multimédia

D'après la définition d'une application multimédia introduite dans la section II.2.1, le traitement intégré de données multimédia ressort comme un aspect central des applications multimédia. Or avant de parler d'intégration, le traitement des données multimédia passe d'abord par la considération des média temporisés ou continus comme l'audio, la vidéo ou les images de synthèse. En effet, la manipulation de ce type d'éléments soulève de nombreux problèmes, à ce jour non résolus. Dans ce nouveau type de données, le facteur temps apparaît comme une dimension essentielle de l'information, à l'opposé des données statiques classiquement gérées par les applications d'édition (texte, images, graphiques, etc.) où cette dimension n'est pas considérée. Tout au plus, on en retrouve une forme très primitive dans les outils d'édition interactifs d'exposés comme Power point, où le temps est simplement considéré comme une succession d'événements (passage d'un transparent ou d'un paragraphe au suivant, au précédent, effet de transition, etc.).

L'activité de recherche autour de la synchronisation multimédia comporte donc deux aspects :

- le support pour les média temporisés au sein d'une application,
- l'intégration des traitements de l'ensemble de ces média.

Apporter des réponses à ces deux aspects passe, avant tout, par une meilleure compréhension des problèmes posés par la synchronisation. Car celle–ci traite à la fois les problèmes liés aux données temporisées, à leur combinaison, ainsi qu'à leur intégration avec des données non–temporisées ou statiques comme le texte et les graphiques. Dans la littérature, on considère trois schémas de synchronisation :

- **Synchronisation intra−élément**

  Ce type de synchronisation s'applique aux relations temporelles d'unités d'information de base formant un élément d'un même média. L'exemple typique d'un tel type de synchronisation est la relation entre les images successives d'une séquence vidéo. Pour une vitesse de présentation de 25 images par seconde, chaque image de la séquence doit être affichée à l'utilisateur pour une durée de 40 millisecondes (voir Fig. 2.3–a).

- **Synchronisation inter−élément**

  La synchronisation inter–éléments s'applique aux enchaînements de la présentation de plusieurs éléments multimédia. Par exemple, la présentation simultanée d'un élément audio et d'une séquence vidéo suivie par des éléments de type texte et des images (voir Fig. 2.3–b).

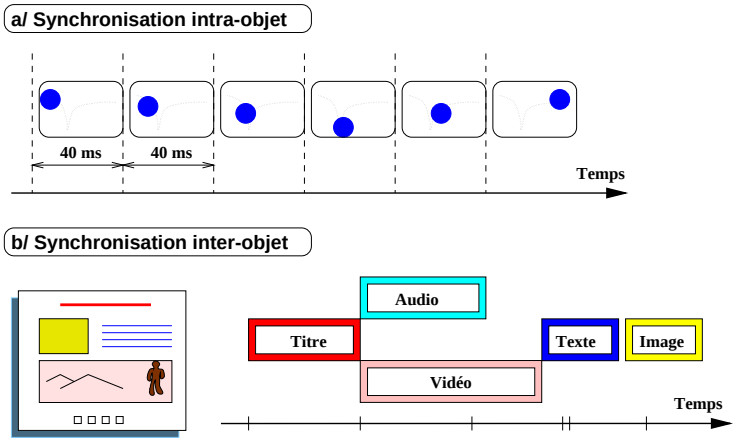


Fig. 2.3 : Synchronisation inter et intra−éléments

- **La synchronisation des lèvres** *(lip−sync)*

  Ce type de synchronisation est une combinaison des deux dernières. La synchronisation des lèvres impose un couplage temporel fort entre la progression temporelle de deux ou plusieurs éléments multimédia (leurs flux). Ce couplage est

généralement exprimé en terme de décalage temporel admissible entre les flux (*skew*) (voir Fig. 2.4–a). L'exemple typique de cette forme de synchronisation correspond à la présentation simultanée d'un discours audio et de la séquence vidéo associée. Ces contraintes permettent d'exprimer les conditions de maintien de la voix en accord avec le mouvement des lèvres. Cette forme de synchronisation n'est cependant pas réservée au couplage des médias de type audio et vidéo. On peut en effet la retrouver dans d'autres situations, comme celle illustrée dans la Fig. 2.4–b. Dans cet exemple, un élément graphique (carré) est utilisé pour annoter un élément en mouvement dans une séquence vidéo [18].

La distinction entre ces différents types de synchronisation est très importante dans les systèmes multimédia. Elle permet de comprendre la nature des relations qui existent entre les éléments et d'isoler les mécanismes adaptés pour la résolution de chacune d'entre elles. On peut déjà constater que, si le traitement de la synchronisation intra–élément peut être effectué au niveau de l'élément, le traitement des deux autres cas ne peut être effectué de la même manière.

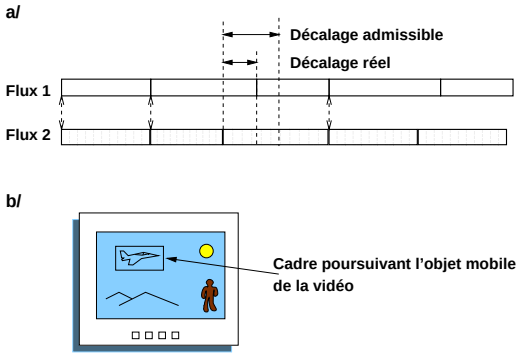


Fig. 2.4 : Synchronisation des lèvres (lip−sync)

Dans la suite, nous commençons par présenter la nature des contraintes auxquelles sont soumises les unités de présentation. Puis nous verrons dans quels cas ces différents types de contraintes sont utilisés en fonction des applications.

II.2.4 Nature des contraintes liées aux unités de présentation

En fonction de leur type, les unités de présentation peuvent être soumises à des contraintes de synchronisation très différentes. Pour illustrer ce fait, prenons l'exemple d'une séquence vidéo enregistrée à 25 images/seconde. Si on considère une application où l'image correspond à l'unité de présentation, chaque unité de présentation est contrainte par un délai d'affichage de 40 millisecondes. Prenons maintenant l'exemple d'un élément multimédia qui correspond à l'enregistrement d'une interaction de l'utilisateur, comme le

mouvement décrit par le déplacement d'une souris au dessus d'un graphique. Si on associe à chaque nouvelle position de la souris une unité de présentation, chaque unité est contrainte par une durée variable par rapport à la précédente (temps écoulé entre deux positions successives de la souris). Ces deux exemples ont comme trait commun l'aspect prévisible des durées associées à chaque unité de présentation.

Dans certains cas, cette durée ne peut pas être connue à l'avance. C'est le cas des unités de présentation qui dépendent des interactions de l'utilisateur. Par exemple, une présentation qui comporte une séquence vidéo avec des boutons de contrôle permettant de la stopper et de la redémarrer. Dans ce genre de situation, la durée effective de la séquence vidéo ne peut être définie qu'au moment où les deux interactions (le démarrage et l'arrêt) prennent effet.

En résumé, on peut donc considérer qu'il existe deux types de contraintes liées aux unités de présentation :

- **Unités de présentation à contraintes temporelles bornées :** ces contraintes pouvant être de durée fixe ou variable sur les unités de présentation successives.
- **Unités de présentation à contraintes temporelles non−bornées :** ces contraintes sont liées aux unités de présentation qui n'ont pas de durées inhérentes et qui dépendent de facteurs externes (interactions utilisateur).

Ayant présenté la nature des contraintes liées aux différentes unités de présentation intrinsèques aux éléments, nous revenons maintenant sur celles concernant deux flux d'unités de présentation couplés par des contraintes de type lip−sync. La nature de ces contraintes a fait l'objet d'études expérimentales [103] qui ont permis de mesurer les seuils tolérables du décalage entre deux flux. Ces seuils sont définis par rapport à des considérations liées à la perception humaine du décalage. Le tableau de la Fig. 2.5 montre d'importantes variations des seuils admissibles selon le mode de couplage des éléments et la nature des média considérés. Ces observations traduisent d'une part la grande dépendance qui existe entre le contenu sémantique des flux multimédia et la nature des contraintes auxquelles elles sont soumises, et d'autre part une grande richesse des contraintes temporelles.

Média		Mode de couplage	Décalage toléré
vidéo	animation	corrélés	+/– 120 ms
	audio	synchronisation des lèvres	+/– 80 ms
	image	annotation superposée annotation non superposée	+/– 240 ms +/– 500 ms
	texte	annotation superposée annotation non superposée	+/–240 ms +/– 500 ms
audio	animation	corrélés	+/– 80 ms
	audio	fortement couplés (stéréo) faiblement couplés (plusieurs orateurs) très faiblement couplés (musique de fond)	+/– 11 µs +/– 120 ms +/– 500 ms
	image	fortement couplés faiblement couplés (audio avec diapositives)	+/– 5 ms +/– 500 ms
	texte	annotations textuelles	+/– 240 ms
	curseur graphique	commentaire associé à un élément désigné	+/– 500 ms

Fig. 2.5 : Nature du couplage temporel entre les différents éléments multimédia

En conséquence, il devient nécessaire, pour une application multimédia, d'avoir un niveau de représentation où le mode de couplage entre les différents média est explicite. L'expression et la gestion de ces couplages sont regroupés dans un seul terme : **la qualité de service** [9].

II.2.5 Applications de la synchronisation

Du point de vue de son utilisation dans les applications multimédia, la synchronisation correspond à deux types de contraintes temporelles que l'on souhaite appliquer aux différents éléments. Le premier fait référence aux contraintes de temps tirées du monde réel, et est appelé synchronisation naturelle, et le deuxième vise surtout à appliquer des contraintes spécifiées par le constructeur de l'application (l'auteur). Dans ce cas, l'objectif consiste à produire des effets temporels qui ne respectent pas nécessairement les contraintes naturelles : c'est la synchronisation synthétique ou artificielle.

II.2.5.1 Synchronisation naturelle

Dans la synchronisation naturelle, la contrainte temporelle qui pèse sur les différentes unités de présentation est intrinsèque aux éléments multimédia considérés. Par exemple, pour les données temporisées comme l'audio et la vidéo, ce type de contraintes traduisent les conditions de reproduction des effets temporels liés à un élément multimédia lors de sa

saisie (sa numérisation). Ces contraintes, souvent exprimées en terme de débit (25 images par seconde), font généralement partie intégrante de la représentation des éléments. Elles sont aussi véhiculées avec la forme codée des éléments (leur format) tout au long des différentes opérations de traitement comme la saisie, le stockage, la restitution ou la transmission à travers le réseau.

Si on considère le traitement de la synchronisation naturelle, on retrouve un schéma de type producteur/consommateur dont la configuration matérielle et/ou logicielle fait intervenir trois composants :

- **Une source :** elle permet de saisir des données éphémères qui sont produites par des périphériques de saisie comme les caméras et microphones.
- **Une connexion :** qui permet de transmettre les données produites par la source vers le composant destination.
- **Une destination :** qui permet de restituer le flux de données reçu par l'intermédiaire du module connexion.

Un cas typique de la synchronisation naturelle se retrouve dans les applications d'audio et de vidéo conférence [106]. Dans ces applications, les flux multimédia sont saisis, transférés, puis restitués sur un site destination qui doit respecter les contraintes temporelles observées à la source. La connexion entre la source et la destination forme alors un ou plusieurs flux d'unités de présentation (un chemin de données) où la difficulté consiste à considérer le traitement de la synchronisation de bout en bout : la contrainte temporelle d'une unité de présentation se traduit par des contraintes sur le temps global accordé au traitement de l'opération de saisie, de transmission et de présentation.

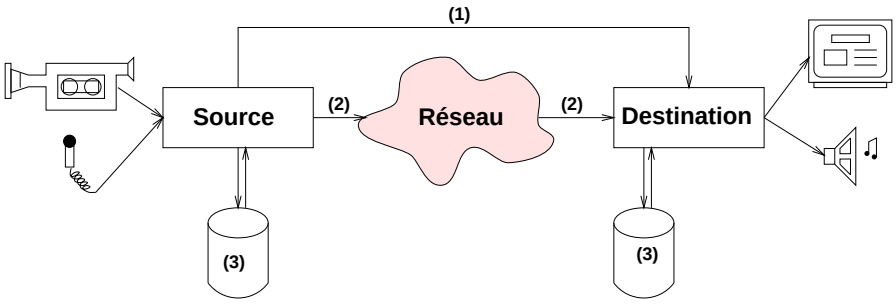


Fig. 2.6 : Chemin de synchronisation lié à un flux multimédia

Le traitement de ce type de synchronisation a fait l'objet d'un grand nombre de travaux [29][34][106]. Les solutions proposées portent sur l'adaptation dynamique de la qualité du flux audio ou vidéo en fonction de la disponibilité des différentes ressources. Par exemple (voir Fig. 2.6), le module destination peut décider de réduire la résolution graphique d'une vidéo afin de diminuer le temps CPU nécessaire au traitement du flux. Un

schéma encore plus avantageux consiste à remonter cette information au niveau de la source. Celle–ci peut, en effet, appliquer l'opération de réduction de la qualité avant la transmission. Cela permet de réduire, dans le cas d'accès distribué, la bande passante nécessaire du réseau. En réalité, de nombreux traitements sur les différents flux viennent s'intercaler entre le module source et le module destination. On peut citer la compression, la décompression et l'adaptation de la qualité graphique de l'image à l'environnement de la destination qui sont autant de causes de dé–synchronisation qu'il faut surmonter. Notons que l'accès à des données multimédia stockées à distance (voir Fig. 2.6 (3)) pose les mêmes problèmes que l'accès à des données produites en temps réel. La seule différence se situe dans les applications pouvant anticiper le chargement des données avant de les restituer à l'utilisateur. C'est le cas, par exemple, des applications de présentation de documents qui peuvent réaliser ce type de chargement en s'appuyant sur une représentation explicite des dépendances temporelles du document (connaissance du futur) [43] [69]. Il est ainsi possible, dans ces applications, d'utiliser le disque local comme cache pour offrir de meilleures garanties sur le respect des contraintes de synchronisation.

II.2.5.2 Synchronisation synthétique

Dans ce type de synchronisation, les contraintes temporelles correspondent à des critères de présentation imposés par le niveau applicatif. La synchronisation synthétique est une opération qui comporte plusieurs étapes. Elle consiste d'abord à regrouper un ensemble d'éléments multimédia provenant de sources différentes. Généralement, ces éléments sont produits et édités par des outils externes spécialisés comme les outils de saisie et d'édition d'audio, de vidéo et d'images. Ils sont ensuite combinés pour produire une nouvelle présentation multimédia plus complexe. Il existe deux schémas permettant la production de ces nouveaux éléments (éléments composites) :

- L'**intégration au niveau du codage** : cette opération consiste à produire, à partir de plusieurs flux multimédia, un nouveau flux qui sera stocké, transmis, et restitué comme une entité atomique (un seul fichier). On obtient un nouvel élément dont les différentes unités de présentation ainsi que les contraintes de synchronisation sont similaires à celles des éléments de base.
- L'**intégration par référence** : ce type d'intégration consiste à produire un nouvel élément en se basant sur la désignation d'éléments stockés de façon externe et indépendante. Un même élément de base peut ainsi faire partie de plusieurs éléments composites sans duplication de son contenu.

Dans la synchronisation synthétique, on retrouve une plus grande variété d'éléments multimédia. En particulier, des éléments comme les animations, les interactions avec l'utilisateur, les images virtuelles ou les effets vidéo sont couramment employés. Ces

éléments n'ayant pas de contraintes temporelles intrinsèques, le constructeur de l'élément composite (l'auteur) doit leur affecter des valeurs de durée avant de pouvoir les utiliser. Les éléments multimédia comme la vidéo ou l'audio peuvent aussi faire l'objet d'une modification de leur contraintes naturelles. Par exemple, le changement de la vitesse d'une vidéo peut être utilisé pour mieux faire ressortir le contenu de l'une de ses scènes (passage au ralenti pour permettre au lecteur de mieux observer ce qui s'y passe).

Les contraintes synthétiques portent aussi sur la disposition temporelle des éléments multimédia. Un exemple typique consiste à présenter les éléments multimédia en parallèle ou en séquentiel par rapport à l'axe du temps. Cette opération, appelée **composition temporelle**, est en fait bien plus complexe qu'elle ne paraît à première vue. En effet, elle se heurte d'une part à la nature temporelle très diverse des éléments multimédia et d'autre part au fait que les contraintes du monde réel et les contraintes synthétiques voulues par l'auteur ne sont pas faciles à combiner.

Les traitements de la synchronisation synthétique recouvrent en partie ceux rencontrés dans le cas de la synchronisation naturelle. Ces traitements portent en particulier, sur les aspects liés à l'accès et à la restitution de données multimédia stockées localement ou sur des serveurs distants. A ces problèmes viennent s'ajouter ceux liés à la combinaison et à l'intégration de plusieurs médias. Selon les deux formes d'intégration citées auparavant, les traitements de la synchronisation peuvent être très différents. Dans le cas de l'intégration au niveau du codage, le résultat de la composition est un nouvel élément. Dans ce dernier, les contraintes sont exprimées par des délais associés à une unité de présentation regroupant des données de divers média (multiplexage de format). Cette solution, utilisée dans le format QuickTime d'Apple [6], est relativement simple à mettre en œuvre car elle ramène le problème de la synchronisation de plusieurs éléments à celui d'un élément multimédia de base. Elle souffre néanmoins de beaucoup de limitations comme le faible niveau de réutilisation des éléments et la rigidité des contraintes imposées aux unités de présentation du fait de leur regroupement en une seule entité physique. L'intégration par référence qui encourage au contraire la réutilisation est de nos jours de plus en plus utilisée. Ce dernier aspect constitue d'ailleurs l'un des points clés du succès du World Wide Web [8].

II.2.6 Niveaux de gestion de la synchronisation

Nous avons vu dans les sections précédentes que la synchronisation multimédia se pose à plusieurs niveaux (inter, intra–média et synchronisation des lèvres) et correspond à plusieurs types d'utilisation (synchronisation naturelle et synthétique). La réalisation d'une application multimédia qui couvre tous ces aspects doit être abordée avec soin, afin de répondre de façon adaptée à chacun de ces problèmes. Il s'agit là de définir une

architecture globale où chaque niveau ou module de l'application est dédié au traitement d'un problème particulier de la synchronisation.

Il existe, dans la littérature, un grand nombre de propositions pour l'organisation des traitements dans une application multimédia [72]. Dans la plupart des cas, cette organisation s'appuie sur une décomposition hiérarchique des différents traitements. Certaines propositions se distinguent par la prédominance d'un aspect particulier, comme par exemple l'orientation objet dans le modèle de Gibbs [39], l'utilisation de langages de programmation synchrones dans le modèle de Blair [9], ou encore l'abstraction des différents niveaux par des horloges hiérarchiques dans le modèle de Rothermel [92].

Dans le cadre de cette thèse, où l'objectif est la construction d'une application d'édition et de présentation de documents multimédia, nous proposons une décomposition hiérarchique en cinq niveaux. Ces niveaux correspondent à des traitements de la synchronisation de types différents :

- **Niveau spécification :** dans ce niveau, il s'agit de décrire les différentes dépendances qui existent entre les différents éléments multimédia. Cet aspect concerne surtout les applications et les outils permettant de créer des spécifications de présentation multimédia, en particulier, les outils d'édition de documents multimédia, de présentation et de navigation hypermédia.
- **Niveau objet :** ce niveau est dédié aux traitements permettant la manipulation des différents éléments multimédia. En général, il s'agit d'identifier un ensemble d'opérations de synchronisation (démarrage, arrêt, avance rapide, etc.) qui s'appliquent à tous les média, ainsi que celles portant sur un groupe d'objets (invocation de groupe). Ces opérations sont souvent décrites au moyen de hiérarchies de classes d'objets (au sens langage à objets). Les méthodes associées à ces classes permettent de séparer les aspects d'exécution (flot de contrôle) des aspects liés aux flux d'unités de présentation multimédia (flot de données).
- **Niveau flux :** ce niveau offre des opérations qui s'appliquent sur un ou plusieurs flux de données multimédia (les éléments continus). En particulier, toutes les opérations de contrôle et de gestion des contraintes de synchronisation de type intra–élément et synchronisation des lèvres sont réalisées à ce niveau. Le mode de traitement système est idéalement de type temps–réel et les principaux problèmes traités ici se rapportent à l'allocation de ressources du système pour garantir les contraintes de qualité de service de la présentation.
- **Niveau média :** ce niveau concerne toutes les opérations de traitement associées à un flux d'unités de présentation d'un média particulier. Par exemple, les opérations concernant les transformations graphiques des images, le changement de

volume de l'audio, les effets visuels appliqués aux séquences vidéo jusqu'à la restitution sur les différents périphériques de la machine (hauts–parleurs, écran, etc.).

- **Niveau support :** le dernier niveau concerne les opérations permettant l'accès au niveau physique du système. Les opérations qu'on retrouve ici sont relatives à la lecture d'un fichier local contenant un élément multimédia, l'écriture et l'affichage sur écran, les primitives et protocoles d'accès au réseau (lecture, écriture), etc.

Chaque niveau de cette hiérarchie fournit un ensemble de mécanismes à travers des interfaces adaptées. Ces interfaces permettent, à un niveau donné de l'application, de fournir les services nécessaires en s'appuyant sur celles du niveau inférieur. En particulier, la construction d'applications d'édition de documents multimédia, dont le rôle est de permettre la description et la restitution de présentation multimédia, se trouve confrontée aux problèmes posés par chacun de ces niveaux. Il est donc important de considérer une telle organisation dans les architectures de ces applications afin de faciliter leur construction.

Une des difficultés de l'organisation en niveaux tient dans l'absence de correspondance 1–1 entre un niveau de décomposition de l'application et une couche du système hôte (voir les limites imprécises représentés dans la Fig. 2.7). On peut, par exemple, retrouver le problème de la spécification d'une présentation (composition) abordé par des langages de description graphique [43], par un langage de description dédié [73], ou encore par une extension d'un langage existant sur la base des fonctions offertes par un système d'exploitation particulier [22].

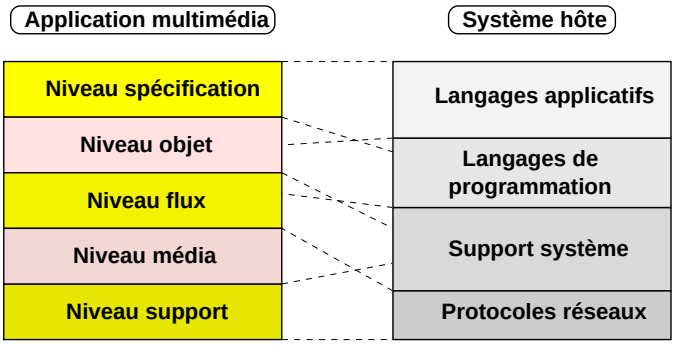


Fig. 2.7 : Les différents niveaux d'une application multimédia

Dans la suite de ce chapitre, nous abordons les problèmes liés à l'organisation d'un document multimédia (on parlera de modèle de documents) qui correspond, par rapport à la décomposition introduite ci–dessus, au niveau spécification.

II.3 Édition de documents multimédia

Un système d'édition de documents multimédia comporte principalement deux types de fonctions :

- *Des fonctions d'édition*. Elles réalisent les opérations de création, de construction et de modification du document par un auteur. L'opération de composition d'un document consiste à y inclure des éléments multimédia de base, ensuite à spécifier des relations entre ces différents éléments. Ces relations peuvent être liées à leur organisation logique (par exemple une scène composée de trois vidéo et d'un accompagnement musical), leur disposition spatiale sur les différents canaux à travers le temps (écran et/ou le haut–parleur), ou encore à leur synchronisation temporelle. Beaucoup d'auteurs attribuent au terme synchronisation dans le contexte des documents multimédia un sens plus large regroupant l'ensemble de ces relations [73].
- *Des fonctions de présentation*. Elles permettent de restituer à un utilisateur (le lecteur) le contenu du document une fois que son édition est achevée. Cette phase consiste à fournir au lecteur un ensemble de commandes permettant d'explorer ou de naviguer dans le document pour découvrir l'information qu'il contient à travers l'espace, le temps et l'interaction avec le document. Les commandes fournies au lecteur du document lui permettent l'arrêt d'une présentation multimédia, son redémarrage, son avance et retour rapide ainsi que la possibilité d'interagir avec certains éléments du document. Ce type d'interaction est aussi fondée sur des liens hypermédia, sur des boutons de présentation contenus dans le document, ou de façon générale sur tous les éléments du document qui le permettent comme les éléments multimédia programmés (*applets*) [104][107].

Le type de processus visé dans les applications d'édition multimédia est similaire à l'approche dite *WYSIWYG (What You See Is What You Get)* dans les documents statiques où les deux phases d'édition et de présentation sont intégrées [88]. À cause de la dimension temporelle des objets multimédia, une telle approche s'apparente plus à la mise au point d'un programme puisque le comportement spatio–temporel n'est restitué que pendant sa phase de présentation (son exécution). Ici les deux phases ne sont pas confondues mais on cherche à les intégrer autant que possible dans un environnement unique pour que l'auteur puisse à tout moment vérifier le résultat de ce qu'il est en train de spécifier.

II.3.1 Besoins des systèmes d'édition multimédia

Pour construire un système d'édition de documents multimédia adapté à l'utilisateur, l'interface d'édition doit offrir un ensemble d'opérations permettant la création, la modification, le stockage et la présentation des documents. La qualité de ces opérations est fortement dépendante du modèle de document choisi et des mécanismes qui le mettent en œuvre. De façon générale, un système d'édition de documents multimédia doit répondre aux exigences suivantes :

- **Une représentation structurée du document :** les aspects liés à l'organisation logique du document doivent être séparés des aspects liés à sa présentation physique et sa synchronisation temporelle. Cette structuration du document permet de garantir l'indépendance de la représentation du document (son format) par rapport à une plate–forme spécifique [88].
- **Un processus d'édition incrémental :** la création d'un document se fait généralement par un enrichissement progressif de son contenu et de sa structure. Chaque opération d'édition effectuée par l'auteur doit lui permettre de voir immédiatement son effet à la présentation.
- **Maintien de l'intégrité du document :** pour chaque opération d'édition effectuée par l'auteur, le système d'édition doit vérifier sa cohérence par rapport à l'état courant du document. Cette opération doit en plus être efficace afin de répondre au critère d'édition incrémentale.
- **Intégration des différents aspects d'un document multimédia :** il est nécessaire de prendre en compte les quatre aspects d'un document multimédia : son organisation logique, sa synchronisation temporelle, sa présentation spatiale et à l'interconnexion intra– et inter–documents (liens hypermédia) (voir II.3.2).
- **Intégration de données multimédia hétérogènes :** dans un système de documents multimédia, il est primordial de rendre homogène la manipulation de données comme le texte, les graphiques, la vidéo, l'audio et les interactions avec l'utilisateur aux sein d'une même structure de document. Cette intégration pose un certain nombre des problèmes liés principalement aux natures très diverses de ces éléments (temporisés, statiques, indéterministes).
- **Résolution automatique des contraintes temporelles :** afin d'offrir à l'utilisateur des opérations de synchronisation de haut niveau, le système d'édition doit fournir des opérations d'édition qui prennent en charge toutes les contraintes numériques induites par ces opérations : c'est une extension à la dimension temporelle du principe de formatage qu'on retrouve dans les formateurs statiques comme LaTeX [61].

Ces différentes exigences attendues d'un système d'édition multimédia proviennent d'une part du besoin de construire des applications interactives simples d'utilisation et d'autre part de permettre une représentation du document riche du point de vue sémantique et échangeable entre différentes applications multimédia. Cette représentation, qu'on appellera modèle de document multimédia, est au centre du travail de recherche mené dans l'équipe Opéra et dont les aspects liés au multimédia sont développés dans le cadre de cette thèse.

II.3.2 Modèles de documents multimédia

La définition d'un modèle de document multimédia est primordiale pour la conception d'un système d'édition. Le modèle doit permettre la représentation de toutes les relations qui peuvent exister entre les éléments d'un document multimédia. Ces relations peuvent porter sur la description de l'organisation logique du document, sa présentation spatiale, sa synchronisation temporelle ainsi que l'interconnexion entre ses différents éléments (hypermédia). On appellera dans la suite ces différentes relations : les **relations multimédia**. L'organisation logique concerne le regroupement des éléments du document en entités sémantiquement liées. Par exemple, pour un document servant de support à une présentation orale (suite de transparents), chaque transparent est généralement formé d'un titre et d'un corps, qui à son tour peut contenir d'autres éléments comme des images, de l'audio ou de la vidéo. La présentation spatiale concerne la disposition des éléments selon les différents canaux à travers le temps (audio, fenêtre d'écran, etc.). Par exemple, le titre du transparent peut apparaître 2 centimètres plus haut que le corps pendant toute la durée allouée au transparent. La synchronisation temporelle concerne la disposition des éléments du document dans le temps. Par exemple, le corps du transparent doit apparaître 2 secondes après le début du titre. La dimension hypermédia concerne la mise en place d'éléments particuliers dans le document qui permettent de naviguer entre des documents différents ou entre des parties différentes d'un même document.

La description de ces différents aspects d'un document (les relations multimédia) peut être réalisée à différents niveaux :

- **Un niveau générique :** à ce niveau, la description de l'aspect logique, temporel et spatial des documents est basée sur la notion de classes. Une classe de document ou DTD (*Document Type Definition*) en SGML [40] spécifie tous les types d'éléments qui peuvent être utilisés dans un document ainsi que toutes les relations structurales liées à l'organisation logique qui peuvent exister entre ces éléments. Ces informations structurales sont ensuite utilisées pour associer à ces classes des modèles de présentation spatiale, temporelle et hypermédia. C'est le principe des normes de documents comme HyperODA [49] et SGML [45].

- **Un niveau spécifique à un document :** À ce niveau, la description des relations multimédia est spécifique à un seul document (une instance). Les relations logiques, spatiales, temporelles et hypermédia sont spécifiées entre les éléments contenus dans le document.
- **Un niveau élément multimédia de base :** À ce niveau, on s'intéresse plus à la description du contenu des éléments multimédia de base. Chaque élément est défini par un ensemble d'attributs qui permettent de décrire tous les aspects liés à sa présentation (la durée, la vitesse de défilement, la taille et la résolution à l'écran, le format, la langue, etc.).

Les modèles de documents multimédia existants couvrent uniquement une partie de ces différents niveaux. L'approche de l'édition développée dans le projet Opéra est une approche du niveau générique qui couvre les aspects logiques, spatiaux et hypermédia d'un document. Dans la suite de cette section, nous allons présenter le modèle de document Opéra en insistant sur l'impact de l'introduction de la dimension temporelle dans ce modèle. Cette description repose donc, en partie, sur la base d'un travail existant [88][91].

II.3.3 Dimension logique

Dans un document multimédia, il est possible de distinguer des parties du document qui sont sémantiquement liées et qui peuvent être regroupées par le biais de relations logiques. Ce regroupement est un élément composite qui constitue une nouvelle entité dont la structure interne, tant sur l'aspect logique, spatial que temporel est spécifiée indépendamment du reste du document (voir exemple de la Fig. 2.8).

De telles entités logiques peuvent être également groupées pour constituer d'autres entités logiques plus complexes. Par exemple, pour construire un document multimédia présentant une équipe de recherche, on peut commencer par une scène de bienvenue de son directeur, ensuite regrouper dans un ensemble de scènes les thèmes de recherche de l'équipe. La présentation de chaque thème étant elle–même composée par la liste des intervenants décrits par un scénario curriculum vitæ et ainsi de suite.

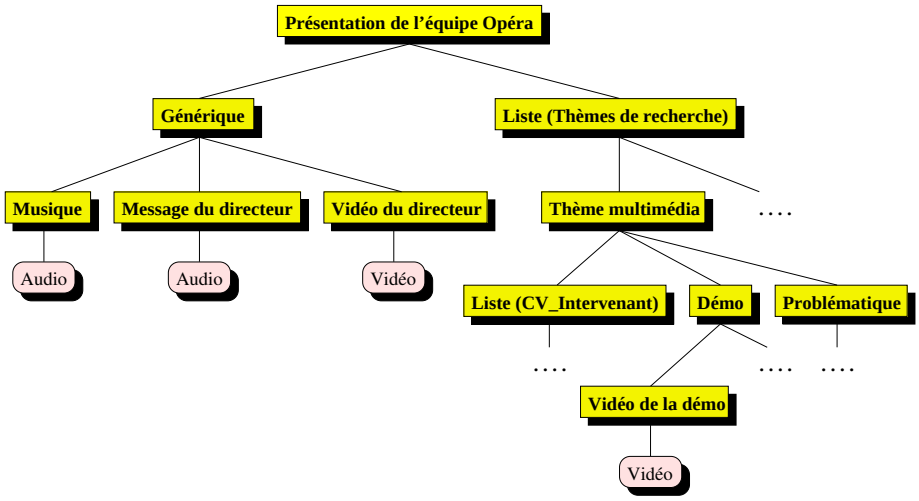


Fig. 2.8 : Structure logique d'une présentation multimédia

Ce type de construction décrit des relations d'inclusion entre les éléments et définit ainsi une classe de documents. Par exemple, toutes les équipes d'un institut peuvent décrire leur activité de recherche de la même façon que dans l'exemple de la Fig. 2.8. Cette description générique des documents est définie par une grammaire basée sur la notion de constructeurs (agrégats, liste, choix, etc.) pour décrire le schéma de structure. Le schéma de structure correspondant à l'exemple précédant est décrit de la façon suivante :

```
Présentation = BEGIN
          Introduction = Générique;
          Corps = LIST OF (Thèmes_de_recherche);
          END;
Générique = BEGIN
          Musique = AUDIO;
          Message_du_directeur = Message;
          Vidéo_du_directeur = VIDÉO;
          END;
Message = LIST OF (CASE OF
          TEXT;
          AUDIO;
          END);
Thèmes_de_recherche = BEGIN
          Schéma = LIST OF (CV_intervenant);
          Problèmatique = ... ;
          END;
```

II.3.4 Dimension spatiale

Cette dimension concerne l'affectation dans le temps des composants multimédia d'un document aux ressources physiques (l'espace). Par exemple, un élément vidéo doit disposer d'un espace géométrique sur l'écran délimité par une certaine zone. De la même façon, un élément audio doit disposer d'un canal audio pour toute la durée de sa présentation. Cette opération de mise en correspondance, appelée formatage spatio–temporel,

tient compte de différents paramètres du document qui peuvent dépendre aussi bien de sa structure temporelle que de sa structure logique. Cette opération peut être appliquée au niveau générique d'un document multimédia en définissant des règles de présentation spatio–temporelles. Par exemple, dans l'exemple de la Fig. 2.9, l'auteur peut définir une règle de présentation qui permet de libérer tous les éléments affectés à la fenêtre du document après chaque fin d'un élément *thème de recherche* présenté. Il est important de souligner que cette règle peut être aussi bien attachée au type *thème de recherche* (règles génériques) que pour une instance de ce type (règle pour un document). Les règles génériques de présentation présentent l'avantage de réduire de façon significative le nombre de spécifications requises pour la construction d'un document.

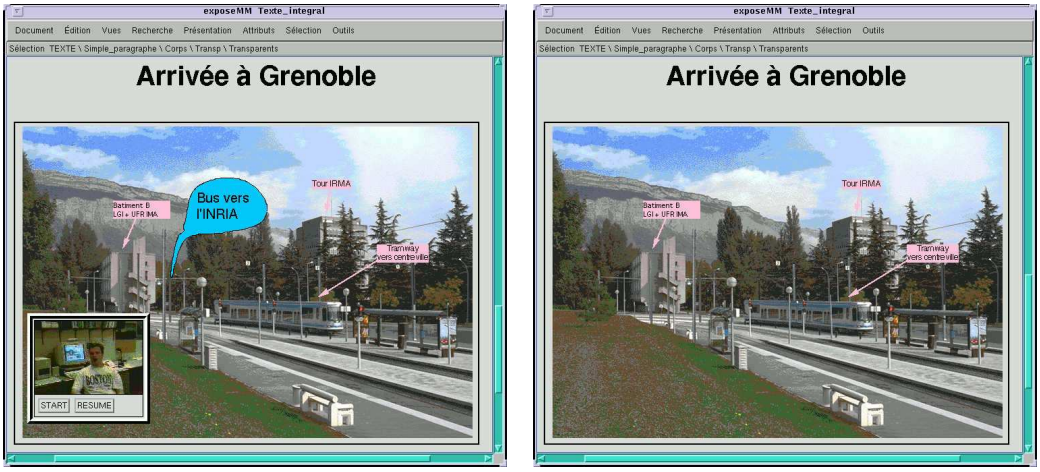


Fig. 2.9 : Exemple de disposition spatio−temporelle (scène du générique)

Le formatage spatio–temporel généralise celui du formatage géométrique dans les éditeurs de documents statiques [91][61]. Dans ce type de documents, le formatage est restreint au positionnement géométrique (sur papier ou sur écran) d'un document à partir de sa structure logique et de définitions de règles géométriques (taille de la page, des colonnes, etc.) ainsi que des caractéristiques des différents éléments du document (taille, fontes, etc.). Dans le cas de documents multimédia, la nouveauté vient de la dimension temporelle des données qui ajoute un aspect dynamique à cette opération. En effet, les espaces géométriques et les canaux sonores requis pour la présentation des éléments multimédia sont alloués, libérés, mixés pour le cas de l'audio au fur et à mesure de la présentation du document global [83]. L'exemple de la Fig. 2.9 montre la scène correspondant au générique de début de la présentation de l'équipe de recherche. La vidéo du directeur (en bas à gauche de la première fenêtre) ainsi qu'un graphique d'annotation de l'image qui accompagne son message sont présentés au début de la présentation. Ensuite,

dès que la vidéo se termine, elle disparaît de l'écran en même temps que le graphique d'annotation de l'image.

De manière générale, à chaque élément du document on fait correspondre un **intervalle temporel** qui correspond à la durée de temps pendant lequel il est actif (temps d'occupation des canaux physiques dont il a besoin). Par exemple, l'affectation au canal audio pour le son, la carte de décompression et la fenêtre de présentation pour la vidéo, ou uniquement la fenêtre de présentation pour les éléments statiques (intemporels). Toutefois, cette règle n'est pas générale et ne permet pas de décrire toutes les situations. Pour décrire de façon plus complète la dynamique d'un élément multimédia dans l'espace et dans le temps, il faut dissocier quatre instants **spatio−temporels** clés pour chaque élément du document. Ces instants permettent de définir de manière précise les relations qui existent entre les deux dimensions spatiale et temporelle des différents éléments. Ces instants, qui sont notés, *(map, start, finish, unmap),* correspondent respectivement :

- à l'affectation d'un élément à une ressource,
- à son démarrage temporel,
- à la fin de sa présentation,
- à la libération de la ressource qu'il occupe.

Ces règles s'appliquent à tous les éléments visuels qu'ils soient du texte, des graphiques ou de la vidéo. Par contre, pour les éléments sonores, les instants *map* et *start* ainsi que *finish* et *unmap* sont temporellement confondus. De la même façon que pour les éléments multimédia de base, les entités logiques possèdent aussi des instants spatio−temporels similaires. L'instant *map* d'une entité logique correspond, par transmission à ses fils logiques, à un ou plusieurs instants *map* des éléments multimédia visuels de base qu'il contient.

II.3.5 Dimension hypermédia

La structure logique présentée dans la section II.3.3 est fondamentalement hiérarchique. Cette représentation n'est pas adaptée pour décrire l'ensemble des relations qui peuvent exister entre les éléments d'un même document ou entre des documents différents. Dans le modèle d'Opéra [89], ces liens sont décrits par des éléments particuliers du document qui permettent de relier des portions de différents documents indépendamment de leur position dans cette structure. Ces liens dits hypermédia permettent de définir des relations de type sémantique entre des documents ou entre des parties de documents, comme les renvois ou les références. Ils mettent en place un réseau de portions de documents dépassant les structures qui les englobent et constituent ainsi un support pour la navigation dans un grand espace d'informations à la manière des hypertextes ou du World

Wide *Web* [8]. Un lien est défini par une ancre de départ (un élément), une ancre d'arrivée (un autre élément) et le lien qui porte une certaine sémantique [80] (renvoi, référence, annotation, etc.). Au niveau de l'interface utilisateur, un lien nécessite d'être activé explicitement par une interaction de l'utilisateur.

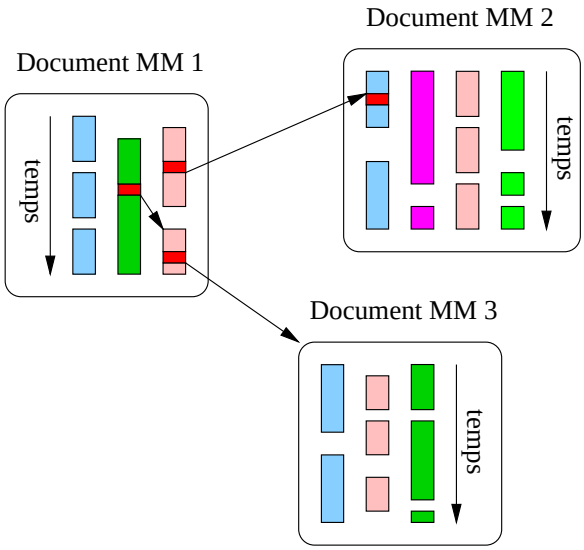


Fig. 2.10 : Exemple de liens hypermédia

Les liens hypermédia dans les documents statiques trouvent aussi leur équivalent dans la dimension temporelle (voir Fig. 2.10). En effet, l'analogie dans le temps des liens statiques est définie par des éléments particuliers (les liens hypermédia) du document qui permettent de relier des documents ou des portions de documents dans l'espace et dans le temps [70]. Par exemple, une partie d'un élément vidéo délimitée dans l'espace (une zone particulière de la vidéo) et dans le temps (la période de son apparition à l'écran) peut constituer un lien vers un nouveau document. L'activation du lien, comme dans les documents statiques, reste basée sur une interaction explicite de l'utilisateur. La seule nuance est que d'une part, l'activation de l'ancre de départ est restreinte au laps de temps couvert par sa présentation et que d'autre part, l'ancre d'arrivée correspond non seulement à un autre élément ou document mais aussi à une date précise de sa présentation (début, fin, etc.)[18].

II.3.6 Dimension temporelle

La caractéristique majeure d'un document multimédia est sa dimension temporelle. Quelle que soit leur granularité, les éléments d'un document multimédia sont reliés temporellement de sorte à définir un ordre global de présentation. Un modèle de

synchronisation temporelle pour les documents multimédia doit principalement représenter ces dépendances.

La Fig. 2.11 illustre un exemple de composition temporelle. Elle montre l'ordre temporel de la présentation de l'équipe de recherche de la .Fig. 2.8. Les boîtes de la Fig. 2.11 représentent des éléments multimédia qui consistent en des images fixes (les icônes et le texte) ou des flots continus comme les images animées (vidéo du directeur, des participants et des scènes des thèmes de recherche), ou encore des flots sonores (musique du générique, voix du directeur et des intervenants). La synchronisation de ces différents éléments par le biais de relations temporelles permet de construire une structure temporelle indépendante pour chaque partie du document formant une entité logique (voir section II.3.3).

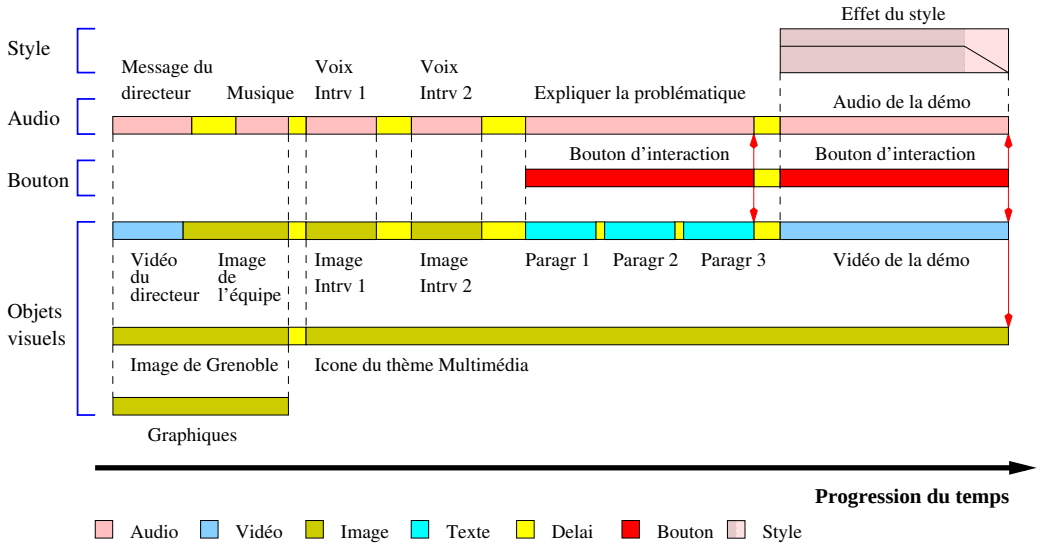


Fig. 2.11 : Scénario temporel de la présentation : équipe de recherche

Quel que soit l'élément auquel il se rapporte, chaque intervalle possède dans la dimension temporelle deux instants extrêmes, le début et la fin. Dans notre modèle, on distingue trois types d'intervalles temporels en fonction de leur rôle dans la présentation du document :

- **Les intervalles associés aux éléments logiques du document :** pour chaque élément logique, c'est l'intervalle abstrait que délimitent respectivement l'instant de début du premier élément de base contenu dans cet élément logique, et l'instant de fin du dernier élément de base de cet élément. Le premier et le dernier élément correspondent dans ce cas à l'ordre temporel et non pas à l'ordre de représentation dans l'arborescence logique du document.

- **Les intervalles de style temporel :** représentent des intervalles qui ne figurent pas dans la structure logique du document, mais qui sont ajoutés pour des besoins de contrôle du style d'une présentation multimédia. La notion de style est relative à chaque média (changement du volume pour le son, changement de couleur du texte au cours de la présentation, défilement de texte, trajectoire d'un élément sur l'écran etc.). Ces intervalles sont reliés aux éléments de la structure logique qu'ils permettent de mieux faire ressortir. Par exemple, au cours des deux secondes avant la fin de la présentation de l'élément logique *Démo* de la Fig. 2.11, on voudrait baisser progressivement le volume sonore jusqu'au niveau zéro, ou encore faire défiler progressivement le contenu de la fenêtre pour laisser place à la scène suivante.
- **Les intervalles d'espacement et de rupture temporels :** ce type d'intervalle permet d'insérer entre les intervalles des éléments logiques du document des intervalles d'espacement temporel. Un tel espacement peut correspondre à un intervalle de durée fixée entre un élément et son successeur (un délai) ou encore à un intervalle de durée indéterminée correspondant à l'activation explicite d'un bouton « suite »par l'utilisateur.

Ces deux derniers types d'intervalles sont importants pour affiner la présentation d'un document multimédia. Ils se distinguent de la première catégorie d'intervalles par le fait qu'ils n'ont pas de représentation dans la structure logique du document.

La description de la structure temporelle d'un document multimédia doit également intégrer des éléments logiques ou de base aux propriétés temporelles très diverses. En effet, un document multimédia peut contenir des éléments ayant les comportements temporels suivants :

- Des *éléments intemporels*, comme les éléments textuels et graphiques.
- Des *éléments déterministes*, comme les éléments audio et vidéo de durée connue, ainsi que les délais.
- Des *éléments indéterministes*, comme les programmes, les flots vidéo de durée indéterminée lorsque, par exemple, on y a accès par le biais du réseau ou qu'ils sont calculés en cours de présentation (images de synthèse, réalité virtuelle, bouton d'interaction avec l'utilisateur).

Le modèle temporel est un aspect central du modèle de documents multimédia. Il permet de définir les méthodes permettant le support de la structure temporelle du document. Il doit donc rendre compte de toutes les dépendances temporelles entre les composants de ce document. La présentation, et donc l'ordonnancement temporel du

document, découlera de ces dépendances. C'est cette structure qui détermine la synchronisation des composants textuels, sonores et visuels du document.

II.4 Approches de l'édition multimédia

Dans cette section, nous décrivons les grandes approches de l'édition de documents multimédia. Ces approches sont présentées à travers des outils représentatifs qui permettent d'illustrer la nature des opérations d'édition offertes à l'utilisateur. Dans le choix de ces approches, nous avons retenu celles qui reflètent le mieux les grandes tendances qui se sont dégagées ces dernières années dans la construction des outils d'édition. Elles sont au nombre de quatre :

- Les éditeurs fondés sur les **axes de temps ou timelines** où la présentation du document est décrite par un flot de données représenté par rapport à un axe de temps fixé,
- Les éditeurs fondés sur les **graphes** qui décrivent le déroulement d'un scénario multimédia au moyen de son flot de contrôle,
- Les éditeurs fondés sur les **programmes ou scripts** qui décrivent la synchronisation d'un document au moyen d'un langage de programmation,
- Les éditeurs fondés sur la **structure**, qui s'appuient sur l'organisation logique du document pour permettre sa synchronisation.

La démarche suivie dans l'analyse de ces différentes approches consiste à présenter les opérations d'édition offertes à l'utilisateur et de montrer leurs avantages ainsi que leurs limitations. Nous portons dans cette analyse un intérêt particulier aux aspects liés à la description de la synchronisation du document et donc à sa dimension temporelle. Dans la classification proposée, un système sera considéré comme faisant partie d'une classe d'approches en fonction du caractère prédominant des fonctions d'édition qu'il offre. En effet, plusieurs systèmes adoptent une approche mixte pour pallier les limitations d'une approche particulière.

II.4.1 Édition fondée sur les timelines

Les systèmes d'édition fondés sur les timelines ou axes de temps sont les plus répandus dans les systèmes commerciaux. Ces systèmes s'appuient sur une métaphore graphique d'un axe pour représenter la dimension temporelle d'un document multimédia (le flot de données formé par la succession d'unités de présentation). Les interfaces à base de timelines (voir Fig. 2.12) représentent les activités de présentation concurrentes au moyen de plusieurs axes parallèles dont chacun est réservé à un média particulier. Un axe

commun (en haut de la Fig. 2.12), partagé par tous les autres axes, est utilisé comme référence pour représenter le temps physique.

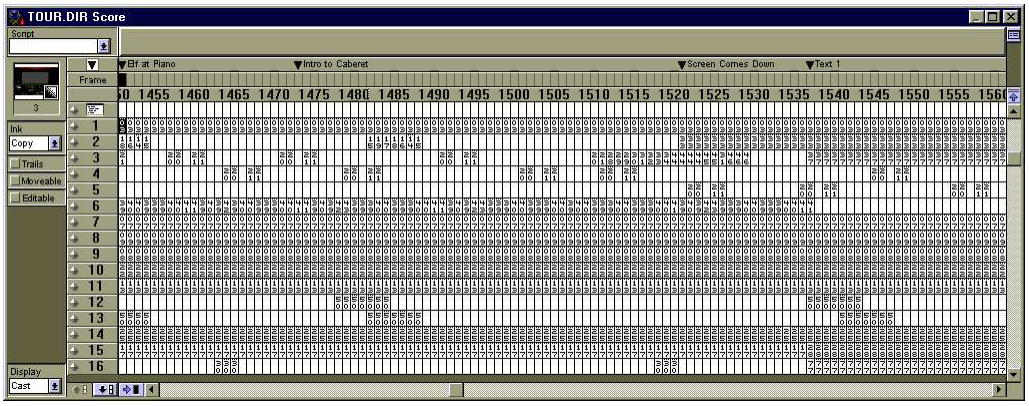


Fig. 2.12 : Interface d'édition de Director

L'activité d'édition dans ce type de systèmes est réalisée en plaçant des icônes représentant les éléments multimédia sur l'un des axes du timeline (on parle de piste). Le placement sur une piste définit alors de façon précise la date d'occurrence de la présentation d'un élément par rapport aux début du document. Certains systèmes, comme Director [75], permettent en plus de définir des effets de transition comme l'apparition ou la disparition progressive d'un élément de l'écran ou son défilement. Ces effets de style appelés « propriétés » peuvent être attachés aux éléments après leur insertion dans le document. La granularité des unités de présentation représenté sur l'axe de temps est souvent de bas niveau. Par exemple, dans le cas de Director (voir Fig. 2.12), chaque division verticale de l'axe est de l'ordre d'une image d'une séquence vidéo.

Les éditeurs à base de timelines sont bien adaptés aux présentations multimédia, en grande partie, grâce à l'interface l'utilisateur. En effet, cette interface rend bien compte et de façon intuitive du placement temporel des différents événements de présentation. Par contre, les inconvénients sont multiples et liés aux effets de bord que cette même interface engendre dans les différents traitements d'édition, à savoir :

- La modification de la durée ou de la date de début ou de fin d'un élément nécessite souvent le repositionnement à la main des autres éléments. En effet, les dépendances temporelles entre éléments ne sont pas maintenues sous une forme relatives, tout y est rapporté au temps physique. Une opération quelconque sur le document est donc susceptible d'engendrer des changements en chaîne qui doivent être effectués à la main un par un, ce qui rend la tâche d'édition fastidieuse et source de beaucoup d'erreurs.

- À cause de la datation explicite des différents événements de présentation (début et fin des éléments), il est impossible de représenter des éléments dont la valeur de durée est imprévisible (les interactions de l'utilisateur).
- Comme conséquence de la place prédominante accordée au temps au niveau de l'interface d'édition, la représentation logique du document est aplatie voire rendue inexistante. L'organisation modulaire du document qui encourage la réutilisation de ses parties est donc très limitée.

Un autre problème concernant ce type d'éditeurs est lié à la navigation. La plupart des systèmes timeline offrent la possibilité de naviguer entre des parties de documents en introduisant des liens vers des dates explicites de la présentation (pointeur vers la minute m de la présentation). Alors que ces opérations semblent relativement intuitives, le résultat de l'interaction entre les différents éléments à un instant donné n'est pas facile à définir (problème de contexte) : si un lien est associé à la seconde s de la présentation de la figure, selon les outils, il n'est pas clair que les éléments définis à cet instant doivent être réactivés (à partir d'un instant interne de leur présentation). Les problèmes de navigation dans les timelines sont en fait dus à l'utilisation systématique de dates explicites pour localiser temporellement un élément au lieu d'exploiter une structure plus riche du document.

Plusieurs systèmes d'édition à base de timelines sont largement utilisés, dont Director [75] qui est un produit commercial, Integrator [100] qui est un prototype de recherche et MAEstro [30], produit commercial issu d'un travail de recherche.

II.4.2 Édition fondée sur les graphes

Dans l'approche fondée sur les graphes, le document est représenté sous la forme du diagramme de son flot de contrôle. Ce diagramme décrit l'interaction entre les éléments multimédia du document à l'exécution. Il est utilisé de deux façons : dans certains éditeurs, son utilisation se limite à un cadre informel de synchronisation souvent motivé par des besoins d'interface graphique, dans d'autres cas, cette utilisation correspond à un véritable support langage pour des modèles temporels formels plus élaborés.

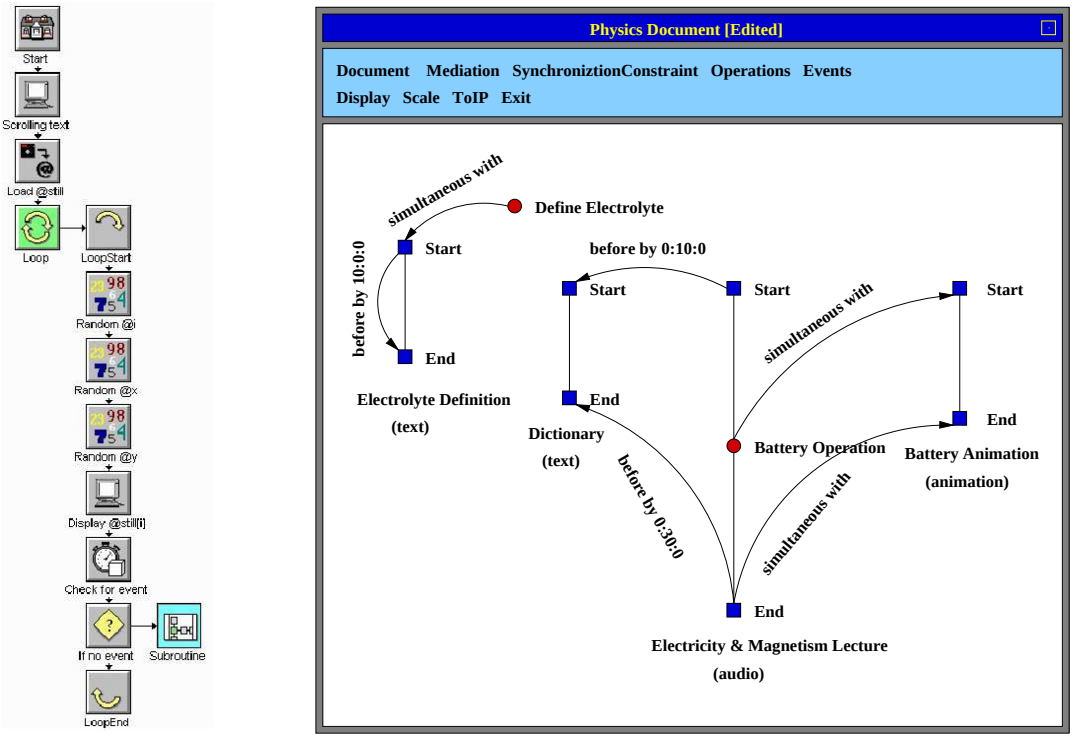


Fig. 2.13 : Interface d'Icon Author (à gauche) et de Firefly (à droite)

L'édition de documents fondée sur les graphes permet des descriptions puissantes de la synchronisation. Il est possible, en effet, de représenter beaucoup plus de combinaisons de scénarios que dans le cas des timelines. Les graphes fournissent en plus un support plus adapté pour les opérations d'édition comme la suppression ou l'insertion d'éléments car la disposition des éléments est maintenue sous forme explicite (voir l'interface de Firefly [14] dans la Fig. 2.13). Malgré ces avantages, l'édition à base de graphes souffre de plusieurs limitations :

- La description temporelle d'un document à tendance à devenir rapidement complexe et très peu lisible dès que sa taille croît. Cette complexité provient d'une représentation événementielle de bas niveau du scénario. En effet, chaque élément fait intervenir plusieurs événements temporels qui correspondent à son début, sa fin et même parfois à ses événements internes qui sont mis en relation avec d'autres éléments.
- La représentation du document est très peu structurée ce qui limite la réutilisation de ses parties.

Pour les graphes formels, ces inconvénients sont compensés par des mécanismes d'édition plus élaborés permettant, par exemple, la détection des incohérences et la

résolution automatique de la synchronisation à partir de spécifications de plus haut niveau (voir Firefly [14] et Isis [55]).

II.4.3 Édition fondée sur la structure

Les deux types de systèmes d'édition décrits précédemment partagent un caractère commun : le document multimédia est défini en terme de primitives de synchronisation temporelles appliquées directement aux éléments multimédia. Une autre approche consiste à exploiter l'organisation logique du document pour permettre sa synchronisation temporelle. C'est l'approche adoptée dans CMIFed [43] (*CWI Multimedia Interchange Format Editor*) qui constitue l'un des outils les plus complets en termes de fonctions d'édition. Le grand avantage de cette approche est la possibilité d'organiser le contenu du document en modules isolés sur lesquels on peut appliquer des primitives globales de synchronisation. Ces primitives sont la mise en parallèle ou en séquence des éléments appartenant à une même entité logique. CMIFed permet la manipulation du document à travers deux vues (voir :Fig. 2.14) : une vue structure qui permet de représenter la hiérarchie d'éléments d'un document sous forme de boîtes encapsulées. Cette vue permet aussi d'avoir un aperçu de la dimension temporelle qui est détaillée dans une deuxième vue appelée la vue canaux (*channel view*).

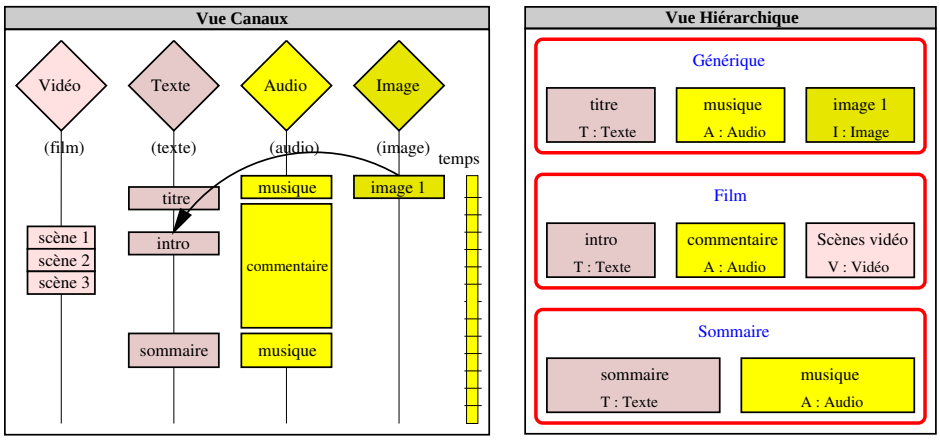


Fig. 2.14 : Vue structurelle (à gauche) et temporelle (à droite) de CMIFed

Le grand avantage de l'approche adoptée dans CMIFed est lié à la convivialité de son interface utilisateur. Le grand point faible reste, néanmoins, le pouvoir expressif limité offert par ses primitives de synchronisation. Pour pallier ce problème, des arcs supplémentaires de synchronisation sont rajoutés au document (voir la vue canaux).

II.4.4 Édition fondée sur la programmation

Dans les trois approches précédentes, des illustrations graphiques de la dimension temporelle sont utilisées au niveau de l'interface utilisateur pour permettre l'édition d'un document. Ces illustrations permettent de rendre compte des dépendances temporelles entre les différents éléments multimédia du document. En favorisant la facilité d'utilisation, ces outils pénalisent les descriptions plus fines et plus complexes de la synchronisation, puisqu'ils limitent le pouvoir expressif des scénarios. Les systèmes fondée sur la programmation comme AVC [46] et Lingo [75] sont apparus pour pallier ce problème en offrant à l'auteur d'un document la puissance d'un langage de programmation. Ils rendent ainsi possible la description d'enchaînements illimités que ce soit du point de vue temporel ou spatial. Le spectre des langages de programmation offerts couvre les langages de prototypage rapide appelés *scripts* jusqu'aux librairies de modules objets que l'on peut intégrer comme une extension d'une application existante.

```
set window = main_win
set cursor = wait
clear win
put background "image.pic"
put text "heading.txt" at 10 10
start video "cannon.mpeg" at 30 30
if (user click) start audio "hello.au" ....
```

Fig. 2.15 : Description d'un document au moyen de scripts (Lingo)

Les possibilités infinies offertes par les langages de script ont pour contrepartie le fait de substituer à la notion de document celle de programme, ce qui modifie totalement le style d'édition. La description d'un document se transforme en activité plus procédurale que déclarative et devient ainsi hors de portée d'un auteur non spécialisé ou inexpérimenté dans le domaine de la programmation. Avec cette approche (voir Fig. 2.15), la structure du document, ses relations temporelles, se dispersent au sein de l'algorithmique des scripts. La réutilisation de parties de documents dans le contexte d'un autre document devient donc difficile et leur mise à jour encore plus compliquée.

II.5 Les standards

Dans cette section, nous donnons un aperçu des deux standards existants permettant la représentation des documents multimédia. Nous examinons en particulier les facilités offertes pour décrire leur synchronisation temporelle. Une synthèse de ces aspects ainsi que la nature de l'impact de ces normes sont données dans la section II.6.

II.5.1 HyTime

HyTime est un standard international proposé pour la représentation structurée des documents multimédia et hypermédia [28][45][81]. Il est basé sur une extension de SGML [40], qui est à l'origine un standard conçu pour la représentation de documents structurés statiques. HyTime reprend le principe du marquage descriptif et des DTD (*Document Type Definition*) pour décrire des documents hypermédia intégrant la dimension temporelle. HyTime est destiné à la description de la hiérarchie de contenu et non pas à celle du codage (format). Il fournit surtout des constructions syntaxiques permettant de décrire les différentes relations entre les éléments d'un document.

HyTime utilise une désignation par adresses pour identifier des portions d'information contenue dans les différents éléments. Il fournit aussi un support de liens pour établir des interconnexions hypermédia et finalement des spécifications de l'alignement spatial et temporel pour décrire les relations multimédia entre les éléments.

HyTime définit des formes architecturales (*Architectural Forms*) qui sont des éléments décrits syntaxiquement en SGML. La sémantique associée à ces formes architecturales fait partie de la norme. La construction d'une application HyTime revient à la création d'une DTD qui lui est conforme. En fonction des besoins, la DTD d'une application donnée peut utiliser un sous ensemble des formes architecturales décrites. Dans une DTD HyTime, chaque type d'éléments est associé à une forme architecturale particulière au moyen d'un attribut spécifique.

Les formes architecturales de HyTime sont groupées dans un certain nombre de modules. Chaque application HyTime peut se restreindre à l'utilisation des modules qui contiennent les formes architecturales dont elle a besoin et ignorer le reste. Il existe un module nécessaire (*The Base Module*) et six modules optionnels interdépendants (*Measurement, Location, Address, Scheduling, Hyperlink et Rendition Module*). Le module *Scheduling* permet de placer les éléments du document dans des espaces de coordonnées finis (*FCS : Finite Coordinate Spaces*). Ces *FCS* sont définis par un ensemble d'axes spatiaux et temporels. Le module *Rendition* permet de mettre en correspondance la représentation des *FCS* avec des unités de mesure «réelles» comme les secondes, les pixels, etc.

II.5.2 MHEG

Le standard *MHEG* [87] est issu des efforts de coopération entre l'Organisation Internationale de Standardisation (*ISO*) et la Commission Internationale d'Électrotechnique *(International Electrotechnical Commission).* MHEG est destiné à la représentation et l'encodage de la forme finale de l'information multimédia et hypermédia. Cette représentation est particulièrement conçue pour l'échange de données multimédia à l'intérieur ou entre les applications. De plus, MHEG vise une grande portabilité et une meilleure garantie de la stabilité de la représentation à travers le temps (au moins une compatibilité ascendante). Le format proposé couvre une grande partie des besoins en termes de synchronisation, de support de l'interactivité et d'échange dans des environnements distribués.

Les objets MHEG

MHEG utilise une approche orientée objet pour décrire l'aspect actif, autonome et réutilisable des données multimédia. Une description, appelée *Content Objects*, permet d'encapsuler les objets d'une présentation multimédia. Pour les aspects liés au codage, MHEG utilise des standards existants comme JPEG [114] et MPEG [66]. Le groupement d'objets est réalisé au moyen d'entités appelées *Composite Objects*. Des *Action Objects* sont utilisés pour représenter les traitements à effectuer sur les objets. Par exemple, l'action *prepare* est utilisée pour initialiser un objet, *run* pour démarrer sa présentation ou encore *stop* pour l'arrêter. Des objets liens *Link Objects* permettent de spécifier des relations spatiales, temporelles ou conditionnelles entres les instances d'objets MHEG. Un lien est une entité orientée qui met en relation un objet source et un ou plusieurs objets destination du lien.

Afin de supporter les besoins spécifiques d'une application, il est possible d'étendre les classes MHEG en ajoutant de nouveaux attributs aux objets, de nouvelles actions à leur appliquer, des conditions supplémentaires d'activation des liens ou encore une représentation pour de nouveaux médias. Au lieu de définir une interface de programmation (API) pour ces extensions, MHEG s'appuie uniquement sur la description du sous–ensemble qui doit être obligatoirement respecté, le reste pouvant être librement défini par les applications. Par exemple, pour les *Actions Objects*, les actions *run* et *stop* sont obligatoires, alors que les actions *create*, *setvalue* et *getvalue* sont optionnelles.

La synchronisation dans MHEG

Dans MHEG, un système de coordonnées virtuelles est utilisé pour spécifier la disposition des objets dans l'espace et dans le temps. Ce système de coordonnées s'appuie sur une unité de mesure générique du temps appelée *GTU* (*Generic Time Unit*). Lors de la présentation, les applications ont la charge de mettre en correspondance cette unité avec

celle de l'horloge physique de la machine appelée *PTU* (*Physical Time Unit*). Trois axes spatiaux sont définis (latitude, longitude et altitude) et sont utilisés pour définir la disposition géométrique des objets. Le contrôle de la présentation des *Content Objects* est fondée sur l'échange, à l'exécution, des *Action Objects* entre les différents objets multimédia. Les *Actions Objets* peuvent être combinés pour former des listes d'actions qui permettent de décrire des actions de présentation parallèles ou séquentielles. Chaque liste est composée d'un délai initial suivi d'un ensemble d'actions de présentation. En utilisant des liens, il est possible d'obtenir un schéma de présentation entièrement basé sur les événements. En effet, des conditions d'activation quelconques peuvent être attachées à ces événements montrant ainsi la nette orientation de MHEG vers la description du flot de contrôle d'un document.

II.6 Synthèse générale

Une réflexion sur ce que doit être l'édition multimédia ainsi qu'un bon modèle de document reste un sujet ouvert. On retrouve d'un côté une panoplie d'outils commerciaux qui souffrent de beaucoup de limitations et sans réelle étude des problèmes de fond posés. D'un autre côté, les travaux de recherche [14][55] sont éparpillés dans plusieurs domaines ou encore limités à une étude théorique avec peu d'applications pratiques. Pour cette raison, nous avons préféré présenter d'abord dans ce chapitre la notion de document multimédia et d'outils d'édition avant de nous consacrer, dans le chapitre suivant, à une étude plus détaillée des modèles temporels sensés les améliorer.

Les standards MHEG et HyTime correspondent au niveau objet dans la décomposition de la synchronisation introduite dans la section II.2.6. La synchronisation dans les deux cas est décrite dans sa forme finale [76] (documents formatés). Ces normes, de l'aveu même de leurs concepteurs, présupposent donc de l'existence d'outils d'édition plus élaborés permettant de produire, à partir de spécifications de plus haut niveau, des documents qui leur sont conformes. Ce qui démontre leur inadéquation pour l'édition.

MHEG a subi beaucoup de modifications depuis sa première version (il en existe cinq). Ces modifications portent principalement sur un allègement de syntaxe (basée sur ASN.1) qui rend lourde et très coûteuse la construction d'applications conformes à MHEG. Les concepteurs d'applications multimédia restent encore très sceptiques quand à son adoption. Les différentes révisions dont il a fait l'objet sont un signe d'instabilité incompatible avec son rôle de standard. En ce qui concerne HyTime, depuis sa parution en 1990, très peu d'applications qui l'utilisent ont été rapportées (une seule à notre connaissance, nommée HyOctane [93]). Ce fait témoigne de la grande complexité de cette

norme qui limite, encore plus que MHEG, l'émergence d'un standard universel pour les document multimédia.

II.7 Conclusion

Dans ce chapitre, nous avons passé en revue la nature des problèmes liés à la synchronisation dans les application multimédia en général, et dans les applications d'édition de documents en particulier. De cette étude, il ressort que la construction d'un système d'édition/présentation de documents multimédia interactifs est conditionnée par :

- une organisation en couches de l'application d'édition traitant de manière adaptée chaque type de synchronisation,
- un format de document adapté à l'opération d'édition, mais surtout portable et intégrant les différentes dimensions du document,
- des mécanismes d'édition permettant la construction d'une interface conviviale et supportant la nature incrémentale de la phase d'édition (i.e. la modification d'un élément doit être aisée),
- des mécanismes de présentation efficaces afin d'assurer la qualité de la restitution d'un document.

Le chapitre qui suit est dédié à l'étude de la modélisation de la structure temporelle d'un document. Les choix faits pour représenter cette structure sont déterminants pour obtenir chacun des points ci–dessus mentionnés.

Chapitre III

Modèles temporels pour les documents multimédia

III.1 Introduction

Dans le chapitre précédent, nous avons montré le rôle important du temps à travers son impact aussi bien sur l'architecture d'une application d'édition et de présentation multimédia, que dans un modèle de documents. Nous avons souligné en particulier que le problème de la synchronisation temporelle s'étend depuis le niveau spécification jusqu'aux autres niveaux liés à la présentation. Pour apporter des réponses adaptées à chacun de ces niveaux, il est nécessaire d'adopter une analyse descendante des différents problèmes. En effet, l'opération d'édition qui consiste à spécifier la synchronisation temporelle d'un document multimédia constitue le point de départ de son cycle de vie. Pour un auteur, ce cycle comprenant la création, la préparation, la correction et la publication est similaire à celui rencontré par un développeur dans le domaine du génie logiciel. La réutilisation maximale, la puissance des fonctions d'édition et la facilité de maintenance des documents et de leurs composants sont des qualités très recherchées. Un système d'édition doit en outre gérer une description précise et cohérente de l'information temporelle. Une telle description conditionne, en effet, la restitution correcte du document à l'utilisateur (étape de présentation).

Cependant, on peut constater que l'on dispose de peu d'expérience en matière de composition temporelle (domaine neuf du fait de la technologie) et, de fait, les travaux pour la spécification de documents multimédia sont encore peu nombreux. Ce facteur, conjugué à la grande richesse temporelle des éléments et de leur schéma de composition temporel, rend actuellement l'opération d'édition particulièrement complexe. La réduction de cette complexité passe par la définition d'un niveau d'abstraction suffisant pour rendre cette opération plus accessible.

C'est dans cette perspective qu'on se propose, dans ce chapitre, d'étudier les modèles temporels pour les documents multimédia.

Un modèle temporel pour l'édition et la présentation multimédia est composé de trois parties [86]:

- *Un langage temporel* qui permet la spécification d'un scénario.
- *Des mécanismes d'analyse* qui permettent de prévenir les incohérences ou de les détecter à l'édition, par exemple, lorsque l'auteur introduit une relation temporelle entre deux éléments qui, compte tenu de leurs valeurs de durée, ne peut être satisfaite.
- *Des mécanismes de synthèse* qui permettent statiquement ou dynamiquement de produire une présentation conforme aux spécifications de l'auteur. Généralement le module du système d'édition qui assure cette fonction est appelé **formateur temporel** par analogie aux formateurs de texte comme Latex [61] ou Thot [88].

La première partie de ce chapitre est consacrée à la définition d'un scénario temporel, à la représentation de l'information temporelle de base et à la spécification de la synchronisation d'un document. Dans la deuxième partie, on aborde les techniques d'analyse et de synthèse des spécifications de scénarios. Cette analyse porte principalement sur le processus de composition temporelle, la vérification de la cohérence et la production automatique d'une représentation de plus bas niveau adaptée à la phase de présentation.

III.2 Scénarios temporels

Dans cette section, nous présentons la notion de scénario temporel, les différentes façons de les décrire, ainsi que les qualités attendues d'un modèle temporel adapté à l'édition de documents multimédia.

III.2.1 Définitions

Un **scénario temporel** définit comment les éléments d'un document s'enchaînent dans le temps. C'est en quelque sorte la projection du document sur la dimension temporelle.

À titre d'exemple, un document reproduisant un exposé est composé d'un ensemble de transparents qui sont enchaînés en séquence. Supposons que chaque transparent soit composé d'un texte, d'une image et d'un commentaire audio et que le temps de présentation du transparent soit égal à la durée du commentaire audio. Les trois entités sont donc reliées entre elles dans le temps (elles démarrent et finissent en même temps) et elles sont dépendantes du temps (le commentaire audio dure par exemple 2 minutes).

À tout scénario, peuvent correspondre une ou plusieurs **traces d'exécution** qui le respectent. Une trace d'exécution est définie de la façon suivante : c'est une suite de

triplets (instant, ensemble d'observations, séquence d'actions). Chaque action correspond au démarrage d'un élément et une observation à une terminaison.

On peut distinguer deux types de scénarios temporels dans une présentation multimédia :

- **Scénarios déterministes :** ils correspondent à des enchaînements entièrement définis avant la présentation d'un document. On peut calculer statiquement une date de début et une date de fin pour tous les éléments. Cela suppose en particulier qu'on connaisse la durée de tous les éléments. La trace d'un scénario déterministe est unique.
- **Scénarios indéterministes :** ils correspondent à des enchaînements qui sont définis en fonction de données connues uniquement au moment de la présentation (occurrence d'une interaction utilisateur). Un scénario indéterministe est caractérisé par l'existence d'un ensemble de traces qui lui correspondent.

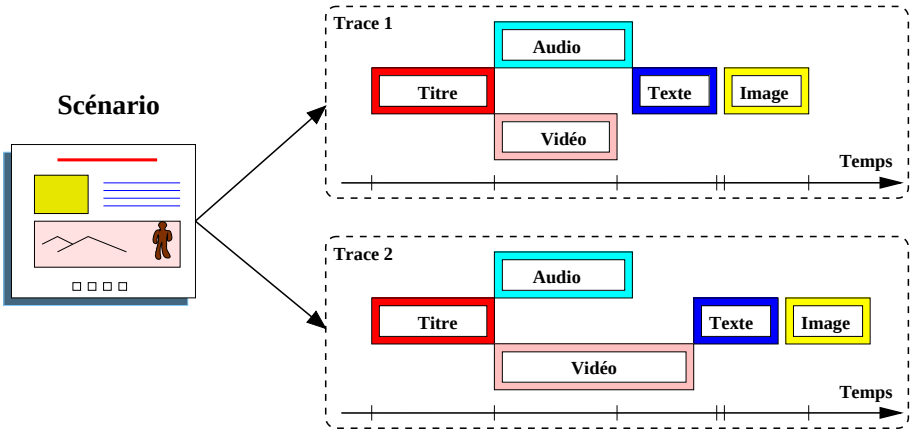


Fig. 3.1 : Exemple de scénario temporel indéterministe

Dans la figure Fig. 3.1, nous décrivons en langage naturel un exemple de scénario de type indéterministe. Dans cet exemple, une présentation est composée de cinq éléments multimédia dont les dépendances temporelles sont les suivantes : un élément de type texte (Titre) est présenté durant 40 secondes. Il est suivi d'une séquence Audio et d'une séquence Vidéo. Ces deux séquences démarrent simultanément. Lorsque les deux sont terminées, un élément Texte est présenté pendant 30 secondes suivi d'une Image. On suppose que les éléments Audio et Vidéo sont stockés sur un site distant et qu'il n'existe aucune connaissance *a priori* sur leur durée de présentation qui peut varier d'une exécution à l'autre. Chaque exécution du scénario conduit à une trace différente (voir Fig. 3.1). Nous constatons donc qu'à partir d'une même description en langage naturel, les traces peuvent varier d'une présentation à l'autre.

De façon analogue à l'exemple présenté, une méthode de spécification d'un scénario doit être suffisamment générale pour couvrir l'ensemble de ses traces. Bien entendu, cette description doit s'appliquer aussi bien aux scénarios déterministes qu'aux scénarios indéterministes.

III.2.2 Niveau de représentation des scénarios

Un scénario peut être spécifié de deux façons. La première consiste à décrire un scénario à travers une trace particulière, c'est le principe des *timelines*. Cela revient à le considérer comme un ensemble d'instants indépendants qui se rapportent à un repère temporel unique (le début du document). Dans ce type de représentation, basé sur un temps absolu, les relations temporelles entre les événements sont implicites et l'ordre induit est total. L'analyse et la synthèse sont dans ce cas inutiles, et la représentation du document peut être exploitée telle quelle pour ordonnancer la présentation. En effet, il suffit simplement de déclencher les différents événements à leur date de début. Ce type de représentation ne permet pas de représenter des scénarios indéterministes. De plus, il n'est pas adapté à la nature incrémentale du processus d'édition. En effet, toute modification nécessite une reconsidération globale de la spécification, du fait de l'incapacité de ce type de représentation à expliciter les relations temporelles. Ce sont ces caractéristiques qui nous la font qualifier de **représentation de bas niveau**.

À l'opposé, la seconde façon consiste à décrire le scénario à partir d'une mise en relation des différents éléments qui le composent. Cette forme de temps relative, contrairement à la précédente, permet de rendre explicite toutes les dépendances temporelles du scénario et induit un ordre partiel entre ses différents instants. Cette **représentation de haut niveau** permet d'obtenir des structures temporelles abstraites plus facile à organiser et à mettre à jour. Cependant, ces qualités ne sont pas acquises sans contre−partie. La nature de la représentation du temps utilisée introduit dans les spécifications de scénarios des risques d'incohérences. De plus, pour effectuer la présentation, il est nécessaire de rapporter cette représentation au temps, d'où l'importance de fournir des mécanismes d'analyse et de synthèse pour ce type de représentation.

III.2.3 Critères d'évaluation d'un modèle temporel

Avant d'aborder plus en détail les modèles temporels, il convient de préciser d'abord les critères qui permettent d'évaluer un modèle donné. Dans notre étude, l'objectif attendu de la modèlisation est double. Il s'agit d'une part de faciliter l'opération d'édition d'un document et donc de satisfaire au mieux les exigences d'un auteur, et d'autre part de produire automatiquement une forme du document adaptée à la présentation. Un modèle

donné doit répondre aux objectifs concernant la dimension temporelle des documents multimédia. Ils se résument ainsi :

- **Puissance expressive :** le langage temporel offert par le modèle doit permettre la spécification de schémas de synchronisation arbitrairement complexes. Une évaluation de l'expressivité peut se ramener à une mesure du "nombre" de schémas de synchronisation exprimables [10].
- **Structuration de l'information temporelle :** l'organisation logique d'un document, permettant la modularité et l'encapsulation de ses parties, doit être prise en compte dans les spécifications temporelles.
- **Vérification de la cohérence temporelle :** le modèle temporel doit être doté de mécanismes permettant de détecter et d'indiquer à l'auteur les incohérences d'un schéma de synchronisation donné.
- **Intégration de données multimédia hétérogènes :** la représentation et le traitement des éléments multimédia de base doivent être homogènes pour faciliter l'opération de composition.
- **Adaptation à la nature incrémentale du processus d'édition :** la création d'un document se fait généralement par une modification progressive de son contenu et de sa structure temporelle. L'incrémentalité, qui consiste à prendre en compte les opérations d'édition une par une, doit être supportée au niveau temporel.
- **Performance des mécanismes d'analyse et de synthèse :** Un environnement d'édition / présentation de documents multimédia est un environnement interactif, d'où la nécessité d'apporter une attention particulière aux performances des techniques employées [27].

À travers les modèles temporels, on cherche ainsi à retrouver deux composants importants de l'édition multimédia : le premier est un langage temporel permettant l'expression de la synchronisation et le second est l'ensemble des méthodes pour son analyse et sa synthèse. Les critères introduits doivent être évalués au niveau du modèle pour juger si celui–ci est adapté à l'édition multimédia.

III.3 Représentation de l'information temporelle multimédia

Dans le chapitre précédent, nous avons clairement identifié deux phases de traitement d'un document multimédia : son édition et sa présentation. Dans ces deux phases la dimension temporelle apparaît au cœur du problème et nécessite donc une étude plus approfondie.

De la même façon que les dimensions logique [45][49] et spatiale [91] des documents ont été modélisées, nous proposons dans cette section un modèle temporel permettant de caractériser les présentations multimédia ainsi que le processus de leur mise en relations dans un document : la **composition temporelle**.

III.3.1 Modélisation de l'information temporelle de base

Avant d'aborder les méthodes de spécification d'un scénario, nous commençons par la définition des entités de base qui le composent. En particulier, nous caractérisons le comportement temporel de chaque élément indépendamment d'un scénario donné. Cette étape nous permet de modéliser les différents éléments multimédia de façon homogène et indépendante de leur contenu et de leur type (audio, vidéo, texte, etc.), on parle alors d'**unités temporelles**.

Tout élément multimédia peut être manipulé à travers trois informations temporelles essentielles :

- Son instant de début.
- Sa durée de présentation.
- Son instant de fin.

L'une de ces trois informations est redondante. En effet, il est possible de calculer l'une en fonction des deux autres. Mais le choix des informations retenues fait partie du langage et donc du modèle, car la mise en relation des éléments se fait à partir de ces informations.

Il existe deux façons de représenter le déroulement d'un scénario : à travers les changements qui surviennent (la terminaison de la vidéo correspond au démarrage de l'audio) ou au contraire en reliant globalement les activités entre elles (la séquence audio est présenté pendant la séquence vidéo). Ceci débouche sur deux types de représentations :

1. Une représentation fondée sur les **instants**. Dans ce cas, un élément multimédia, qu'il soit logique ou de base, est décrit dans un scénario par un instant de début et un instant de fin(1), comme dans *Firefly* [14] et *Maestro* [30].
2. Une représentation fondée sur les **intervalles**. Un élément multimédia est considéré comme une entité temporelle de base décrite par sa durée comme dans *OCPN* [73] et Cmifed [16].

---

(1) On parle aussi d'événements dans certains systèmes.

Dans les représentations multimédia fondées sur les intervalles, les unités temporelles de base peuvent être classées en trois catégories en fonctions des caractéristiques attachées à leurs durées :

- *Les intervalles discrets.* Ce sont des unités dont la présentation ne dépend pas du temps, comme le texte ou les images fixes. En vue de leur intégration avec d'autres données ayant une dimension temporelle, celles–ci peuvent être affectées d'une durée de présentation explicite ou implicite dans le contexte d'un scénario donné.
- *Les intervalles déterministes continus.* Ils correspondent à des données dont la présentation dépend du temps et dont la valeur de durée est connue *a priori*. Des flots audio et vidéo sont des exemples de telles unités temporelles. Dans certains cas comme la vidéo, la durée effective de présentation de ces données est liée à la vitesse de restitution des images qui la composent.
- *Les intervalles indéterministes (discrets ou continus).* Ces intervalles se distinguent par le fait qu'ils n'ont pas de durée connue *a priori*. Ils correspondent, par exemple, à des flots audio ou vidéo continus auxquels on accède à travers le réseau. Ce sont, en partie, ces éléments qui engendrent des scénarios indéterministes.

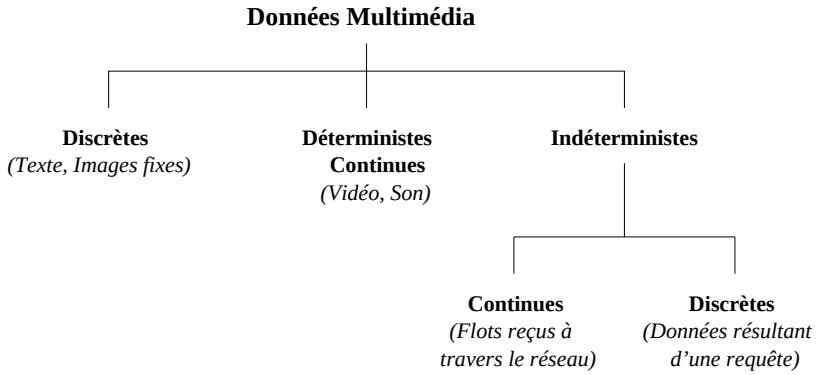


Fig. 3.2 : Les unités temporelles de base

Domaine de validité d'un intervalle

Les unités de présentation à base d'intervalles peuvent être modélisées par un ensemble décrivant leur valeurs possibles de durée ; on parle de leur domaine de validité ou de leurs bornes. Le domaine de validité d'une unité de présentation peut être défini par un triplet de valeurs [*min*, *nom*, *max*] :

- La durée minimale *min* représente la borne inférieure acceptable (au regard de la qualité de restitution) pour la durée de présentation d'un élément multimédia. La

date de l'instant de fin correspondant à cette durée est appelée **date au plus tôt**.

- La durée nominale *nom* correspond aux contraintes de synchronisation intrinsèques d'un élément.
- La durée maximale *max* représente la borne supérieure acceptable pour la durée de présentation d'un élément multimédia. La date de l'instant de fin correspondant à cette durée est appelée **date au plus tard**.

Le comportement temporel de l'ensemble des éléments multimédia peut être représenté par ces domaines de validité. Si on se restreint à la manipulation des éléments à travers ces domaines, il devient possible d'abstraire leur contenu au sein du modèle et de rendre homogène leur représentation et leurs traitements. Les contraintes liées à la perception humaine de la synchronisation se traduisent alors simplement par un choix adapté de ces trois valeurs.

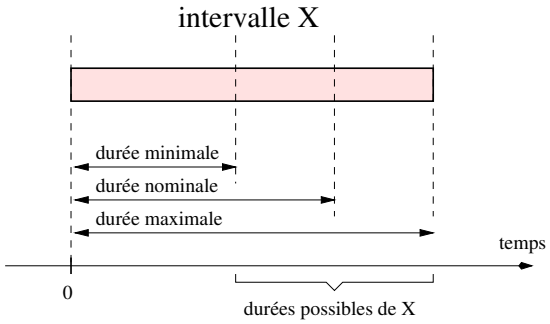


Fig. 3.3 : Domaine de validité d'un intervalle temporel

On peut noter qu'il existe des différences dans le sens attaché à ces bornes en fonction des catégories introduites plus haut (continus, discrètes, indéterministes) [32]. Ces différences peuvent se traduire par l'amoindrissement du sens de l'une des valeurs du triplet. C'est le cas des unités temporelles de type continu où le domaine de validité correspond à une notion de **flexibilité temporelle** inhérente à certains éléments multimédia. Par exemple, pour un élément de type animation représenté par les bornes [20, –, 30] secondes, l'auteur peut choisir à sa guise la valeur qui lui convient dans cette plage car il s'agit de contraintes temporelles synthétiques. Pour les éléments discrets comme le texte, cette flexibilité est infinie (bornes]0, –,∞]), et se distingue comme le cas de l'animation par l'absence de valeur optimale. Le domaine de validité constitue donc un paramètre d'édition puisqu'il est sujet à un choix d'une part pour la définition des valeurs admissibles et d'autre part pour la valeur définitive retenue pour un scénario donné. Dans certains cas comme pour la vidéo et l'audio, le domaine de validité peut être lié à une notion de **solution préférable** mesurée en fonction de l'écartement par rapport à la valeur nominale

(qui correspond aux contraintes naturelles). Par exemple, pour un élément de type audio modélisé [23, 25, 27], une durée de 26 sera jugée plus souhaitable que la valeur maximale de 27, bien que cette dernière reste acceptable. Intuitivement, ce dernier type d'éléments introduit plus de contraintes puisque le choix des valeurs de validité est généralement plus restreint que dans les exemples de l'animation et du texte.

La mise en relation d'éléments est la seconde source de contraintes. Par exemple dans un scénario dans lequel une animation est présentée en même temps qu'un élément audio, les domaines de validité doivent être combinés pour retenir celui qui convient le mieux.

III.3.2 Expression des relations temporelles

Les relations temporelles permettent de décrire la façon dont les éléments multimédia doivent être combinés temporellement pour produire le scénario d'un document. Les relations temporelles servent à la fois comme entrée de la partie analyse d'un modèle et comme une mémoire des intentions de l'auteur. Cette représentation permet de définir l'état courant d'un document sur lequel s'appliquent les différentes opérations de composition.

Afin de définir un jeu de relations ou d'opérateurs temporels permettant la construction d'un document, nous étudions dans cette section trois aspects importants liés à ces relations temporelles qui peuvent intervenir pour décrire un scénario :

- Les unités temporelles mises en jeu dans les relations.
- La sémantique temporelle associée à ces relations.
- La topologie de la structure temporelle du scénario (arbre, graphe).

Étant donnée l'existence de deux modes de représentation des unités temporelles (les instants et les intervalles), il en résulte deux classes de relations temporelles. Les relations temporelles fondées sur les instants et les relations fondées sur les intervalles. Dans ces deux types de relations, les scénarios sont représentés à partir d'un ensemble *B* de relations temporelles dites **primitives**. Ces relations sont exclusives et permettent d'exprimer comment les unités temporelles d'un document (les éléments) se situent les unes par rapport aux autres.

III.3.2.1 Relations à base d'instants

Dans les relations à base d'instants (algèbre d'instants[(2)] notée PA : *Point Algebra*), les unités temporelles considérées dans les relations sont les instants de début et de fin des éléments. Étant donnés deux instants dans un scénario, trois relations peuvent exister entre

---

(2) Nous reviendrons dans la section III.4 sur la justification de ce terme d'algèbre.

eux. Un instant peut en précéder un autre (<), lui succéder (>) ou lui être égal (=). L'ensemble des relations primitives de l'algèbre d'instant est noté $B = \{ <, >, = \}$.

Dans certains cas, les relations entre certains instants d'un scénario peuvent être connues de façon moins précise. Dans l'opération d'édition, il existe deux sources à ce genre d'imprécision :

- Le scénario est en cours de construction et l'auteur n'a pas encore placé de façon définitive les éléments les uns par rapport aux autres.
- Des intervalles indéterministes sont présents dans le scénario, ayant comme conséquence le positionnement temporel imprécis entre certains éléments.

Afin d'illustrer le premier type de situations, supposons que la seule relation que nous connaissions entre deux instants *e1* et *e2* est que *e1* ne peut avoir lieu après *e2*. Cela signifie que *e1* a lieu avant ou en même temps que *e2*. Cette relation est notée $e1 < e2 \vee e1 = e2$ ou encore $e1 \{<, = \} e2$.

L'ensemble des **relations composées** ou disjonctives, noté 2^B, représente des disjonctions des relations primitives. Puisqu'il existe 3 relations primitives, il en résulte $2^3 = 8$ relations composées qui forment les parties de *B*. Dans la suite, chacune de ces relations sera notée par un symbole particulier. Par exemple, au lieu de noter $e1 \{<, =\} e2$, nous utiliserons $e1 \leq e2$. Les relations composées à base d'instants sont : $\perp, \leq, <, =, >, \geq, \neq$, ?, où la relation ? représente une disjonction contenant l'ensemble des relations primitives et $\perp$ l'ensemble vide des relations[(3)].

De façon intuitive les relations primitives <, >, = sont utiles pour la composition multimédia car elles interviennent dans la construction des scénarios multimédia les plus simples. En est–il de même pour toutes les formes disjonctives $\{\leq, \geq, \neq, ?\}$? Afin de répondre à cette question, nous devons d'abord tenir compte des caractéristiques temporelles d'une présentation multimédia. Dans le chapitre précédent, nous avons montré que de légères fluctuations sur l'instant de début des éléments et sur leur durée (domaine de validité) sont tolérées. Par exemple, la présentation d'un élément audio une milli seconde plus tard ou plus tôt que sa date prévue n'affecte en rien la qualité de la présentation. Ceci signifie qu'il n'y a pas de différence perceptible si on spécifie entre deux instants *e1* et *e2* la relation $e1 < e2$ ou $e1 \leq e2$. De façon similaire la relation $\neq$ se démarque de la relation ? sur un seul instant du temps. Les quatre relations d'instants : $\{ <, =, >, ? \}$ sont donc les seules qui sont utiles pour la spécification de scénarios multimédia [113][78].

---

(3) Cette relation traduit l'impossibilité de relier temporellement deux instants : l'incohérence.

III.3.2.2 Relations à base d'intervalles

Dans les relations à base d'intervalles (algèbre d'intervalles notée IA : *Interval Algebra*), les relations possibles entre deux éléments multimédia se réduisent à toutes les combinaisons de positionnement possibles de deux intervalles sur une droite orientée. Le modèle le plus général, proposé par J.F. Allen [3], dresse la liste exhaustive de toutes ces relations. La liste des combinaisons possibles entre éléments multimédia comporte 13 relations consistant en 7 relations de base (cf. Fig. 3.4) et de leurs relations inverses, l'égalité étant elle–même son inverse. Ces treize relations se répartissent en deux classes, celle des relations de séquentialité et celle des relations introduisant le parallélisme de présentation. Dans l'avant–dernière colonne du tableau, nous présentons la traduction de chaque relation sous la forme d'une suite de relations à base d'instants. Les variables x et y représentent des intervalles et les notations x^- et x^+ correspondent respectivement aux instants de début et de fin de l'intervalle x.

Relation	**Symbole**	**Inverse**	**Relation à base d'instants équivalente au symbole**	**Classe**
x before y	b	bi	$x^- < x^+ < y^- < y^+$	Seq
x meets y	m	mi	$x^- < x^+ = y^- < y^+$	Seq
x overlaps y	o	oi	$x^- < y^- < x^+ < y^+$	Par
y finishes x	f	fi	$x^- < y^- < x^+ = y^+$	Par
y during x	d	di	$x^- < y^- < y^+ < x^+$	Par
x starts y	s	si	$x^- = y^- < x^+ < y^+$	Par
x equals y	e	ei	$x^- = y^- < x^+ = y^+$	Par

Fig. 3.4 : Les relations à base d'intervalles

De façon analogue aux relations à base d'instants, il existe $2^{13} = 8192$ relations composées. Par exemple, si l'auteur d'un scénario veut décrire que deux intervalles A et B sont disposés séquentiellement dans le temps, la relation qui les lie correspond à la forme disjonctive suivante : A {b, bi, m, mi } B. Cette relation consiste simplement à éliminer les relations de classe parallèle.

Afin de mesurer l'apport de l'algèbre d'intervalles au multimédia, nous commençons d'abord par la comparer avec l'algèbre d'instants. Le tableau de la Fig. 3.4 indique que chaque relation primitive à base d'intervalles est équivalente à plusieurs relations d'instants. Il est alors légitime de se demander s'il en est de même pour les relations composées

ou disjonctives à base d'intervalles. Auquel cas, ces deux représentations seraient équivalentes [108]. Le tableau de la figure Fig. 3.5 donne un résumé du nombre de relations disjonctives de IA qui peuvent être générées à partir d'un sous−ensemble des relations composées de PA.

Relations à base d'instants de PA	**Nombre de disjonctions de IA pouvant être générées**
<, =, >	13
<, =, >, ?	29
≤, <, =, >, ≥, ?	82
≤, <, =, >, ≥, ?, ≠	187

Fig. 3.5 : Relations disjonctives de IA engendrées par les relations de PA

La dernière ligne montre que seules 187 relations sont exprimables à partir des relations à base d'instants : c'est l'algèbre d'intervalles restreinte qu'on notera RIA (*Restricted Interval Algebra*). Ce fait témoigne de la plus grande richesse des relations de IA par rapport à celle de PA. Sur les 8192−187 = 8004 relations non exprimables moyennant PA, il existe plusieurs relations qui peuvent être utiles dans un scénario multimédia. En particulier, la relation A {b, bi, m, mi} B qui dénote l'exclusion mutuelle entre deux intervalles. Cette relation peut être utilisée pour traduire l'impossibilité de présenter A et B en même temps. Par exemple, si ces derniers nécessitent la même ressource matérielle (carte de décompression, haut−parleur, etc.).

Notons que les relations d'instants jugées utiles pour le multimédia dans PA engendrent uniquement 29 relations d'IA (deuxième ligne du tableau Fig. 3.5). Plusieurs auteurs [78][113][96] les retiennent d'ailleurs comme une référence pour mesurer l'expressivité d'un langage de synchronisation temporelle multimédia(4).

III.3.2.3 Choix entre algèbres d'intervalles et d'instants

Dans le cadre des systèmes d'édition de documents multimédia, plusieurs arguments plaident en faveur d'une spécification des documents multimédia à base d'intervalles :

- Les opérations d'édition de documents portent sur les éléments de base ou composés dont la projection dans le temps correspond à des intervalles.
- La description du scénario est syntaxiquement plus compacte.
- Elle répond mieux au critère d'encapsulation et de modularité puisqu'il est possible d'une part de délimiter et de regrouper des portions d'un scénario sous

(4) Nous renvoyons le lecteur à ces travaux pour une étude plus détaillée de ces 29 relations.

forme d'un seul intervalle fictif (sous–scène), et d'autre part d'isoler les éléments et les relations qu'il contient afin de le relier aux autres portions du scénario.

En revanche, la représentation à base d'instants fournit un très bon support d'exécution pour la présentation. En effet, on peut considérer que l'état du système de présentation ne change pas de façon continue dans le temps mais évolue lors de l'apparition d'événements discrets (on parle de Système Dynamique à Événements Discrets (SDED) [96]). Dans ce cas, la représentation d'un scénario à base d'instants permet de manipuler une structure interne de type graphe, qui peut être exploitée comme une description des tâches à ordonnancer lors de la présentation.

III.3.3 Expression des relations et langage temporel multimédia

Après avoir présenté et analysé la nature des relations temporelles, nous nous proposons dans cette section d'étudier la capacité des deux langages temporels à modéliser un scénario multimédia. Cela revient à poser la question : est–ce que la sémantique attachée à ces relations est suffisante pour traduire tous les schémas de synchronisation multimédia ?

Afin de répondre à cette question, il est nécessaire d'analyser les différents besoins qui apparaissent lorsqu'on cherche à spécifier un scénario multimédia. Notre analyse nous a conduit à séparer trois types de besoins :

- Exprimer des informations **qualitatives** : L'auteur doit pouvoir positionner les éléments dans le temps en se basant uniquement sur des informations de placement relatif, sans contraintes numériques, par exemple x avant y.
- Exprimer des informations **quantitatives** : Il doit pouvoir aussi manipuler des informations numériques, par exemple x avant y de 3 secondes.
- Exprimer des informations **causales**.

En ce qui concerne les informations causales, il est nécessaire dans certains cas de pouvoir exprimer des scénarios dans lesquels l'occurrence d'un instant de fin d'un élément provoque l'arrêt d'un autre, même si ce dernier n'a pas restitué tout son contenu à l'utilisateur. Pour le cas d'une représentation à base d'intervalles, nous avons identifié trois relations pertinentes de ce type :

- La relation *Parmin(A, B)* dans laquelle les deux éléments démarrent en même temps et le premier des deux éléments qui se termine provoque la terminaison de la construction.
- La relation *Parmax(A, B)* dans laquelle les deux éléments démarrent en même temps et le dernier des deux éléments qui se termine provoque la terminaison de la construction.

- La relation *Parmaster(A_m, B)* dans laquelle les deux éléments démarrent en même temps et c'est la terminaison de l'élément désigné par l'auteur (dans ce cas c'est A_m) qui provoque la terminaison de la construction.

À titre d'exemple, dans un scénario composé d'une séquence vidéo et d'une séquence musicale, si l'auteur souhaite que ces deux éléments se terminent en même temps même s'ils ont des durées différentes, il lui suffit d'introduire la relation Parmin(audio, vidéo). Cette relation ne peut être exprimée ni par l'algèbre d'instants, ni par l'algèbre d'intervalles. Si ces algèbres permettent de construire des scénarios non interactifs, les relations causales permettent d'intégrer une certaine forme d'interactivité. Par exemple, si l'auteur souhaite décrire un scénario où l'activation d'un bouton permet l'interruption d'une vidéo, il lui suffit d'introduire la relation suivante : Parmaster($Bouton_m$, vidéo).

III.4 Analyse et synthèse de scénarios temporels

Les problèmes liés à la représentation et à la composition d'éléments multimédia ressemblent, sur beaucoup d'aspects, à des problèmes rencontrés dans d'autres domaines de l'informatique. Parmi ces domaines, les plus proches du multimédia et plus particulièrement de l'édition de documents, concernent la représentation et la gestion des connaissances temporelles [77], l'ordonnancement de tâches [74] et la planification [110]. Parmi ces travaux, l'algèbre d'intervalles d'Allen [3] et l'algèbre d'instants proposée par Kautz [52] ont beaucoup marqué la recherche en multimédia, en particulier en ce qui concerne l'expression des relations temporelles, comme nous l'avons présenté dans la section III.3.2. En revanche, les travaux portant sur la synthèse et l'analyse des scénarios temporels, qui sont au cœur du processus d'édition, restent de nos jours encore très peu abordés. Pourtant beaucoup de travaux, dont ceux d'Allen [4] et Kautz [52], ont largement étudié ces deux aspects dans le cadre des systèmes temporels à base de contraintes. Dans la suite de ce chapitre, nous montrons qu'ils constituent un cadre pertinent pour aborder l'édition de documents multimédia.

Dans cette section, nous étudions les fondements théoriques des modèles temporels basés sur les contraintes. Cette étude, qui constitue l'originalité de notre démarche, place le problème de l'édition multimédia comme un cas particulier appartenant à une classe de problèmes plus généraux appelés *CSP* (*Constraint Satisfaction Problem*).

III.4.1 Introduction aux CSP

Dans le domaine des *CSP*, la notion de **contrainte** désigne un énoncé déclaratif exprimant une mise en relation entre les données d'un problème. Une contrainte constitue donc un élément de représentation de la connaissance relative à un problème. Un système de gestion de contraintes est l'outil qui permet d'une part d'exprimer dans un formalisme déclaratif les contraintes qui décrivent un problème, et qui fournit d'autre part des algorithmes d'analyse et de synthèse capables de déterminer sa cohérence et de le résoudre s'il admet une solution. L'approche fondée sur les contraintes allie donc expression du problème (aspect langage) et techniques algorithmiques de résolution (gestion des contraintes). Justement, ces deux éléments correspondent parfaitement aux besoins d'édition que nous visons.

Un *CSP* est défini par la donnée d'un ensemble de variables, chacune associée à un domaine de valeurs, et par la donnée d'un ensemble de contraintes liant ces variables. Une contrainte est généralement décrite par l'ensemble de combinaisons de valeurs qui la satisfont. Une solution d'un *CSP* est une instantiation de ces variables qui satisfait simultanément toutes les contraintes d'un problème. Nous nous intéressons par la suite uniquement aux CSP binaires, c'est–à–dire ne manipulant que des contraintes entre deux variables.

Formellement, un **problème** de satisfaction de contraintes binaires $P = (X, D, C, R)$ est défini par :

- un ensemble de **variables** $X = \{X_1 , ..., X_n\}$;
- un ensemble de **domaines** $D = \{D_1, ..., D_n\}$ où D_i est le domaine de la variable X_i ;
- un ensemble de **contraintes** $C = \{C_{i,j}, / 1 \leq i, j \leq n \}$;
- un ensemble de **relations** $R = \{R_{i,j} / 1 \leq i, j \leq n\}$; $R_{i,j}$ est l'ensemble des combinaisons de valeurs qui satisfont la contrainte $C_{i,j}$.

Une **solution** σ d'un tel CSP est définie par :

$$\sigma = \{x_1 \in D_1, ...,x_n \in D_n\} / \forall C_{i,j} \in C, (x_i, x_j) \in R_{i,j}$$

Un CSP est **cohérent** si et seulement si il existe au moins une solution qui le satisfait.

On peut associer à tout CSP binaire un réseau où les sommets sont les variables et les arcs représentent les contraintes étiquetés par les domaines des variables. On qualifie de **minimal** un tel réseau lorsque les domaines des variables sont restreints aux seules valeurs appartenant à au moins une solution.

Dans le domaine du multimédia, les mérites que nous attribuons à une approche fondée sur les contraintes peuvent être résumés ainsi :

1. Elle permet une **description formelle** de la synchronisation temporelle des documents multimédia.
2. Elle permet une **expression déclarative** de la synchronisation temporelle. Ce type d'expression rend possible la construction de langages de synchronisation plus faciles à utiliser que les langages impératifs comme les scripts.
3. Elle permet à l'auteur d'un document de se focaliser sur l'édition d'un document (la formulation de ce qu'il veut obtenir) sans se préoccuper de la façon dont il va l'obtenir (la résolution ou le formatage).

Dans cette section, nous établissons le lien qui existe entre le problème de synchronisation d'un document multimédia et les systèmes de contraintes *CSP*. Nous commençons d'abord par un rappel des principales définitions relatives à la théorie des *CSP*. Ces définitions nous seront utiles lorsque nous aborderons les techniques de vérification de la cohérence d'un scénario et de recherche de solutions qui sont présentées dans les sections III.4.3 et III.4.4.

III.4.2 TCSP : les CSP temporels

La synchronisation temporelle multimédia peut être abordée dans le cadre d'une classe particulière des *CSP* désignée par *TCSP* (*Temporal Constraints Satisfaction Problem*). Dans cette classe, les variables modélisent des instants ou des intervalles temporels de durée, et les relations leurs placements relatifs à travers le temps. Il existe deux grandes familles de techniques permettant de gérer les contraintes temporelles :

- La gestion d'un scénario à partir de relations **qualitatives** ou symboliques (*TCSP* symboliques).
- La gestion d'un scénario à partir de relations **quantitatives** ou numériques (*TCSP* numériques).

En réalité, ces deux types de représentation sont nécessaires dans les systèmes multimédia, car dans un processus d'édition, l'auteur peut vouloir exprimer des relations de synchronisation sous forme symbolique, mais leurs conséquences dans le scénario peuvent se traduire sous forme de contraintes numériques. Ces mêmes contraintes peuvent à leur tour limiter les relations symboliques que l'auteur peut vouloir ajouter. Ceci explique la difficulté du problème de synchronisation des documents multimédia qui combine ces deux types de représentation.

III.4.3 Gestion symbolique des relations temporelles

La gestion symbolique des contraintes temporelles, introduit par Allen, s'appuie sur l'organisation des relations sous forme d'un **graphe temporel**. Les sommets sont des symboles qui représentent des intervalles, et les arcs sont des relations temporelles disjonctives d'IA (voir section III.3.2.2).

La gestion symbolique des relations (analyse et synthèse) consiste à appliquer un mécanisme qui s'apparente à une activité de déduction. Ces déductions aboutissent au retrait de certaines disjonctions des relations du graphe, ce qui revient à les préciser. C'est cette caractéristique qui nous fait donner le nom de symbolique à ce type de système, par opposition aux systèmes numériques dont la gestion porte sur des contraintes numériques (domaines de validité des intervalles).

Pour les deux types de relations à base d'instants où d'intervalles, nous nous intéressons aux deux aspects suivants :

- Les aspects liés à la puissance expressive des relations symboliques.
- Les aspects liés à la gestion des contraintes temporelles symboliques, en particulier le problème de **cohérence** d'une spécification et de recherche d'une **solution**.

Nous allons d'abord présenter ces deux types de langage permettant l'expression des contraintes temporelles. En particulier, nous décrivons les mécanismes de composition qui leur sont associés ainsi que les avantages ou les inconvénients de gérer un document multimédia en adoptant l'une ou l'autre des deux méthodes.

III.4.3.1 Loi de composition temporelle

Dans les deux algèbres d'instants et d'intervalles, chaque relation de 2^B (l'ensemble des relations composées) est inversible et son inverse est une relation de 2^B ; par exemple, x *before* y est l'inverse de y *before* x dans l'algèbre d'intervalles. Pour chaque relation x R y de l'ensemble 2^B entre deux éléments temporels x et y, on définit la relation inverse notée y R^{-1} x.

Pour permettre de combiner des connaissances temporelles exprimées par de telles relations, deux opérations sont définies sur les éléments de l'ensemble 2^B :

- L'**intersection** $\cap$ de deux éléments de l'ensemble 2^B est un élément de l'ensemble 2^B.
- L'**union** $\cup$ de deux éléments de l'ensemble 2^B est un élément de l'ensemble 2^B.

- De plus, ? est l'**élément neutre** de l'intersection et l'élément absorbant de l'union.
- $\perp$ est l'**élément neutre** de l'union et l'élément absorbant de l'intersection.

On définit également une loi de composition interne $\otimes$, associative, non commutative d'élément absorbant ? Son élément neutre est l'égalité (= dans PA et equals dans IA). Si R1 et R2 sont deux relations de 2^B, nous obtenons une nouvelle relation R3(x, z) = R1(x, y) $\otimes$ R2(y, z) qui fait aussi partie de 2^B. Cette loi permet d'inférer, à partir de la relation R1(x, y) et R2 (y, z) la relation R1 $\otimes$ R2(x, z).

Les trois opérations, $\otimes$, $\cap$ et $\cup$ sont deux à deux distributives. Qu'il s'agisse d'instants ou d'intervalles { 2^B, $\cup$, $\otimes$ } est une structure d'algèbre. C'est pour cette raison qu'on parle d'**algèbre** d'instants ou **d'algèbre** d'intervalles.

III.4.3.2 Composition dans l'algèbre d'instants

La loi de composition $\otimes$ s'applique de la façon suivante dans PA. Par exemple, si nous avons trois instants temporels x, y et z avec les relations x < y et y = z, on déduit la nouvelle relation x < z (voir le tableau de la Fig. 3.6).

$\otimes$	<	=	>
<	<	<	?
=	<	=	>
>	?	>	>

Fig. 3.6 : Loi de composition de PA pour les relations primitives

En utilisant la distributivité de $\otimes$, le même tableau peut être utilisé pour les relations de 2^B. Par exemple, à partir des deux relations : x {<, =} y et y {<, =} z, on déduit :

$$x [\{<, =\} \otimes \{<, =\}] z = x [(<\otimes<) \cup (<\otimes=) \cup (=\otimes<) \cup (=\otimes=)] z$$

$$= x [< \cup < \cup < \cup =] z$$

$$= x [\{<, =\}] z$$

Ainsi, si l'auteur d'un document veut exprimer un scénario temporel multimédia moyennant l'algèbre d'instants, il construit une base d'instants temporels qui correspondent aux différents événements du scénario. Ensuite, il les relie temporellement en spécifiant des relations entre eux.

Par exemple, supposons que dans l'exemple de déduction ci–dessus, l'événement x corresponde au début d'une vidéo, y au début d'un élément audio et z au début d'un élément texte. Nous avons montré à travers l'exemple précédent qu'une application de la loi de composition entre les relations interdit d'introduire la relation x > z, au risque de rendre le scénario incohérent.

III.4.3.3 Composition avec l'algèbre d'intervalles

Pour faire de la vérification de cohérence, Allen suggère [3] l'utilisation des lois de composition appliquées aux intervalles afin d'exprimer un scénario temporel et de raisonner à son propos. L'expression d'une relation de synchronisation qui lie un élément temporel à un autre est alors simplement traduite sous forme de relation symbolique entre les deux éléments.

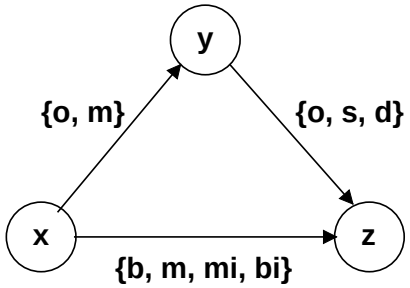


Fig. 3.7 : Graphe de contraintes de l'exemple 1

Exemple 1 : Soient trois éléments multimédia x, y et z. Supposons que l'auteur d'un scénario veut exprimer les contraintes temporelles suivantes (voir Fig. 3.7) :

- L'élément x chevauche temporellement y ou démarre juste avant :
  x {o, m} y.
- L'élément x ne se joue pas en même temps que l'élément z :
  x {b, m, mi, bi} z.
- L'élément y chevauche, démarre en même temps ou se joue pendant l'élément z :
  y {o, s, d} z.

Comme pour l'algèbre de points, le pouvoir de déduction du système d'Allen repose sur l'établissement d'une table de transitivité 7x 7 (voir Fig. 3.8) entre les relations primitives, permettant de calculer R3 dans la formule :

$$x\ R1\ y \otimes y\ R2\ z \Rightarrow x\ R3\ z$$

En général, la relation R3 est disjonctive, car la connaissance de R1 et de R2 ne donne pas toujours assez d'information pour que R3 soit suffisamment précise.

Lorsque R1 et R2 sont elles−mêmes disjonctives, l'algorithme de calcul de R3 est très simple :

1. Décomposer R1 et R2 en une disjonction de relations élémentaires.
2. Développer tous les termes en appliquant la distributivité de $\cap$ par rapport à $\cup$. On obtient ainsi des clauses qui sont des paires conjonctives.
3. Calculer grâce à la table de la Fig. 3.8 le résultat de chaque paire conjonctive.
4. Les conjonctions ayant disparu, composer les clauses disjonctives en une seule relation composée.

R1 ⊗ R2	**b**	**m**	**s**	**f**	**d**	**o**	**e**
b	b	b	b	b	b,m,s,d,o	b	e
m	b	b	m	b,m,s,d,o	s,d,o	b	b
s	b	b	s	d	d	b,m,o	m
f	b	m	d	f	d	s,d,o	s
d	b	b	d	d	d	b,m,s,d,o	f
o	b	b	o	s,d,o	s,d,o	b,m,o	d
e	b	m	s	f	d	o	o

Fig. 3.8 : Loi de composition d'IA

L'algorithme proposé par Allen permet la détection d'incohérences temporelles. Son principe est le suivant : à partir de la table de transitivité ci−dessus, on calcule toutes les relations induites et on les confronte aux relations existantes, ceci permet d'obtenir des relations déduites, plus précises. L'algorithme propage ces relations par application d'une autre transitivité et ainsi de suite jusqu'au repos.

Ce principe s'applique pour les deux types de relations, d'instants ou d'intervalles. Dans les deux cas, le problème est représenté sous forme d'un graphe (S, R) où $S=\{X_i,...,X_j\}$ est l'ensemble des variables et $R_{i,j}$ est l'ensemble des contraintes entre les variables X_i et X_j. L'algorithme nécessite un graphe complet : les arcs entre variables qui ne sont liées initialement par aucune relation sont étiquetées par la relation ?

Initialisation : Pour i, j ∈ [1, n], i ≠ j :

Q ← {i, j} *Q est la pile des relations en cours de propagation*

Propagation : Tant que Q ≠ ∅

1) {i, j} ← dépiler(Q) *On considère un arc*

2) Pour k ∈ [1, n], k ≠ i, j

– *On calcule l'influence de l'arc courant sur les autres arcs*

$R'_{ik} \leftarrow (R_{ij} \otimes R_{jk}) \cap R_{ik}$

$R'_{kj} \leftarrow (R_{ki} \otimes R_{ij}) \cap R_{kj}$

– Si R'_{ik} ou $R'_{kj} = \perp$ Alors STOP *nous avons détecté une incohérence*

– *Si la nouvelle relation diffère de l'ancienne, il faut la propager*

Si $R'_{ik} \neq R_{ik}$ Alors empiler {i, k} dans Q; $R_{ik} \leftarrow R'_{ik}$

Si $R'_{kj} \neq R_{kj}$ Alors empiler {j, k} dans Q; $R_{kj} \leftarrow R'_{kj}$

Cet algorithme permet de préciser de plus en plus le graphe en éliminant des disjonctions. Étant donné que le nombre de relations et la taille du graphe sont finis, et que ce graphe est borné inférieurement par le graphe minimal, l'algorithme s'arrête toujours : soit parce qu'il n'y a plus de relations apportant une information nouvelle, soit parce qu'une relation composée devient vide, auquel cas on a détecté l'incohérence du scénario.

L'application de cet algorithme à l'exemple 1 aboutit à la suppression de trois relations incohérentes (voir les relations barrées de la Fig. 3.9). Cette méthode d'analyse a été proposée pour les documents multimédia par Kshirasagar [58].

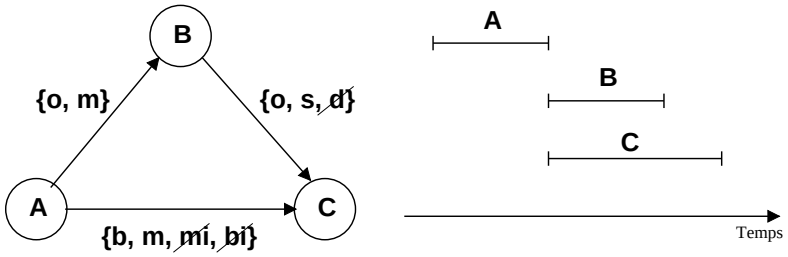


Fig. 3.9 : Graphe minimal et scénario cohérent pour l'exemple 1

III.4.3.4 Discussion

L'approche proposée par Allen [3] constitue le principal travail accompli dans le cadre de la résolution d'un *TCSP* symbolique. Cet algorithme, polynômial, n'est malheureusement pas complet. En effet, Vilain et Kautz [112] ont montré que l'algorithme d'Allen

pouvait ne pas détecter certaines incohérences et que le problème d'existence d'une solution dans IA est NP−complet. Ce qui signifie que, pour rechercher une solution, il faut utiliser des méthodes de recherche exhaustives. Ces techniques, même appliquées à des problèmes de petite taille, restent pour le cas des systèmes multimédia coûteuses car elles sont de complexité exponentielle.

En conclusion, si le modèle proposé par Allen permet une représentation plus riche que la représentation à base d'instants, c'est au prix de l'incomplétude et d'un accroissement important de la complexité algorithmique.

III.4.4 Gestion numérique des relations temporelles

Contrairement à l'approche symbolique, les systèmes numériques manipulent explicitement des nombres(5), ce qui leur permet de couvrir l'aspect métrique du temps. Le modèle précédent représente et traite le temps uniquement à travers une représentation symbolique (traitement qualitatif). Cette représentation bien que très utile pour l'édition ne prend pas en compte des informations numériques. En particulier, dans la construction d'un scénario, on peut être intéressé par les trois opérations suivantes :

- Déterminer si un scénario est **quantitativement cohérent** : étant donné les valeurs possibles des durées des intervalles, il existe une valuation de ces durées qui satisfait toutes les contraintes.
- Retrouver toutes les valeurs d'un intervalle susceptibles de figurer dans une solution : c'est la contrainte numérique minimale. Cette donnée permet d'indiquer à l'auteur, compte tenu des contraintes déjà introduites, dans quelle mesure il peut modifier la durée d'un intervalle tout en préservant la cohérence du scénario.
- Effectuer le formatage temporel, ce qui revient à trouver une solution spécifique pour un scénario cohérent donné.

III.4.4.1 Réseaux de contraintes numériques TCSP et STP

Les contraintes temporelles liées aux domaines de validité et aux relations temporelles symboliques peuvent s'exprimer sous forme de différences bornées entre dates d'instants. En partant de cette constatation, Dechter et Meiri [27] proposent de représenter les problèmes temporels sous forme de *TCSP* numériques.

Un *TCSP* numérique est défini par un ensemble de variables X_1, .., X_n représentant des instants ayant des valeurs de dates dans $\aleph$, et un ensemble de contraintes binaires. Une contrainte $C_{i,j}$ entre les variables X_i et X_j, est une disjonction de différences bornées du type :

---

(5) Dans notre cas, on s'intéresse uniquement à des nombres entiers

$$a^1_{i,j} \leq X_i - X_j \leq b^1_{i,j} \vee ... \vee a^n_{i,j} \leq X_i - X_j \leq b^n_{i,j}$$

Par conséquent, une contrainte $C_{i,j}$ définit un ensemble d'intervalles de validité $\{I^1_{i,j}, ..., I^n_{i,j}\}$ où $I^k_{i,j} = [a^k_{i,j}, b^k_{i,j}]$. Un élément multimédia peut donc être modélisé dans ce formalisme par une union de domaines de validité. En fait, le seul cas de *TCSP* qui peut être résolu en temps polynômial est celui d'un *TCSP* ne comportant qu'un intervalle par arc (le problème *STP* : *Simple Temporal Problem*). Cela correspond à la modélisation de chaque élément multimédia par un seul domaine de validité. Pour cette raison, nous allons nous restreindre dans la suite de cette étude aux cas des *STP*.

Le problème de satisfaction de contraintes se présente dans les *STP* par un graphe conjonctif de différences bornées. Dans ce type de graphe, chaque arc reliant deux instants temporels i et j est étiqueté par des bornes $[a_{ij}, b_{ij}]$ et indique la présence d'une contrainte de distance entre i et j :

$$a_{ij} \leq X_i - X_j \leq b_{ij}$$

Cette contrainte peut aussi être décrite par une paire d'inégalités :

$$X_i - X_j \leq b_{ij}$$

$$\text{et } X_i - X_j \leq -a_{ij}$$

Chaque contrainte $C_{j,i}$ d'un sommet j à un sommet i peut être retrouvée à partir de $C_{i,j}$: $C_{j,i} = [-b_{ij}, -a_{ij}]$, en général une seule de ces contraintes est représentée dans le graphe. De plus, un instant particulier, défini par le sommet 0 et la variable X_0 est ajouté au graphe pour représenter le début du scénario. Toutes les valeurs X_i sont définies par rapport à X_0 et chaque contrainte $C_{0,i}$ représente la contrainte de distance par rapport au début du scénario, on notera cette contrainte C_i et on considère pour des raison de simplicité que $X_0 = 0$.

En conséquence, la résolution d'un problème *STP* revient à la résolution d'un problème décrit par ensemble d'inéquations linéaires sur les variables $X_0, ..., X_n$.

III.4.4.2 Systèmes utilisant la programmation linéaire

Le problème de résolution d'un système linéaire d'inégalités est bien connu dans le domaine de la recherche opérationnelle. Il peut en effet être résolu en temps exponentiel moyennant l'algorithme du simplex [26] ou par l'algorithme de Khachiyan [53]. L'algorithme du simplex optimise n'importe quelle fonction linéaire sur un domaine exprimé au moyen de contraintes linéaires. Dans le problème d'analyse d'un scénario, il n'y a *a priori* pas de fonction à optimiser. On cherche seulement à définir les domaines de validité des éléments multimédia pour le scénario en construction. Par contre, pour le formatage on peut toujours considérer qu'une solution qui s'écarte le moins possible des valeurs

optimales des éléments est celle qu'il faut rechercher en priorité. Dans ce cas, il est possible de définir une fonction coût, à minimiser, qui permet de guider le choix de la solution. S'il n'existe pas de solution, l'ensemble de contraintes est incohérent. De plus, on a toujours "sous la main" une solution totalement explicitée, mais les domaines des variables eux ne sont pas explicites. Par conséquent, l'auteur ne sait pas dans quelle mesure les valeurs des variables dans la solution exhibée sont modifiables tout en restant une solution au problème.

Ce genre de méthodes permet d'utiliser toute la puissance de représentation de l'algèbre linéaire. Un point révélateur est que les contraintes sont rarement présentées du point de vue d'un graphe temporel. De plus, on ne peut pas vraiment considérer le simplex comme une méthode d'analyse de scénarios, dans la mesure où l'algorithme ne fait qu'exhiber une solution. Dans le cas d'incohérence, l'algorithme rejette en bloc l'ensemble des contraintes, ce qui empêche l'auteur de localiser avec précision la source de l'incohérence.

Quant à la complexité de cette méthode, elle est fortement alourdie par le coût théoriquement exponentiel de l'algorithme du simplex, qu'il faut faire fonctionner chaque fois que l'on ajoute une contrainte (il n'est pas incrémental). C'est cette solution qui a été adoptée dans les deux seuls systèmes multimédia ayant implanté un module de recherche automatique de solution ou formateur temporel multimédia : *Firefly* [14] et plus récemment *Isis* [55].

III.4.4.3 Systèmes basés sur les réseaux de contraintes

Nous avons vu que non seulement les relations temporelles s'expriment de façon linéaire, mais aussi que les intervalles de validité peuvent être représentés de la même façon (inégalités sur la distance entre instants de début et fin). Les approches reposant sur les réseaux de contraintes partent du constat que les *STP* admettent des solutions plus efficaces que celles présentées dans la section précédente : les inéquations représentant les contraintes temporelles sont structurées sous forme d'un graphe auquel une algorithmique de graphe peut être appliquée.

Formellement, à chaque problème STP est associé un graphe étiqueté orienté appelé graphe de distance :

Définition 1 : Un graphe de distance temporelle $G_d = (V, E_d)$ est défini par un ensemble de sommets 0, .., n représentant des variables $V = \{X_0, ..., X_n\}$ et un ensemble d'arcs E_d, où chaque arc $X_i \rightarrow X_j$ relie deux sommets distincts X_i et X_j du graphe. Chaque arc est étiqueté par une valeur entière. Si $[a_{i,j},b_{i,j}]$ est l'intervalle de validité associé à la distance entre X_i et X_j, alors l'arc $X_i \rightarrow X_j$ est étiqueté par $b_{i,j}$ et l'arc $X_j \rightarrow X_i$ est étiqueté par $-a_{i,j}$.

Chaque chemin d'un sommet i à un sommet j dans le graphe G_d, $i = i_0, ..., i_k = j$, introduit la contrainte temporelle suivante sur la distance $X_j - X_i$:

$$X_i - X_j <= \sum_{j=1}^{k} a_{i_{j-1}, i_j}$$

S'il existe plus d'un chemin de i à j, alors il est facile de vérifier que l'intersection de toutes les contraintes de chemins donne :

$$X_j - X_i \leq d_{ij}$$

Où d_{ij} représente la durée du plus court chemin entre i et j. En se basant sur cette observation, la condition fondamentale suivante pour la cohérence d'un problème *STP* peut être établie :

Théorème : (Shostak [97], Liao et Wong [68], Leiserson et Saxe [67]) Un *STP* est cohérent si et seulement si son graphe de distance G_d ne contient pas de cycle négatifs.

Preuve : Supposons qu'il existe un chemin négatif C formé des sommets $i_1, i_2, ..., i_k=i_1$. La somme des inégalités le long du chemin C donne :

$X_{i1} - X_{i1} < 0$, qui ne peut évidemment pas être satisfaite.

Dans l'autre sens de l'équivalence, s'il n'existe aucun cycle négatif dans G_d, alors le plus court chemin entre chaque paire de sommets est bien défini. Pour chaque paire de sommets i et j, le plus court chemins satisfait $d_{0j} \leq a_{ij} + d_{0i}$; d'où,

$$d_{0j} - d_{0i} \leq a_{ij}$$

Donc le tuple $(d_{01}, ..., d_{0n})$ est une solution du *STP*.

Corrolaire : Soit G_d le graphe de distance d'un *STP* cohérent. Il existe aux moins deux scénarios cohérents qui sont :

$$S_1 = (d_{01}, ..., d_{0n}) \text{ et } S_2 = (-d_{10}, ..., -d_{n0})$$

Ces deux scénarios représentent une instantiation des variables à leur date d'occurrence au plus tôt et au plus tard respectivement.

Deux problèmes *STP* sont dits équivalents s'ils représentant le même ensemble de solutions, et un *STP* est dit minimal s'il n'existe aucun *STP* équivalent dont les contraintes soient plus restrictives.

Algorithmes utilisant les graphes

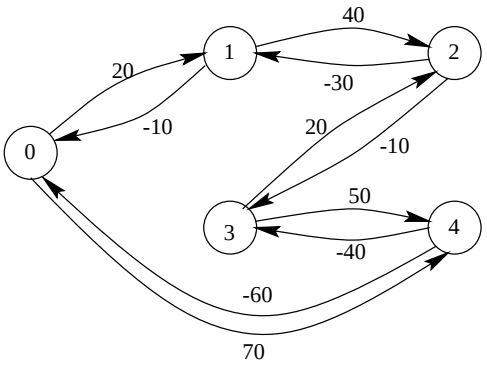
Le graphe de distance d'un *STP* peut être construit en appliquant l'algorithme du plus court chemin (*all pairs shortest paths*) de Floyd–Warshal [85]. Cet algorithme peut être considéré comme un algorithme de relaxation : à chaque pas la valeur d'un arc est mise à jour d'un montant qui dépend uniquement de celles des arcs adjacents.

Algorithme du plus court chemin

1. Pour i := 1 à n faire $d_{ii} \leftarrow 0$
2. Pour i, j := 1 à n faire $d_{ij} \leftarrow a_{ij}$
3. Pour k := 1 à n faire
4. Pour i, j := 1 à n faire
5. $d_{ij} \leftarrow \min \{d_{ij}, d_{ik} + d_{kj}\}$

Fig. 3.10 : Algorithme all pairs shortest paths

L'algorithme s'exécute en $O(n^3)$ et détecte les cycles négatifs simplement en examinant le signe des éléments diagonaux d_{ii}. Considérons l'exemple de la Fig. 3.11, puisqu'il n'y a pas de cycle négatif, le *STP* correspondant est cohérent. Les plus courts chemins d_{ij} sont présentés dans le tableau suivant :



	0	1	2	3	4
0	0	20	50	30	70
1	–10	0	40	20	60
2	–40	–30	0	–10	30
3	–20	–10	20	0	50
4	–60	–50	–20	–40	0

Fig. 3.11 : Longueur des chemins les plus courts de l'exemple

L'algorithme de Floyd–Warshal constitue ainsi un algorithme polynômial pour déterminer la cohérence du *STP* et le calcul des domaines de validité minimaux et donc le graphe minimal. Une fois que ce graphe est disponible, l'instantiation d'une solution ou le formatage temporel peut prendre $O(n^2)$, car chaque instantiation d'une valeur doit être conforme aux instantiations précédentes et ne sera plus modifiée ultérieurement (cf. section III.4.4.4).

Algorithmes manipulant explicitement les domaines de validité

Cette classe d'algorithmes est très proche d'une formalisation du type étiquetage cohérent du graphe introduit par Allen (sans que leurs concepteurs n'en fassent mention toutefois). Les variables sont les arcs du graphe qui représentent les domaines de validité des intervalles contrairement au cas d'Allen où elles représentaient des relations et à l'algorithme de Floyd–Warshal où les étiquettes représentaient des entiers relatifs. En réalité, il existe toute une famille d'algorithmes de ce type développés dans le cadre des systèmes à base de contraintes [77] [110]. Dans cette étude, nous allons seulement en présenter un, appelé PC–2 [74], qui est représentatif de cette classe d'algorithmes et qui permet d'illustrer leur principe de fonctionnement.

Algorithme PC−2 (Macworth [74])

1. Initialisation
2. Q ← Graphe initial
3. Propagation
4. Tant que $Q \neq \emptyset$ Faire
5. 1) extraire un élément de {i, j} ← Q
6. 2) Pour k = 1 à n, $k \neq i, j$ Faire
7. $D'_{ik} \leftarrow D_{ij} \otimes D_{jk} \oplus D_{ik}$
8. $D'_{kj} \leftarrow D_{ki} \otimes D_{ij} \oplus D_{kj}$
9. − Si D'_{ik} ou $D'_{kj} = \perp$ Alors Stop : une incohérence a été détectée
10. − Si $D'_{ik} \neq D_{ik}$ Alors empiler {i, k} dans Q ; $D_{ik} \leftarrow D'_{ik}$
11. − Si $D'_{kj} \neq D_{kj}$ Alors empiler {j, k} dans Q ; $D_{kj} \leftarrow D'_{kj}$

Fig. 3.12 : L'algorithme PC−2

Dans PC–2, la vérification de la cohérence du graphe se fait en même temps que le calcul du graphe minimal. Le graphe minimal est obtenu en appliquant des opérations

algébriques sur les domaines de validité. Ces opérations permettent à chaque étape d'éliminer les valeurs impossibles d'un domaine de validité. Soient deux domaines de validité : $D1 = [d1_{min}, d1_{max}]$ et $D2 = [d2_{min}, d2_{max}]$. On définit les deux opérations suivantes :

- L'intersection, notée
  $D1 \oplus D2 = [d_{rmin} = \max(d1_{min}, d2_{min}), d_{rmax} = \min(d1_{max}, d2_{max})]$
  $D1 \oplus D2 = \perp$ si $d_{rmin} > d_{rmax}$
- La composition, notée
  $D1 \otimes D2 = [d1_{min} + d2_{min}, d1_{max} + d2_{max}]$

À l'échelle du graphe tout entier, une modification d'un domaine de validité peut donc en entraîner une autre. On propage donc ces modifications jusqu'à ce que le réseau soit au repos, ou que l'on détecte une incohérence.

La complexité de l'algorithme PC−2 est cubique $O(n^3)$. PC−2 possède une propriété intéressante qui consiste à réexaminer le graphe localement d'abord. En effet, il arrive souvent que l'insertion d'un élément ou d'une relation n'affecte qu'une portion limitée du graphe, auquel cas, il n'est pas utile de le remettre en cause dans sa totalité (voir les lignes 10−11 de l'algorithme qui permettent d'anticiper dans ces cas l'arrêt de l'algorithme), contrairement à l'algorithme de Floyd−Warshal où chaque modification nécessite de reconsidérer tout le graphe.

III.4.4.4 Décomposabilité d'un STP

A partir de la discussion précédente, nous avons établi que la spécification d'un scénario peut se ramener au maintien de la structure d'un graphe. Ce graphe est une représentation plus explicite, puisqu'il ne retient que les domaines de validité susceptibles de figurer dans une solution.

Étant donné un graphe cohérent, la présentation du scénario correspondant nécessite le choix d'une solution spécifique parmi celles qui sont possibles : c'est l'opération de synthèse ou formatage. Les problèmes *STP* possèdent une propriété importante permettant la recherche efficace d'une solution : **la décomposabilité**. Un réseau de contraintes *STP* est dit décomposable si chaque instantiation d'une valeur de durée pour un élément n'est jamais remise en cause lors du choix des valeurs des durées des autres éléments du scénario.

Le théorème proposé par Meiri établit que tout réseau de contraintes *STP* est décomposable. C'est un résultat fondamental sur lequel repose notre mise en œuvre.

Théorème de décomposabilitié (Meiri [77]) : Tout problème *STP* est décomposable par rapport aux contraintes décrites par son graphe minimal de distance.

Preuve : Il suffit de montrer qu'une instantiation quelconque d'un sous−ensemble S de k variables qui représentent des dates d'instants ($1 \le k < n$) et qui satisfait tous les plus courts chemins appliqués à S est extensible à n'importe quelle autre variable. On procède par induction sur $|S| = k$.

Pour $|S|=1$, S consiste à instancier une variable unique $X_i = x_i$. Pour n'importe quelle variable X_j, nous devons montrer qu'il est possible de trouver une valeur $X_j = v$ qui satisfait les contraintes du plus court chemin entre ces deux variables. La valeur v doit satisfaire :

$$-d_{ji} \le v - x_i \le d_{ij} \quad (1)$$

Puisque tous les cycles du graphe de distance sont non−négatifs, nous avons :

$$d_{ji} + d_{ij} \ge 0$$

Donc, il existe au moins une valeur v qui satisfait (1)

Pour $|S| = k$, supposons que le théorème est vrai pour $|S| = k-1$, et montrons qu'il reste vrai pour $|S| = k$: soit $S = \{X_1, ..., X_k\}$ et soit $\{X_i = x_i \mid 1 \le i \le k\}$ des instantiations qui satisfont les contraintes des plus courts chemins entre les variables de S. Soit $X_{k+1} \notin S$. Nous devons trouver une valeur $X_{k+1} = v$ qui satisfait les contraintes des plus courts chemins entre X_{k+1} et toutes les variables de S. La valeur v doit donc satisfaire :

$$v - x_i \le d_{i,\,k+1}$$

$$x_i - v \le d_{k+1,\,i}$$

Pour =1, ..., k ou encore

$$v \le \min\{ x_i + d_{i,\,k+1} \mid 1 \le i \le k \},$$

$$v \ge \max\{ x_i + d_{k+1,\,i} \mid 1 \le i \le k \},$$

Supposons que le minimum est atteint à i_0 et le maximum à j_0. En conséquence, v doit satisfaire :

$$x_{j0} - d_{k+1,\,j0} \le v \le x_{i0} + d_{i0,\,k+1} \quad (2)$$

Puisque x_{i0} et x_{j0} satisfont les contraintes du plus court chemin, nous avons :

$$x_{j0} - x_{i0} \le d_{i0,\,j0}$$

$$x_{j0} - d_{k+1,\ j0} \leq v \leq x_{i0} + d_{i0,\ k+1}$$

Ceci ,avec $d_{i0,\ j0} \leq d_{i0,\ k+1} + d_{\ k+1,\ j0}$ donne :

$$x_{j0} - d_{k+1,\ j0} \leq x_{i0} + d_{i0,\ k+1}$$

Donc, il existe bien une valeur v qui satisfait les conditions de (2)

L'importance de la propriété de décomposabilité pour le multimédia est double :

- Le formatage pour la présentation peut être facilité, puisque la construction de la solution peut être effectuée sans retour arrière (sans remise en question des instantiations précédentes).
- L'ordre d'instantiation des variables n'étant pas important, on peut définir plusieurs politiques de priorités lors du choix d'une solution. On peut, par exemple, commencer par les variables qui représentent des éléments multimédia ayant un réel contenu (audio, vidéo, etc.) en respectant le plus possible leurs valeurs optimales, les variables qui correspondent à de simples délais étant choisies en seconde priorité.

Ce résultat peut avoir des implications importantes sur l'organisation d'une application de présentation multimédia. En effet, un STP étant décomposable, l'opération de formatage (en particulier la politique adoptée pour réaliser le formatage) peut être effectuée aussi bien à l'édition que pendant l'exécution de la présentation (formatage dynamique). Cette possibilité est particulièrement intéressante pour des systèmes qui permettent de produire dynamiquement des parties de scénarios [25].

III.4.5 Synthèse des techniques multimédia existantes

À la différence des systèmes présentés dans le chapitre précédent où il s'agissait de décrire les caractéristiques fonctionnelles des outils d'édition, nous évaluons dans cette section les modèles temporels qui sont proposés pour les documents multimédia. Cette évaluation est basée sur un ensemble de critères introduits dans les sections précédentes comme la représentation de l'unité temporelle de base, le type de relations temporelles utilisées, leur sémantique (qualitative, quantitative, causale) et la structure des scénario considérés. À ces critères, nous ajoutons les suivants, qui sont à prendre en compte aussi bien dans la phase d'édition que dans la phase de présentation :

- L'existence de mécanismes d'analyse pour déterminer d'une part la cohérence d'un scénario et d'autre part les domaines de validité (valeurs pour lesquelles on peut effectivement trouver un scénario cohérent).
- L'existence de mécanismes de synthèse pour trouver une solution particulière (c'est l'opération de formatage temporel).

- Les performances, c'est–à–dire le temps de mise à jour d'un scénario et éventuellement de la vérification de sa cohérence et du formatage qui sont induits.
- L'incrémentalité des opérations de mise à jour, ce qui consiste à fournir des mécanismes d'analyse et de synthèse applicables à chaque opération d'édition sans remettre en cause la totalité du scénario.
- La distinction dans le modèle entre les intervalles déterministes et les intervalles indéterministes (indéterminisme vs. flexibilité).

Modèle temporel	Timeline	Höpner & AHM	OCPN	Firefly & Isis
Technique de représentation	Axe de temps absolu	Expressions de parcours	Réseau de Pétri	Réseau de points
Représentation de l'unité temporelle de base	instant	intervalle	intervalle	instant
Expression des relations	Implicite (datation)	Séquence & Par-min & Parmax	13 IA	3 PA
Sémantique des relations	quantitative	causale	qualitative	qualitative quantitative
Structure du scénario	Non	Arbre	Arbre	Graphe
Analyse : Cohérence Domaines valides	–(6) –	Non –	Non Oui	Oui Non
Synthèse (formatage)	–	–	Non	Simplex
Performances	–	–	–	Très faibles
Incrémentalité	Oui	Oui	Non	Non
Indéterminisme vs Flexibilité	Non	Non	Non	Non

Fig. 3.13 : Évaluation des principaux modèles de synchronisation multimédia

Les modèles proposés n'abordent, comme le montre le tableau de la Fig. 3.13, qu'une partie des problèmes posés. La complexité de conception d'un système d'édition multimédia complet apparaît clairement dans la diversité des aspects mis en avant. En particulier, les méthodes de construction d'un document, de vérification de la cohérence et de formatage temporel s'adaptent peu à un processus interactif d'édition.

(6) «–» désigne une information non précisée ou non considérée dans le modèle.

III.5 Conclusion

Dans ce chapitre, nous avons présenté les principaux aspects liés à la modélisation des documents multimédia. En partant d'une modélisation des éléments par des intervalles de validité et d'une étude des relations multimédia, nous avons montré que le problème de spécification d'un scénario multimédia peut se ramener à un système de contraintes de type *STP*. Les *STP* qui reposent sur l'algèbre d'instants ou, ce qui revient au même, sur l'algèbre d'intervalles restreinte, semblent être le meilleur compromis entre efficacité et pouvoir de représentation.

L'approche fondée sur les contraintes est attrayante sur plus d'un aspect. En effet, elle apporte des techniques de vérification de la cohérence et d'analyse d'un scénario adaptées pour le traitement de l'aspect qualitatif et quantitatif du temps. Le formatage (ou la recherche de solution) se trouve aussi simplifié grâce à la propriété de décomposabilité. Il reste cependant plusieurs questions en suspend qui ne sont pas abordées et que nous allons tenter de compléter dans la suite. Ces questions concernent principalement :

- La causalité et l'indéterminisme temporel, jusque là ignorés par les modèles existants.
- La construction incrémentale d'un scénario temporel dans le cadre d'un système d'édition.

Chapitre IV

Architecture et représentation des documents dans Madeus

IV.1 Introduction

Ce chapitre présente le système d'édition et de présentation Madeus qui met en œuvre, dans un environnement interactif, un modèle temporel de documents multimédia fondé sur les contraintes.

La démarche adoptée consiste à mettre en correspondance d'une part le processus d'édition avec la phase de formulation et d'analyse d'un problème de type *CSP* temporels (insertion d'intervalles, de relations temporelles et vérification de la cohérence), et d'autre part le processus de présentation avec la phase de résolution de ce problème (la synthèse). La Fig. 4.1 illustre cette démarche.

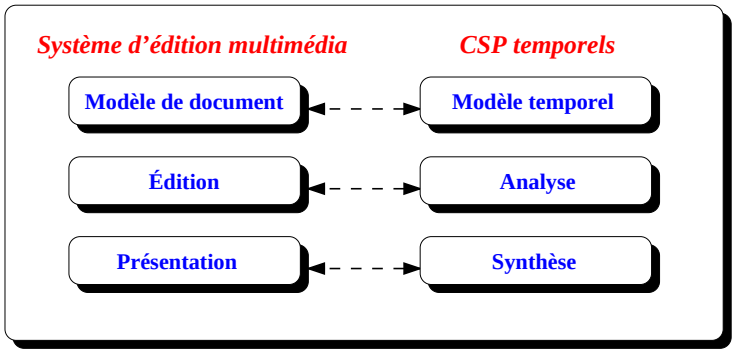


Fig. 4.1 : Système d'édition multimédia et CSP temporels

La validation de cette approche s'appuie, en grande partie, sur la réalisation d'un environnement opérationnel complet dont nous décrivons les objectifs et les principales caractéristiques dans la première section de ce chapitre.

La deuxième partie est consacrée à la description du format de document utilisé. Cette description, illustrée à travers un document exemple, couvre les organisations logique, temporelle et hypermédia des documents. L'interface graphique d'édition et l'organisation spatiale ne sont pas présentées de façon détaillée dans la mesure où, n'étant pas au cœur de

cette thèse, la solution actuelle n'est pas figée [17]. Cependant, un des choix de réalisation de Madeus est la construction d'une application ouverte et extensible pour permettre, dans une perspective à plus long terme, un travail plus complet sur la dimension spatiale [63].

IV.2 Le système Madeus

Après une discussion des principes qui ont présidé la conception de Madeus, nous présentons son architecture générale ainsi que la méthode de construction des documents que nous avons définie. Ensuite, une présentation générale de l'éditeur lui–même est effectuée, de façon à identifier les fonctions de manipulation de la structure temporelle de celles du système de présentation des documents. Ces deux tâches fondamentales du système Madeus font respectivement l'objet des chapitres V et VI.

IV.2.1 Principes de conception

L'approche adoptée dans cette thèse tente d'allier édition interactive et présentation multimédia avec un domaine de l'intelligence artificielle qui traite du problème des contraintes.

Du côté de l'édition, ce qui nous paraît intéressant dans une approche interactive similaire au *Wysiwyg*, c'est la possibilité de voir et de modifier le document sans changer d'environnement : le passage du mode d'édition au mode de présentation, et vice–versa, doit être aussi fluide que possible. L'intérêt d'une telle approche réside dans l'interactivité, qui rend plus facile l'accès au système et permet à l'utilisateur de vérifier en permanence ce qu'il fait. Ce principe, largement adopté pour les documents statiques, est cependant plus complexe à mettre en œuvre pour les documents multimédia. Cela est du à la nature temporelle très diverse des éléments multimédia et des dimensions multiples du document.

D'un autre côté, les systèmes de contraintes permettent à l'auteur d'exprimer, par l'insertion (potentiellement interactive) de relations, ce qu'il souhaite obtenir, sans se soucier de la complexité du scénario qu'il construit. Les systèmes de contraintes permettent ainsi, par des méthodes plus systématiques d'analyse et de synthèse, de vérifier la cohérence d'un scénario temporel et de rechercher une solution, même pour des situations complexes. De plus, l'approche déclarative, inhérente aux systèmes de contraintes, ouvre des perspectives plus larges au traitement électronique du document multimédia. En effet, à partir de la description des relations entre les différents éléments d'un document, cette approche permet de construire des structures de données (graphe de contraintes) qui se prêtent mieux aux traitements adaptatifs, voire génériques, contrairement aux descriptions impératives qui produisent des programmes moins adaptables. Des exemples typiques de

ces traitements sont l'adaptation du contenu d'un document en fonction de la langue du lecteur ou encore en fonction des ressources disponibles dans son environnement (taille d'écran, possibilités graphiques ou sonores). Du point de vue des contraintes, ces adaptations peuvent se ramener à une modification et une ré–évaluation du graphe de contraintes. Par exemple, le changement de la langue peut être ramené au changement de la durée des éléments audio dans le graphe de contraintes.

En partant de ces deux constats, le système Madeus tente de tirer profit des avantages des systèmes de contraintes pour l'édition multimédia en offrant à l'utilisateur, à travers la même interface graphique, des fonctions d'édition et de présentation. Du point de vue théorique, il s'agit aussi de compléter les systèmes de contraintes pour prendre en compte l'incrémentalité, l'indéterminisme et la causalité inhérents à l'édition et aux présentations multimédia.

IV.2.2 Architecture générale de l'application Madeus

Nous présentons dans cette section l'architecture générale de l'application Madeus. Cette architecture est organisée sous forme de quatre parties qui interagissent tout au long d'une session d'édition (voir Fig. 4.2) :

- **L'interface utilisateur** : elle regroupe les moyens graphiques offerts à l'utilisateur comme les boutons, les formulaires et les palettes pour réaliser les opérations d'édition et de présentation (voir section IV.2.4).
- **Le système d'édition** : il offre un ensemble de fonctions permettant de construire ou modifier de façon interactive un document multimédia, de définir ses relations temporelles, spatiales et hypermédia. Ce module permet aussi le chargement en mémoire d'un document ainsi que son stockage sur disque en fin de session d'édition.
- **Le gestionnaire temporel** : il prend en charge toute la gestion des opérations d'édition et de présentation qui concernent la dimension temporelle du document.
- **Le système de présentation** : il restitue le document à l'utilisateur en assurant l'ordonnancement de ses éléments, la gestion des ressources ainsi que la prise en compte de l'environnement d'exécution (interactions de l'utilisateur, indéterminisme des durées, etc.).

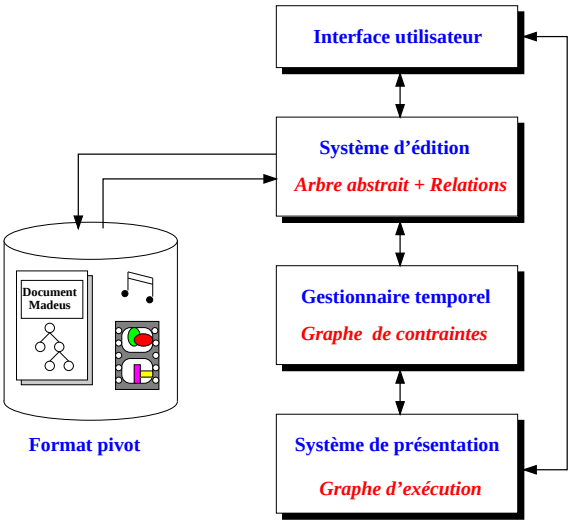


Fig. 4.2 : Architecture générale de l'éditeur Madeus

Lors du chargement, le document est pris en charge par le système d'édition qui en produit une forme interne. Cette représentation, appelée **arbre abstrait**, rend compte de l'organisation logique du document en termes d'éléments composites et d'éléments de base (cf. Chapitre II). Chaque nœud composite de cet arbre est décoré par la liste des relations temporelles ou spatiales définies entre ses fils. Le gestionnaire temporel intervient à chaque opération qui modifie la structure temporelle du document. Il maintient une représentation du document sous forme de **graphes de contraintes** (un graphe de contraintes est associé à chaque élément composite) exploitables dans leur forme finale par le système de présentation. Le système de présentation assure la fonction d'exécution du document sur une structure interne appelée **graphe d'exécution**, dérivée des graphes de contraintes.

L'interface utilisateur et le système d'édition sont brièvement décrits dans les deux sections suivantes, à travers la suite des opérations effectuées par l'auteur pour composer un document. La syntaxe utilisée et la structure de l'arbre abstrait qui en découle sont données dans la section IV.3, principalement à l'aide d'un exemple complet développé. Le gestionnaire temporel fait l'objet du Chapitre V tandis que le système de présentation est décrit dans le Chapitre VI.

IV.2.3 Vue d'ensemble du processus d'édition dans Madeus

L'utilisateur d'un système de production de documents multimédia peut s'intéresser à différents aspects d'un document : en tant qu'auteur il va s'intéresser au contenu et à l'organisation en composants du document, alors qu'en tant que graphiste son intérêt sera axé sur la mise en page et la disposition graphique. En tant que compositeur de scénario,

son attention se portera sur l'enchaînement des éléments dans le temps. À partir des facettes multiples d'un document introduites dans le chapitre II, nous pouvons considérer que le processus d'édition d'un document est une combinaison des quatre fonctions suivantes :

- **Saisie et structuration :** cette fonction consiste à insérer de nouveaux éléments multimédia dans le document et à organiser son contenu (tels que chapitres, scènes, titres, paragraphes, annotations, etc.).
- **Composition temporelle :** elle consiste à définir les relations temporelles entre les éléments d'un document multimédia.
- **Placement spatial :** cette opération consiste à définir l'apparence graphique du document à l'écran et le positionnement géométrique entre ses éléments en termes de relations spatiales [17].
- **Mise en place d'une structure de navigation :** cette opération consiste à superposer à la structure logique d'un document des parcours de navigation qui traversent son organisation hiérarchique (sauts intra et inter–documents) ainsi que de son scénario temporel (sauts temporels).

L'ordre d'accès à ces fonctions est totalement libre pour l'auteur, hormis la nécessité d'effectuer l'opération de création d'éléments avant de les composer avec d'autres. La construction progressive d'un document multimédia est facilitée grâce à deux aspects fondamentaux de Madeus :

- L'incrémentalité du gestionnaire temporel, qui permet à l'auteur d'être avisé à chaque opération de la validité des relations temporelles qu'il insère.
- L'intégration dans un même système des fonctions d'édition et de présentation, qui permet à l'utilisateur de basculer aisément du rôle d'auteur au rôle de lecteur, et vice–versa.

Les différentes fonctions d'édition et de présentation sont rendues accessibles à travers l'interface utilisateur décrite ci–dessous.

IV.2.4 Interface utilisateur de Madeus

L'interface d'édition et de présentation de Madeus (voir Fig. 4.3) est construite au–dessus de la boîte à outils graphique de OSF/Motif [79]. Elle est composée d'une fenêtre principale organisée sous forme d'un ensemble de *widgets* permettant à l'utilisateur de réaliser les fonctions citées dans la section précédente. À travers cette interface, l'utilisateur peut charger un document, le modifier, interagir avec ses éléments ou contrôler sa progression temporelle. Il peut aussi agir sur les paramètres de sortie du

document comme la taille de la fenêtre, sa partie visible (à travers les barres de défilement) ou encore contrôler le volume audio global de la présentation.

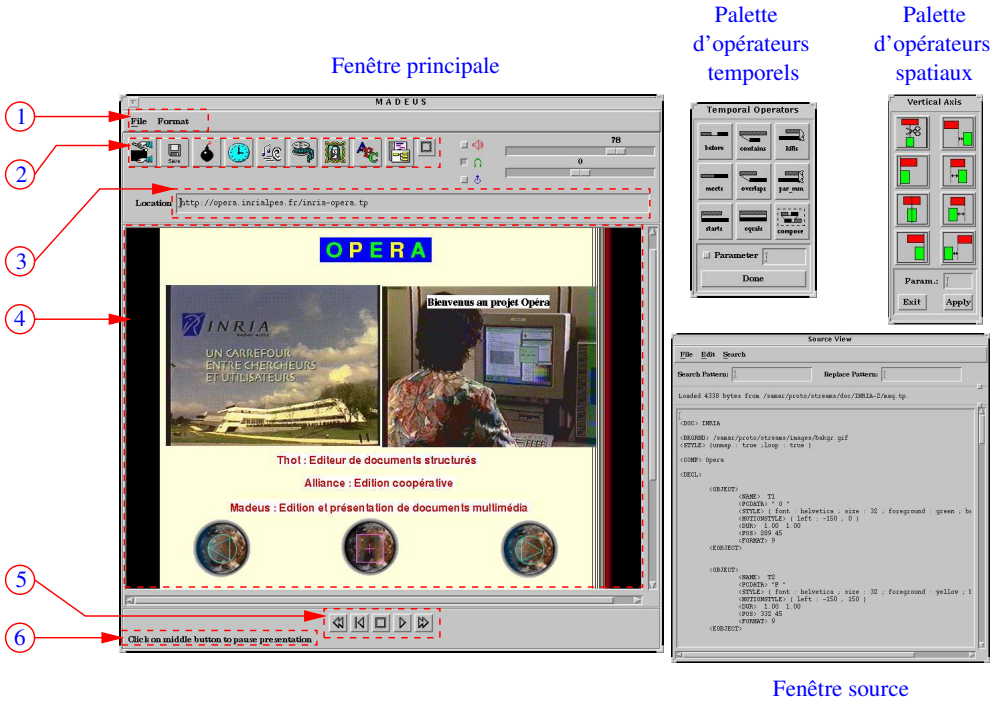


Fig. 4.3 : Interface utilisateur de l'éditeur Madeus

Un document Madeus peut être visualisé selon deux formats dans deux fenêtres différentes : le document composé (ou en cours de composition) est affiché et exécuté dans la fenêtre « principale » tandis que sa forme pivot (textuelle) est affichée dans la fenêtre « source ». À la fenêtre principale sont associées deux palettes d'édition : la palette d'opérateurs temporels et la palette d'opérateurs spatiaux.

La fenêtre principale est composée de six parties (numérotées sur la Fig. 4.3) qui offrent des fonctions d'édition et de présentation :

1. Une cascade de menus permettant de définir les attributs des éléments du document, par exemple leur durée ou la fonte des caractères (cf. IV.3.5).
2. Une barre de boutons permettant de charger un document, de lancer sa présentation ou sa sauvegarde, ainsi que d'insérer de nouveaux éléments.
3. Une zone de saisie du nom ou de l'adresse d'un document permettant son chargement à partir d'un fichier local ou d'un serveur distant (*HTTP*). La syntaxe des noms de fichiers utilisée est identique à celle employée dans le web : les URL (*Uniform Resource Locator*) [8].

4. Une surface d'édition/présentation et d'interaction qui est la zone principale de restitution graphique du document.
5. Un panneau de contrôle de la progression temporelle de la présentation d'un document. Ce panneau contient un ensemble de boutons permettant l'accès à des opérations génériques de navigation telles que l'arrêt, le démarrage, l'avance rapide et le retour arrière du document en cours de présentation.
6. Une zone de messages utilisée par Madeus pour fournir des indications à l'utilisateur (compte−rendu d'opérations).

L'utilisateur peut agir directement sur la zone principale (4) tant pour effectuer des opérations d'édition que pour des actions de présentation. Par exemple, il peut sélectionner des éléments pour les composer spatialement, temporellement ou les détruire. Lors de l'exécution, l'utilisateur perçoit à travers cette zone l'évolution de la présentation et peut interagir directement au moyen de la souris. Cette interaction permet l'activation d'éléments comme les boutons, les références ou tout élément multimédia susceptible de réagir à ce type de manipulation (en particulier les programmes *applets*).

Les documents édités au travers de cette interface de manipulation sont stockés dans un format textuel appelé « **format pivot** » (cf. Fig. 4.2) que l'utilisateur peut visualiser en ouvrant la fenêtre « source ». C'est le format d'échange des documents Madeus dont la syntaxe est décrite dans la section suivante.

IV.3 Représentation des documents dans Madeus

Le système Madeus manipule des documents décrits selon un format déclaratif sous forme d'un marquage descriptif. Cette description couvre l'organisation logique, temporelle, spatiale et hypermédia des documents.

Afin de faciliter la compréhension du format utilisé, nous allons d'abord donner un exemple d'un document multimédia. Cet exemple met en valeur les possibilités principales du système Madeus. Puis, nous décrivons le format pivot de Madeus, principalement à travers cet exemple.

IV.3.1 Exemple de présentation multimédia

Supposons que l'on souhaite construire un document multimédia *Inria−Opéra* qui présente les activités de recherche du projet Opéra (Fig. 4.4). Le document comporte un élément texte *Titre* de la présentation, une *Musique* de fond, et deux sous−scènes *Intro−Inria* et *Activité−Opéra*. *Intro−Inria* contient un reportage composé d'une vidéo *(Vidéo1)* qui présente le projet au sein de l'unité de recherche Inria Rhône−Alpes. Un

bande audio *(Audio1)* commente ce reportage et un bouton *(Bouton2)* permet d'interrompre la sous–scène *Intro−Inria*. *Activité−Opéra* est une sous–scène composée de parties : *Protos−de−Recherche* formée de trois textes (*Thot*, *Alliance* et *Madeus*) et *Présentation−Opéra* qui contient une audio (*Audio2*) et une vidéo (*Vidéo2*). Les trois textes (*Thot*, *Alliance* et *Madeus*) contiennent des liens externes vers d'autres documents. Deux boutons *Bouton1* et *Bouton3* permettent de naviguer dans le document *Inria−Opéra* : *Bouton1* permet de revenir au début du document et *Bouton3* permet d'accéder directement à la partie *Présentation−Opéra*.

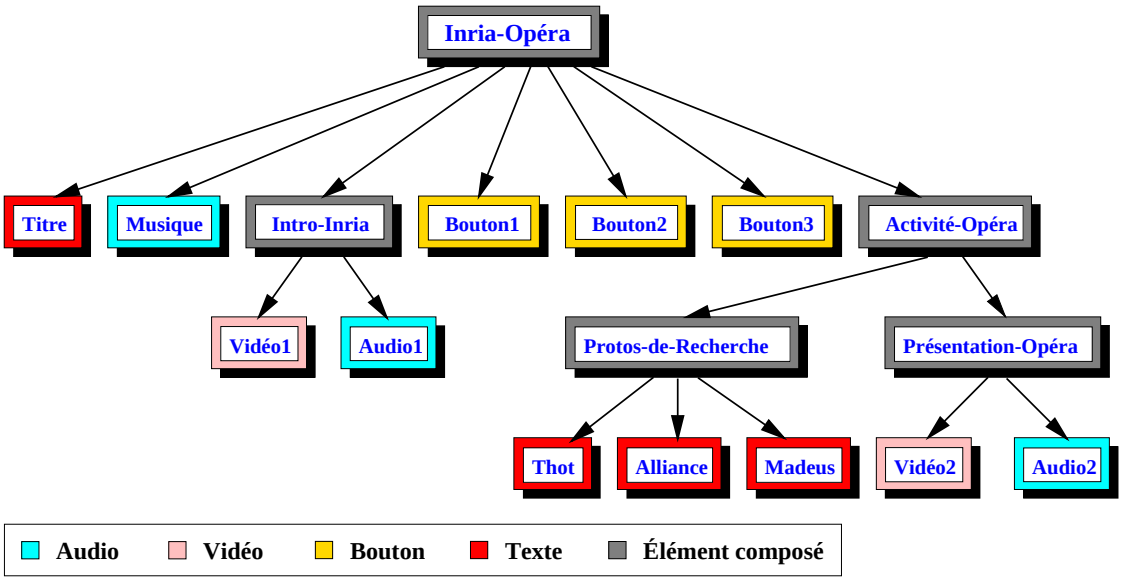


Fig. 4.4 : Structure logique du document multimédia Inria−Opéra

Le scénario temporel qu'on veut construire (cf. Fig. 4.5) consiste à présenter le *Titre* en même temps que la *Musique*. Entre 5 et 7 secondes après le début de la *Musique*, l'élément *Intro−Inria* est démarré. *Intro−Inria* consiste à présenter en même temps la vidéo *Vidéo1* et l'audio *Audio1* et toute la sous–scène peut être interrompue à tout moment par le bouton *Bouton2*. La terminaison de *Intro−Inria* a aussi pour effet de rendre ce bouton inactif. Après la terminaison de *Intro−Inria* (de façon naturelle ou par l'action du Bouton2), la partie *Activité−Opéra* est lancée. Dans cette sous–scène, les trois textes *Thot*, *Alliance* et *Madeus* de *Protos−de−Recherche* sont présentés progressivement avec un délai de 2 secondes entre deux textes successifs. *Présentation−Opéra* démarre 6 secondes après la fin de *Protos−de−Recherche*. Pendant toute la présentation du document *Inria−Opéra* les deux boutons *Bouton1* et *Bouton3* sont activables.

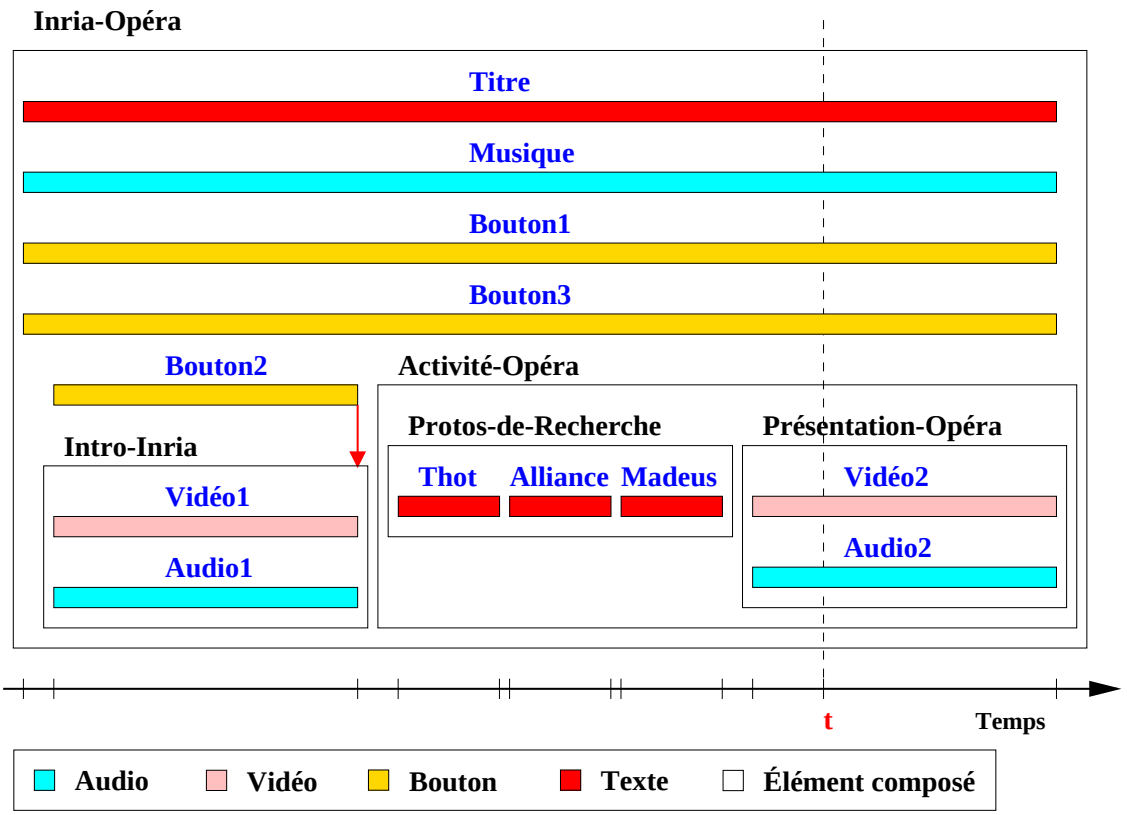


Fig. 4.5 : Scénario temporel du document Inria−Opéra

La disposition graphique du document Inria–Opéra est présentée dans la Fig. 4.6. La configuration de l'écran sur la Fig. 4.6 correspond à l'instant t (Fig. 4.5) de la présentation. Les éléments Vidéo1, Thot, Alliance, Madeus sont encore présents à l'écran, mais ils sont figés dans l'état qu'ils avaient à la fin de leur exécution. On entend l'élément Audio2 et la Musique.

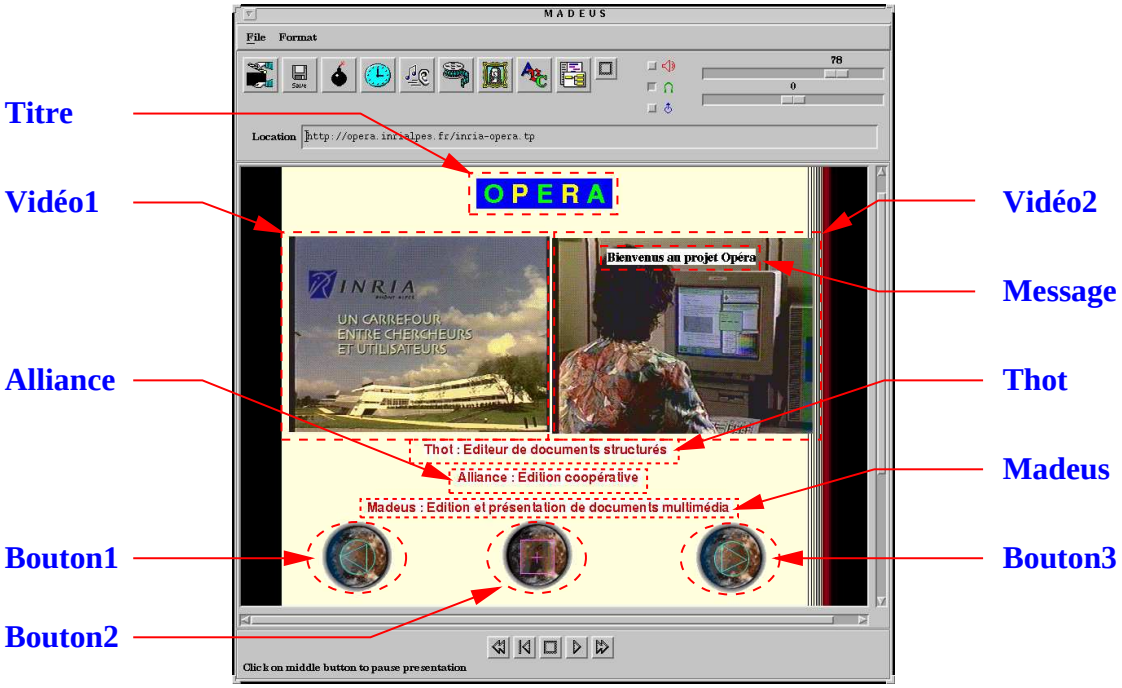


Fig. 4.6 : Présentation graphique du document Inria−Opéra à l'instant t

IV.3.2 Langage de marquage

Le marquage descriptif des éléments est le principe de base du format pivot de Madeus. Des marques (ou balises) sont associées à chacun des éléments. Les marques sont utilisées pour typer, délimiter et identifier les éléments d'un document et pour leur associer des attributs. Le marquage est une technique bien connue dans le domaine des documents et a été normalisé dans SGML [40].

Dans SGML, la structure logique générique est décrite dans une DTD (Document Type Definition) qui spécifie la grammaire d'une classe de documents (HTML en est un exemple) sous la forme d'un ensemble de types d'éléments et d'attributs, et des règles qui permettent de les assembler.

La notion de *constructeurs* (*connecteurs* et *indicateurs d'occurrences* dans SGML) exprime le mode d'assemblage des éléments de la structure. Par exemple, le constructeur « liste » qui permet d'exprimer le fait qu'un élément est composé d'une suite d'éléments de même type, est représenté en SGML par l'indicateur d'occurrence « + » qui indique qu'un type est obligatoire et répétable. Ainsi, le *Corps* d'une *Présentation* peut être décrit comme une liste de *Scènes* (Fig. 4.7).

```
<!ELEMENT Présentation (Titre, Corps) >
<!ELEMENT Titre (#PCDATA)             >
<!ELEMENT Corps (Scène+)              >
```

Fig. 4.7 : La DTD d'une présentation

Un *attribut* est une information attachée à un type pour en préciser la fonction dans le document. Il ajoute une information d'ordre sémantique et peut être utilisé par diverses applications (par exemple l'attribut langue est utilisé par un correcteur orthographique dans Thot [88]). Il existe plusieurs types d'attributs : énuméré, entier, texte et référence, etc.

Dans la figure Fig. 4.8 nous présentons une instance de document selon le format SGML et respectant la DTD décrite en Fig. 4.7. Les marques *<Scène>* et *</Scène>* sont les balises qui délimitent le début et la fin respectivement du contenu des différentes scènes d'une présentation.

```
<Présentation>
<Titre> Opéra </Titre>
<Corps>
<Scène>
... Cette scène traite des problèmes ...
</Scène>
<Scène>
...  ...
</Scène>
</Corps>
</Présentation>
```

Fig. 4.8 : Un document Présentation en SGML

Il est à noter que dans Madeus, la généricité n'est pas prise en compte, contrairement à des systèmes d'édition comme Thot [88]. Le marquage s'appuie sur une DTD dont la spécificité est de prendre en compte les aspects relatifs au temps : nouveaux éléments de base, durées, relations temporelles et effets dynamiques sur le style.

IV.3.3 Structure d'un document Madeus

On a vu dans le chapitre II qu'un document multimédia pouvait être considéré selon les quatre dimensions logique, spatiale, temporelle et hypermédia. Le modèle de documents que nous avons défini dans Madeus, dont la DTD simplifiée est donnée dans Fig. 4.9, permet de décrire ces quatre dimensions :

- L'organisation logique est une suite de structures hiérarchiques, la première étant l'arbre principal du document (Princ) et les suivantes étant des arbres associés (Assoc). Chaque arbre est formé d'éléments composés (Comp) et d'éléments de base (Texte, Image, Audio, Vidéo et Objet).

- Les autres dimensions sont définies par le biais d'attributs attachés aux éléments composés ou aux éléments de base. Les attributs RelTemp pour les relations temporelles et RelSpat pour les relations spatiales ne peuvent être attachés qu'aux éléments composés. Les attributs définissant les liens peuvent être soit des Références soit des Inclusions.

```
<!ELEMENT Doc – – (Princ, (Assoc)*)          >
<!ELEMENT (Princ | Assoc) – – (%Base | Comp) >
<!ELEMENT Comp – – (%Base | Comp)+           >
<!ENTITY  % Base "Texte | Image | Audio | Vidéo | Objet" >
<!ENTITY  % AttCommuns
          "Nom       NAME  #IMPLIED
           Durée     CDATA #IMPLIED"         >
<!ATTLIST Comp
          %AttCommuns;
          RelSpat    CDATA #IMPLIED
          RelTemp    CDATA #IMPLIED          >
<!ATTLIST %Base;
          Contenu    CDATA #IMPLIED
          %AttCommuns;
          Référence  CDATA #IMPLIED
          Inclusion  CDATA #IMPLIED          >
```

Fig. 4.9 : DTD du modèle de documents de Madeus

Les arbres associés sont des éléments flottants : ils n'ont pas de place fixe dans le document. Les éléments associés appartiennent à des structures secondaires car il peuvent ne pas apparaître pendant la présentation.

Dans les sections suivantes, nous détaillons les composants principaux de cette syntaxe en les illustrant avec des parties de l'exemple *Inria−Opéra* dont la structure générale est donnée en Fig. 4.10 :

- Éléments de base
- Attributs de structuration
- Attributs de présentation
- Attributs temporels
- Attributs hypermédia

```
<Doc Nom=Inria-Opéra>
  <Princ>
    <Texte Nom=Titre
           Contenu="Opéra"
           Durée="...">
    <Comp Nom=Intro-Inria
          Rel-Temp="..."
          Rel-Spat="...">
          <Vidéo Nom=Vidéo1
                 Durée="...">
          <Audio Nom=Audio1
                 Durée="...">
          ...
    </Comp>
    ...
  </Princ>
  <Assoc>
     ...
  </Assoc>
</Doc>
```

Fig. 4.10 : Extrait du format pivot du document Inria−Opéra

IV.3.4 Les éléments de base

Un élément de base est atomique. Il correspond au vocabulaire terminal de la grammaire. Il existe cinq types de base actuellement supportés dans l'application Madeus :

- Le type **Text** sert à construire des chaînes de caractères dans un certain alphabet.
- Le type **Image** désigne les images fixes définies par des matrices de points. Les formats supportés sont : *gif*, *jpeg*, *png* et *pixmap*. Les formats vectoriels comme *cgm* sont supportés à travers le type **Objet**.
- Le type **Vidéo** désigne les éléments au format *Mpeg* et *Mjpeg*.
- Le type **Audio** désigne les éléments au format *Mpeg_audio* et *Au*.
- Le type **Objet** regroupe des éléments qui correspondent à des programmes permettant d'intégrer dans les documents Madeus d'autres types de média. Ce type a le même rôle que l'élément OBJECT de HTML.

Mis à part le type texte qui peut être directement représenté dans le document, le contenu des autres types d'éléments multimédia est stocké dans des fichiers externes (cf. l'attribut **Contenu** décrit ci−dessous).

IV.3.5 Les attributs

L'utilisation des attributs permet dans Madeus non seulement de compléter la description logique des documents, mais aussi de définir leur organisation spatiale, temporelle et hypermédia. Avant de décrire ces différentes catégories d'attributs, nous en identifions les caractéristiques.

IV.3.5.1 Caractéristiques des attributs

Trois caractéristiques peuvent être identifiées pour les attributs définis dans Madeus. Le caractère facultatif ou obligatoire ainsi que la portée des attributs sont des notions bien connues dans les modèles de documents. Ce qui l'est moins, c'est la caractéristique temporelle que nous avons introduite sur les valeurs d'attributs. Pour résumer, un attribut peut être :

- **Facultatif ou obligatoire**.
- **Global ou local** : les attributs globaux peuvent être portés par tous les éléments du document, par exemple les attributs **Nom** et **Durée**. Les attributs locaux sont définis pour certains types d'éléments, un cas particulier étant les attributs média–dépendants (la **Fonte** pour les caractères ou le nombre d'images par seconde pour une vidéo).
- **Statique ou dynamique :** les attributs statiques conservent la même valeur durant toute leur durée de présentation tandis que la valeur des attributs dynamiques varie au cours du temps comme le déplacement d'une fenêtre à l'écran.

IV.3.5.2 Les attributs de structuration

Chaque élément de base ou construit possède les attributs suivants :

- L'attribut obligatoire **Nom** qui est un identifiant permettant de désigner l'élément dans tout le document.
- L'attribut **Activable** permet de spécifier si l'élément peut être utilisé comme un bouton d'activation (Fig. 4.12).

En outre, les éléments de base portent les attributs obligatoires suivants :

- L'attribut **MimeType** qui précise le format de l'élément [12], comme *gif* ou *jpeg* pour les images ou *au* pour les audios (Fig. 4.11).
- L'attribut **Contenu** donne l'information de contenu (chaîne textuelle) ou de localisation du contenu (URL).

IV.3.5.3 Les attributs de présentation

Certains attributs de présentation peuvent être portés par les éléments composés pour permettre des mécanismes de propagation par héritage (héritage de la fonte par exemple).

La portée de ces attributs dans le document dépend non seulement de son organisation logique, mais aussi de son scénario temporel. Par exemple, si on applique un attribut de changement de volume sur les 3 dernières secondes à un élément composé, les effets dynamiques de cet attribut porteront sur les éléments audio qui seront en cours de présentation sur cette période de temps.

Les éléments de base ne peuvent porter que les attributs servant à la présentation de leur média ; par exemple, la fonte pour les éléments textuels ou le volume sonore pour les éléments audio.

Les attributs de présentation peuvent être statiques ou dynamiques. Les attributs dynamiques permettent de décrire des effets de style, par exemple le déplacement d'un texte à l'écran ou l'apparition progressive à l'écran d'une image. L'idée de base consiste à définir, à partir de la durée totale d'un élément, des tranches de temps (ou projections) pendant lesquelles la valeur de l'attribut varie de façon linéaire, discrète ou continue. Afin de faciliter la modification du scénario, la position de ces tranches est définie par rapport à l'instant de **Début** ou de **Fin** de l'élément.

Dans l'exemple de la Fig. 4.11, nous présentons une application de ce principe à deux éléments dynamiques du document *Inria−Opéra*, le premier est l'élément textuel *Thot* et le deuxième, l'élément audio *Musique*. Pendant les deux premières secondes de sa présentation, l'élément *Thot* se déplace à l'écran de la gauche vers la droite jusqu'à sa position finale. Pour la séquence audio, le volume de départ augmente progressivement entre le début et 60 secondes de 0 à 50 unités, puis garde une valeur constante pendant 75 secondes avant de diminuer progressivement jusqu'à la fin.

```
<Texte NOM=Thot
     Durée="[0, −, 30]s"
     Fonte=Times
     Xpos="[0, 200]pts, [Début, 2]s"
     Contenu="Thot : Éditeur de documents ..." >
<Audio NOM=Musique
     MimeType="audio/au"
     Durée="[180, −, −]s"
     Volume="[0, 50]unités, [Début, 60]s;
             50 unités, [60, 135]s;
             [50, 0]unités, [135, Fin]s"
     Contenu="http://opera.inrialpes.fr/Bach.au">
```

Fig. 4.11 : Exemple d'attributs dynamiques dans Madeus

Le placement spatial des éléments fils d'un élément composé peut être spécifié par des relations spatiales définies avec l'attribut **RelSpat**. L'idée est de permettre à l'auteur de composer son document de la même façon, que ce soit dans la dimension spatiale ou dans la dimension temporelle, par des relations :à gauche de/ avant, à droite de/ après, etc. La

définition des relations spatiales ainsi que leur gestion dans le système Madeus fait l'objet d'un travail réalisé en dehors du cadre de cette thèse [17].

IV.3.5.4 Les attributs temporels

Pour chaque élément de base ou composé, il existe trois attributs à partir desquels le scénario temporel est construit :

- L'attribut **Durée** précise les bornes de durée d'un élément : c'est un triplet de valeurs (durée minimale, nominale et maximale) (cf. Chapitre III).
- L'attribut **Contrôlable** qui spécifie si la durée de l'élément peut être fixée par le système de présentation (à l'intérieur de la plage **Durée**), ou si **Durée** indique la plage des valeurs que l'élément prendra à l'exécution sans contrôle de la part du système (cas des éléments indéterministes comme le bouton d'interaction Fig. 4.12).
- L'attribut **Itération**, optionnel, est un entier qui représente le nombre de fois qu'un élément est présenté à l'utilisateur dans un scénario.

```
<Image NOM=Bouton2
     Activable=Oui
     Contrôlable=Non
     Durée="[0, -, 60]s"
     Contenu="http://opera.inrialpes.fr/im.gif">
```

Fig. 4.12 : Exemple d'élément non contrôlable

Dans le format de Madeus, les relations temporelles sont des relations d'intervalles. Elles sont décrites au niveau de chaque élément composé par l'attribut **RelTemp**. Il existe deux façons de spécifier ces relations :

- Globalement sur l'élément composé : en spécifiant une relation n–aire entre les éléments composants, comme par exemple **List−Before** pour l'élément *Protos−de−Recherche* (Fig. 4.13).
- De façon spécifique, entre des couples d'éléments fils, comme pour l'élément *Inria−Opéra* (Fig. 4.13).

```
<Doc NOM=Inria-Opéra
     ...
     RelTemp=
      "Titre Equals Musique;
      Intro-Inria During [5, -, 7]s Titre;
      Intro-Inria Before [4, -, 6]s Activité-Opéra;
      ...">
<Comp NOM=Protos-de-Recherche
      ...
      RelTemp="List_Before [2, -, 2]s" >
```

Fig. 4.13 : Les relations temporelles

La durée d'un élément composé est donc fonction non seulement de son attribut **Durée** et de sa composition temporelle avec ses frères, mais aussi de la durée résultat de la composition temporelle de ses descendants.

Il est important de noter que la structure logique (arborescente) des documents n'est pas le résultat de la composition temporelle entre les éléments contrairement à OCPN [71] ou Video Algebra [116]. Le langage offre cependant un moyen simple de spécifier au niveau d'un élément composé la composition temporelle entre des fils lorsque les deux dimensions coïncident (par des relations **List−Before** ou **List−Finishes**).

Afin de faciliter la mise à jour et la réutilisation de parties de document dans Madeus, des éléments ne peuvent être liés par une relation temporelle que s'ils sont frères selon la structure logique. Ce choix qui semble à première vue quelque peu restrictif est en réalité utile pour les opérations d'édition. En particulier, il permet d'isoler des portions de document que l'auteur peut couper, coller ou modifier indépendamment des autres éléments du document. Cette manière de disposer les relations permet de définir de façon précise, pour chaque élément composé de la structure logique, un instant de début, un instant de fin et une durée au même titre que pour les éléments de base du document. Ainsi, les éléments composés peuvent être manipulés au sein de l'éditeur de la même façon que les éléments de base du document.

IV.3.5.5 Les liens hypermédia

Les attributs **Référence** et **Inclusion** définissent des liens de navigation et des liens d'inclusion entre un élément de base et un autre élément (de base ou composé) du même document ou d'un document externe.

Les références constituent le principal support de la navigation intra et inter−documents. Dans le déroulement d'un scénario temporel, leur activation permet d'effectuer de véritables "sauts temporels" vers les instants de début ou de fin d'un élément ou d'un document. Par exemple Fig. 4.14, le Bouton3 permet de sauter au début de

la présentation de l'élément *Présentation−Opéra* et l'élément *Thot* permet d'accéder à un autre document.

```
<Image NOM=Bouton3
       Référence="Présentation-Opéra"
       Durée="[0, -, 75]s"
       Activable=Oui
       Contenu="http://opera.inrialpes.fr/Im3.gif">

<Texte NOM=Thot
       Référence="http://opera.inrialpes.fr/thot.tp"
       Durée="[0, -, 30]s"
       Activable=Oui
       Contenu="Thot : Éditeur de document ...">
```

Fig. 4.14 : Références internes et externes

Les inclusions sont des liens qui permettent d'ajouter, pendant la présentation d'un scénario, une portion de document définie dans un arbre associé (IV.3.3). Les inclusions peuvent avoir plusieurs utilisations : elles permettent à l'utilisateur d'insérer des annotations dans le document, comme l'élément *Message* associé à *Vidéo2* (Fig. 4.15) ; c'est aussi un moyen de définir un scénario avec des choix multiples, par exemple, un choix de titres de films qui incluront les séquences du film sélectionné par l'utilisateur.

Lors de l'activation d'un lien d'inclusion, deux modes de présentation sont possibles en fonction de la valeur donnée à l'attribut **Arrêt** (cf. Fig. 4.15): soit le scénario de l'inclusion est exécuté en parallèle avec scénario courant (comme dans notre exemple), soit le scénario courant est arrêté, l'inclusion est jouée, puis le scénario initial reprend depuis son point d'interruption.

```
<Vidéo NOM=Vidéo2
     MimeType="video/mpeg"
     Durée="[56, -, -]s"
     Activable=Oui
     Arrêt=Non
     Inclusion="Message"
     Contenu="http://opera.inrialpes.fr/opera.mpg">

<Assoc>
<Texte Nom=Message
       Contenu="Bienvenue ... ">
</Assoc>
```

Fig. 4.15 : Élément associé Message et son inclusion

Contrairement aux autres éléments du document, les références et les inclusions introduisent un aspect imprédictible dans le déroulement du scénario (asynchronisme temporel). En effet, leur activation dépend uniquement du comportement du lecteur du document au moment de la présentation.

IV.4 Conclusion

Dans ce chapitre, nous avons présenté le système d'édition et de présentation de documents multimédia Madeus. En particulier, nous avons présenté le format de document permettant l'expression des dimensions logique, spatiale, temporelle et hypermédia. Le mode d'expression des scénarios permet de définir de façon souple la dimension temporelle des documents multimédia, sans contraindre leur structure logique.

L'originalité principale de ce format de document réside dans le mode de définition déclaratif de la dimension temporelle (attributs de présentation avec effet de style, attribut d'activation, de répétition et de relations temporelles) qui supprime tout recours à la programmation.

Deux propriétés temporelles sont définies pour identifier les éléments indéterministes indispensables dans la construction d'un scénario hypermédia :

- L'attribut **Contrôlable** permet de séparer les intervalles indéterministes des intervalles flexibles.
- Les hyperliens ainsi que les boutons sont représentés par des intervalles qui correspondent à la période de temps où ils sont **Activables**. Ces intervalles s'insèrent de façon homogène dans le scénario et fournissent un support pour la navigation dans la structure temporelle du document (saut temporel vers un instant choisi du document).

C'est à partir de ce format (et de sa forme internalisée dans l'éditeur) que toutes les fonctions d'édition et de présentation liées au temps sont mises en œuvre au sein du gestionnaire temporel qui fait l'objet du chapitre suivant.

Chapitre V

Représentation et gestion des contraintes temporelles dans Madeus

V.1 Introduction

Ce chapitre rend compte du module de Madeus qui permet le traitement temporel des documents multimédia. Ce traitement consiste à prendre en charge les différentes opérations d'édition permettant de créer et de mettre à jour un document. Pour chaque opération d'édition, le gestionnaire temporel permet d'analyser l'état du scénario temporel en cours de composition. Cette analyse porte sur la vérification de la cohérence temporelle d'un point de vue qualitatif, quantitatif, causal et indéterministe. Une fois cette opération effectuée, l'auteur du document peut à tout moment lancer sa présentation. Cette opération nécessite le formatage du document, ce qui permet de déterminer une valeur de durée pour chaque élément du document.

Le travail présenté ici se veut avant tout une contribution aux systèmes d'édition de documents multimédia. Nous avons souhaité exposer d'abord dans le Chapitre III le raisonnement temporel dans le cadre plus général des systèmes de contraintes. En ce qui concerne l'application Madeus, nous avons accordé une attention particulière à deux aspects, l'incrémentalité et une certaine approche du temps. Comme on se place dans un environnement d'édition interactif, les différents traitements des contraintes temporelles doivent être faits par une manipulation **incrémentale**. D'autre part, il est nécessaire de prendre en compte l'aspect causal et indéterministe du temps inhérents à une présentation multimédia. Les propositions faites dans ce chapitre constituent une première solution dont le but est de permettre la construction d'une application d'édition et de présentation opérationnelle, Madeus. Elles ne prétendent pas résoudre toutes les difficultés théoriques posées, qui dépassent largement le cadre de cette thèse.

Dans la première section, nous présentons une vue d'ensemble du module de gestion de contraintes temporelles, appelé gestionnaire temporel. La section V.2 est consacrée à

la description des éléments de base, des relations temporelles supportées et à leur représentation interne sous forme d'un graphe d'instants qui constitue aussi notre réseau de contraintes, nous parlerons de **graphe de contraintes**. Dans la dernière section, les algorithmes de gestion des relations sont développés. Ces algorithmes forment le cœur de la phase d'édition de Madeus. Leur rôle est de produire une forme exécutable du document. À partir de cette forme, le module de présentation peut alors ordonnancer la présentation. Ceci fait l'objet du chapitre VI.

V.1.1 Objectifs

Dans Madeus, toute la conception du système d'édition repose sur la notion de **temps élastique**. L'idée fondamentale consiste à offrir à l'auteur la possibilité de définir et de manipuler le document de façon à la fois explicite et flexible. Pour atteindre cet objectif, il est nécessaire que le système d'édition puisse adapter automatiquement le scénario pour traduire tout au long d'une session les mises à jour de l'auteur et lui signaler, le cas échéant, la présence d'incohérences. Pour faciliter la tâche de l'auteur, il faut lui signaler les incohérences au moment même où il effectue l'opération qui en est la cause. Il faut donc faire des vérifications incrémentales.

Au delà de la difficulté théorique inhérente aux problèmes de synchronisation temporelle, une mise en œuvre efficace d'une application d'édition multimédia doit offrir une réponse adaptée aux problèmes qui viennent du processus d'édition/présentation :

- La construction d'un document multimédia est pour l'auteur un processus cyclique d'édition et de présentation. Il faut donc permettre à l'auteur de visualiser immédiatement le document en cours d'édition, même si celui–ci n'est que partiellement construit. La solution passe par la définition d'une structure de données interne du document qui soit à la fois adaptée à l'édition (facilitant les opérations de mise à jour comme l'insertion et la suppression d'éléments et de relations entre éléments), mais aussi à la présentation.
- Madeus est conçu pour servir aussi d'outil de présentation. Il nous faut donc viser de bonnes performances dans le processus de construction du scénario. En particulier, tous les mécanismes utilisés doivent pouvoir traiter de gros documents. L'adaptation de tous les traitements à une manipulation incrémentale et efficace du document revient à limiter la portée du traitement de chaque opération d'édition aux seules portions du document susceptibles d'être affectées. Il faut exploiter le plus possible la localité.

Le traitement de la synchronisation est donc intimement lié à la représentation interne d'un document. En effet, les opérations d'édition font intervenir à la fois la modification de

paramètres numériques représentant les durées des éléments d'un document (l'analyse et la synthèse) mais aussi des opérations de mise à jour de ces structures de données, c'est–à–dire des transformations de graphes.

V.1.2 Vue d'ensemble du gestionnaire temporel de Madeus

Le **gestionnaire temporel** de Madeus occupe une place particulière qui est au cœur de l'architecture générale de l'application. Ce module interagit avec le système d'édition de deux façons : tout au long d'une session interactive d'édition et pendant le chargement initial d'un document. Dans les deux cas, cette interaction se fait à travers des mises à jour de l'arbre abstrait par l'insertion ou la suppression d'éléments multimédia, de relations temporelles ou par la spécification des valeurs de leurs attributs temporels. C'est à travers ces mises à jour que tous les traitements réalisés par le gestionnaire temporel sont déclenchés.

Avant d'aborder en détail les traitements réalisés par le gestionnaire temporel, nous présentons dans cette section une vue d'ensemble de son architecture logicielle.

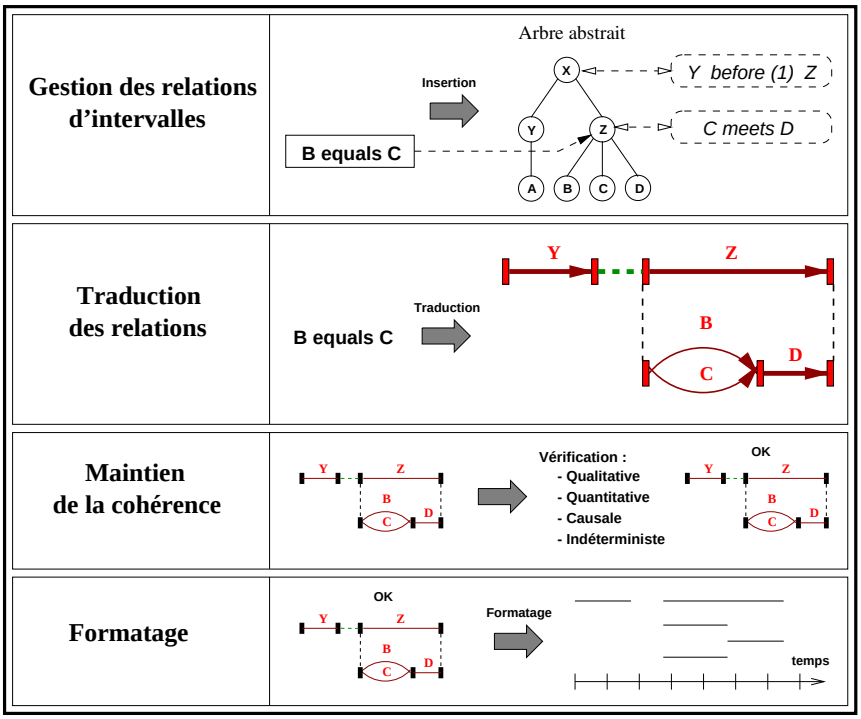


Fig. 5.1 : Organisation du système de gestion de la synchronisation de Madeus

Le gestionnaire temporel de Madeus est organisé sous forme de quatre modules dont chacun assure une fonction particulière du système d'édition et de présentation :

1. Le **module de gestion des relations d'intervalles** : Ce module met en correspondance, à travers l'arbre abstrait, les opérations d'édition de l'auteur et le traitement effectué par le gestionnaire temporel.
2. Le **module de traduction des relations** : il permet de maintenir en correspondance les relations temporelles à base d'intervalles et les relations à base d'instants.
3. Le **module de maintien de la cohérence** : il assure la vérification de la cohérence de chaque contrainte temporelle résultant de l'insertion et de la suppression d'une nouvelle relation d'instant.
4. Le **module de formatage** : il attribue à chaque élément une valeur de durée pour la présentation. Une partie de cette opération est effectuée dynamiquement pendant la présentation.

La Fig. 5.1 montre le cheminement des traitements qui, partant de l'organisation logique hiérarchique d'un document (l'arbre abstrait) et d'une suite de relations temporelles, aboutissent au niveau du formateur à une forme du document exploitable pour la présentation.

V.2 Représentation des contraintes dans Madeus

Au sein du gestionnaire temporel, la structure temporelle d'un document est maintenue à deux niveaux :

- un niveau au sein de l'arbre abstrait sous forme d'un ensemble de relations n–aires ou binaires à base d'intervalles,
- un niveau interne sous la forme d'un graphe de contraintes qui représente des contraintes temporelles aussi bien numériques que symboliques à base d'instants [77].

L'approche adoptée dans Madeus repose donc principalement sur une mise en correspondance entre une représentation de haut niveau et une gestion interne de plus bas niveau fondée sur les instants. L'objectif est de ramener la représentation temporelle à un *STP* qui permet des traitements efficaces de la synchronisation (voir III.4.4.1). Signalons néanmoins qu'au fur et à mesure de la prise en compte des relations causales et de l'indéterminisme, nous nous éloignerons du cadre des *STP* classiques [110]. Nous reviendrons sur ces aspects dans la section V.3.2.4.

V.2.1 Représentation de l'information de base

Dans la classification des éléments multimédia de base introduite dans la section III.3.1, nous avons fait la distinction entre les éléments multimédia de base de type discret, continu et indéterministe. Dans notre modèle de contraintes, tous ces types d'éléments ainsi que les délais peuvent être représentés par une abstraction à base d'intervalles.

Chaque intervalle est décrit par une borne inférieure **l** (*lower bound*) et une borne supérieure **u** (*upper bound*) et est noté **[l, u]** (où $\mathbf{l} < \mathbf{u}$, **l** et **u** sont des valeurs positives et **u** pouvant valoir l'infini).

Les intervalles considérés dans Madeus peuvent être de deux types *contrôlables* ou *incontrôlables* :

> **Définition 1 :** *Un intervalle contrôlable* est un intervalle ayant une contrainte numérique sur la durée $\delta = \mathbf{[l,u]}_c$, et cette contrainte peut être réduite librement par le système d'édition entre **l** et **u**.

Ce type d'intervalle permet de modéliser le comportement des éléments de type continu et discret dont la durée de présentation peut être choisie dans un certain intervalle. Cette durée est bornée pour les éléments continus par des limites liées à l'intelligibilité de leur contenu, mais pour les éléments discrets comme le texte et les graphiques, elle peut être infinie. Ce type d'intervalle présente un comportement temporel élastique.

> **Définition 2 :** *Un intervalle incontrôlable* est un intervalle ayant une contrainte numérique sur la durée $\omega = \mathbf{[l,u]}_i$, et cette contrainte n'est connue qu'à l'exécution.

Ce type d'intervalle permet de modéliser le comportement indéterministe de certains éléments multimédia dont la durée de présentation ne peut être connue pendant la phase d'édition. Par exemple, pour les boutons d'interaction , la durée qui leur est associé correspondant à la période de temps ou ils sont activables, mais la valeur effective de ces durées ne dépend que de l'instant même de leur activation.

D'après ces deux types d'intervalles, nous pouvons distinguer deux propriétés qui caractérisent le scénario temporel d'un document multimédia : la **flexibilité** et la **contrôlabilité**.

La flexibilité mesure l'aptitude d'un scénario temporel à pouvoir être adapté aux contraintes numériques induites par les spécifications de l'auteur tout en garantissant sa

cohérence globale. La flexibilité d'un scénario dépend de l'élasticité des éléments multimédia (définie par la différence u–l des intervalles contrôlables) présents dans le document ainsi que de leur emplacement dans le graphe de contraintes.

Cette flexibilité du document peut être mise à profit pour compenser le comportement indéterministe qu'il contient (cf. V.3.2.4). Cela introduit la notion de contrôlabilité qui mesure l'aptitude d'un scénario à pouvoir être adapté dynamiquement, à l'exécution, au comportement indéterministe des éléments multimédia qu'il contient. Il s'agit donc d'une généralisation de la notion de cohérence pour les scénarios indéterministes [111].

V.2.2 Graphe de contraintes temporelles de Madeus

Arbre abstrait

Dans Madeus, l'arbre abstrait est défini par un ensemble d'éléments multimédia de base ou composés où :

- Les éléments multimédia de base correspondent aux feuilles de l'arbre, par exemple les éléments A, B, E, F, G et C dans la Fig. 5.2.
- Les éléments composés correspondent aux nœuds intermédiaires de l'arbre. Chaque élément composé c contient la description des relations temporelles qui lient ses fils directs, par exemple Doc, X et D dans la Fig. 5.2.

Cette description des relations temporelles peut être de deux types :

- Soit sous forme d'une liste de relations binaires r (e1, e2) qui lie toute paire d'éléments e1 et e2, et r ∈ R={before, meets, overlaps, ..} l'ensemble des relations temporelles binaires définies dans Madeus.
- Soit une relation n–aire, notée r^n qui lie tous les fils directs d'un élément composé c et $r^n \in R^n$ = {list_before, list_after, ..}, telle que pour tout ensemble ordonné de ces éléments dans l'arbre abstrait $E = \{e_1,.. ,e_n\}$, nous avons :

$$r^n (e_1,.. ,e_n) \Leftrightarrow \forall i\ (1 \leq i < n)\ \ e_i\ r\ e_{i+1}$$

Graphe de contraintes

À partir de l'arbre abstrait (voir exemple dans la Fig. 5.2) et les relations à base d'intervalles, nous définissons, pour chaque élément composé d'un document, un graphe de contraintes G = (I, A, E) à base d'instants, de la façon suivante :

- I représente un ensemble de sommets dont chacun correspond à un ou plusieurs instants de début et/ou de fin d'éléments simples ou composés du document.
- A représente un ensemble d'arcs orientés (i, j), i et j ∈ I, et chaque arc modélisant un élément du document ou un délai temporel.

- E représente un ensemble d'étiquettes. À chaque arc (i, j), on associe une étiquette de durée sous forme d'un intervalle de type contrôlable ou incontrôlable. Ces étiquettes sont notées [l, u], l et u étant des entiers positifs (voir V.2.1), ou D_{ij} si on veut y faire référence à partir des sommets du graphe (i et j ∈ I).

Notons que tous les intervalles sont représentés moyennant des bornes (min, max) de durées.

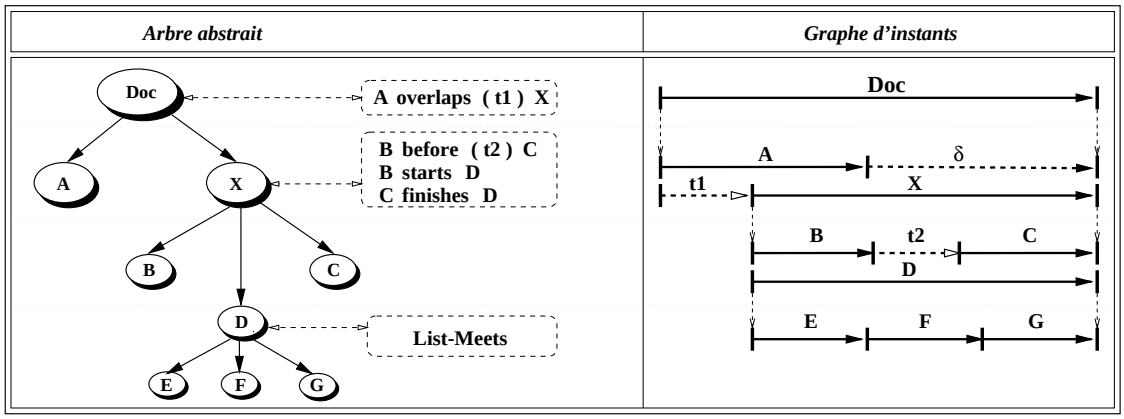


Fig. 5.2 : Arbre abstrait et graphe de contraintes dans Madeus

Deux instants abstraits de début et de fin sont attachés aux éléments composés du document, afin de délimiter leur contenu dans le temps. Ceci permet de définir pour chaque élément composé c un sous−graphe de contraintes $G_c = (I_c, A_c, E_c)$ qui est construit de la même façon que celui du document lui−même. Ainsi, nous pouvons ramener le traitement temporel pour tout un document à celui d'un seul niveau de la hiérarchie logique. Le graphe obtenu est orienté (voir Fig. 5.2) et acyclique (*Directed Acyclic Graph : DAG*) (voir la condition de cohérence qualitative présentée dans la section V.3.2.1). À chaque niveau de la hiérarchie logique, le graphe de contraintes correspondant possède un seul sommet **source** (l'instant début de l'élément englobant) et un seul sommet **puit** (l'instant de fin de l'élément englobant).

Chaînes temporelles

Le graphe G étant à base d'instants, il nous permet d'en avoir une vue supplémentaire sous forme d'un ensemble de chaînes temporelles. Ces chaînes sont définies de la façon suivante :

Définition : Une **chaîne temporelle** $Ch=\{a_1,.. , a_n\}$ est une séquence d'arcs contigus du graphe G tels que chaque arc a_j, $2 < j < n-1$, n'est relié qu'à un seul arc

prédécesseur et à un seul arc successeur. De plus, le sommet a^{+}_{j-1} est confondu avec le sommet a^{-}_{j}, chaque arc étant étiqueté par un intervalle contrôlable ou incontrôlable.

À toute chaîne Ch = $\{a_1,.. , a_n\}$, telle que a_j est étiqueté par $[l_j, u_j]$ dans le graphe, nous pouvons associer comme étiquette l'intervalle suivant :

$$[\sum_{j=1}^{n} l_j , \sum_{j=1}^{n} u_j]$$

On peut ainsi définir un graphe de chaînes noté G_{Ch}, obtenu à partir du graphe G en remplaçant la suite des arcs composant une chaîne Ch par l'arc Ch dans G_{Ch} (Fig. 5.3).

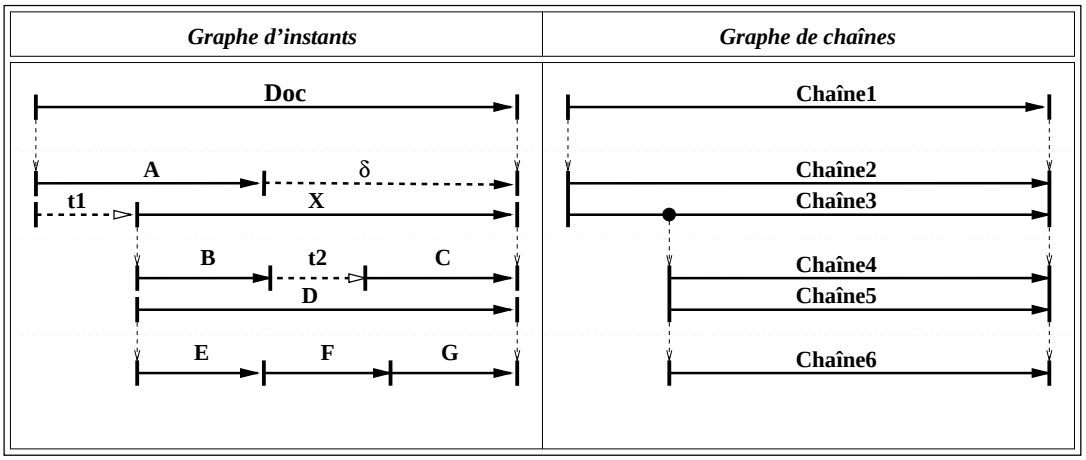


Fig. 5.3 : Graphe d'instants et graphe de chaînes dans Madeus

L'intérêt de cette structuration en chaînes dans Madeus est principalement lié à deux considérations. Elle permet d'améliorer les performances des opérations de mise à jour du document, car la plupart des traitements portent sur des chaînes et non pas directement sur des intervalles, en particulier, en ce qui concerne la vérification de la cohérence. Par ailleurs, les chaînes peuvent être mises à profit pour gérer l'indéterminisme pendant l'édition et pendant la présentation puisqu'elles constituent les branches parallèles du scénario.

V.2.2.1 Relations temporelles qualitatives et quantitatives

Comme nous l'avons évoqué dans le chapitre précédent, les relations retenues dans Madeus couvrent à la fois l'aspect qualitatif, quantitatif et causal des scénarios temporels. Pour être traitées, les relations temporelles considérées sont d'abord traduites sous forme de relations à base d'instants. Nous avons enrichi les relations temporelles de Allen [3] de la notion de délai, de manière à pouvoir transformer chaque relation d'intervalles dans le graphe G. Il en découle deux types de relations temporelles de base :

- Les relations qui n'impliquent pas de paramètre de délai.
- Les relations qui impliquent un paramètre de délai que l'auteur peut explicitement spécifier. Dans le cas où ce paramètre n'est pas précisé, il est défini par défaut lors de l'insertion de la relation.

Par exemple, pour la relation A before(t) B, le début de l'intervalle B a lieu à t unités de temps après la fin de l'intervalle A. Pour la relation, A overlaps B, si l'auteur ne précise pas le délai temporel t associé à cette relation, le délai est introduit par défaut avec des bornes [1, dur(A)] (la valeur 0 est exclue pour distinguer la relation overlaps de la relation starts). Afin de simplifier l'unité de mesure, nous considérerons dans la suite que les durées des intervalles d'un document sont exprimées moyennant la même unité de mesure (valeur discrète entière).

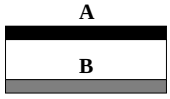
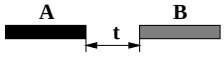
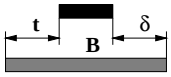
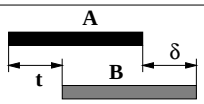
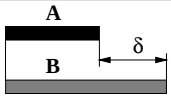
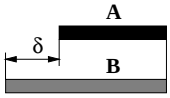
Relations	Représentation Graphique	Contraintes Numériques
A equals B		dur(A) = dur(B)
A before(t) B		____
A during(t) B		t < dur(B) - dur(A) & t > 0 & δ > 0
A overlaps(t) B		t > dur(A) - dur(B) & t > 0 & δ > 0
A meets B		____
A starts B		dur(A) < dur(B)
A finishes B		dur(A) < dur(B)

Fig. 5.4 : Les relations temporelles de base

La sémantique de ces relations est définie par un ensemble de contraintes entre les instants de début et de fin des intervalles. Ces contraintes font intervenir la durée et le type (contrôlable ou non) des intervalles associés aux éléments multimédia du document. La Fig. 5.4 illustre la correspondance entre les relations d'intervalles d'Allen, dotée

éventuellement d'un paramètre quantitatif de durée, et leur représentation sous forme d'un graphe à base d'instants, avec les contraintes associées. Les lignes verticales de la Fig. 5.4 représentent des contraintes de simultanéité entres les différents instants qui interviennent dans les relations.

V.2.2.2 Relations temporelles causales

Dans cette section, nous nous intéressons à la représentation des relations permettant d'exprimer des effets de causalité entre les différents instants du document. La Fig. 5.5 illustre les opérateurs supportés dans Madeus. Les flèches verticales de la figure signifient que l'occurrence d'un instant source provoque l'occurrence de l'instant cible.

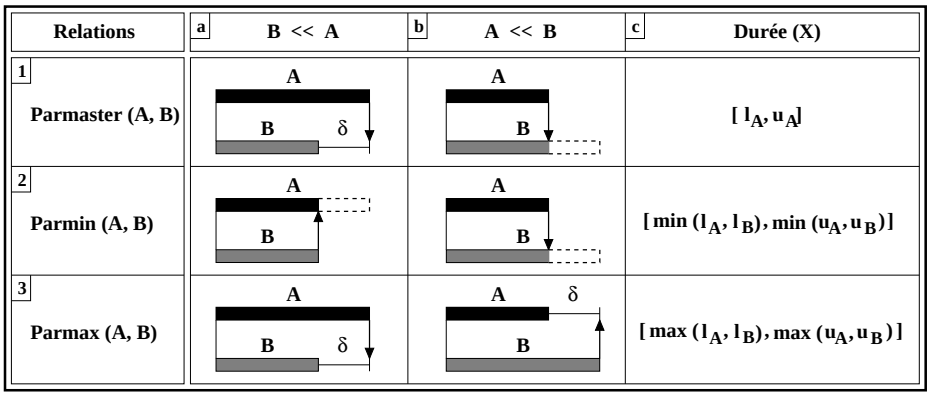


Fig. 5.5 : Les relations temporelles causales

Les relations causales dans Madeus ont la sémantique suivante :

- Parmaster (A, B) : signifie que l'occurrence de l'instant de fin de l'intervalle A provoque la terminaison de l'intervalle B. Si B est de type composé, la terminaison de A provoque la terminaison de tous les intervalles englobés dans B.
- Parmin (A, B) : signifie que l'occurrence de fin de l'intervalle A ou B provoque respectivement la terminaison de B ou A.
- Parmax (A, B) : signifie que l'occurrence de fin de l'intervalle de plus longue durée des deux éléments provoque la terminaison de la construction.

En fonction de leur durée, chacune des relations Parmaster, Parmin et Parmax peut avoir deux représentations graphiques, et donc deux transformations dans le graphe de contraintes (voir les colonnes b et c de la Fig. 5.5). Pour pouvoir les départager, il est nécessaire de définir une relation d'ordre entre les intervalles qui interviennent dans ces relations. Par exemple, pour deux intervalles A et B tels que A=[5,7] et B=[15,20], compte tenu que les durées possibles de A sont nécessairement plus petites que celles de B, il est facile de vérifier que ce cas correspond à l'illustration donnée dans la colonne b du

tableau. De façon plus générale, l'ordre partiel << entre deux intervalles est défini de la façon suivante :

$$\text{Pour } A = [l_A, u_A], B = [l_B, u_B], (\, l_A, u_A, l_B, u_B > 0\,), A << B \Leftrightarrow u_A < l_B$$

Cet ordre nous permet de déterminer laquelle des deux situations se présente. En particulier, lors de la transformation des relations Parmaster et Parmax en relations d'instants, nous pouvons savoir s'il faut ou non insérer un délai et à quel endroit du graphe. Dans le cas où l'ordre << ne peut être établi, le problème peut être contourné en encapsulant dans un élément composé X les deux intervalles A et B ainsi que la relation correspondante. Ensuite, le traitement avec le reste du document est réalisé à partir de cette nouvelle entité. Dans ce cas, l'intervalle associé à l'élément composé X, qu'il soit contrôlable où incontrôlable, possède des bornes de durées bien définies (voir la colonne c de la Fig. 5.5). Dans la section V.3.2.2, nous montrons comment le type (contrôlable ou incontrôlable) de l'intervalle X est déterminé. Cette information intervient dans la méthode de vérification de la cohérence causale proposée.

V.2.3 Primitives de manipulation des relations

Les primitives de manipulation des relations sont fournies dans le noyau de Madeus à travers deux fonctions d'insertion et de suppression. Ces fonctions sont appelées à partir de deux modules du système d'édition/présentation :

- à travers l'interface graphique lors d'une session interactive d'édition,
- par l'intermédiaire de l'analyseur syntaxique lors du chargement initial d'un document.

Dans le cas du chargement d'un document à partir d'un fichier, l'insertion des relations est réalisée de façon ascendante par rapport à l'arbre abstrait du document. Chaque fois qu'une relation est reconnue par l'analyseur syntaxique, elle est insérée dans le document.

Fonctions d'insertion et de suppression de relations

RelationId ← Crée_Relation (Parent, Intervalle1, TypeRelation, Paramètre_Temporel, Intervalle2).

Supprime_Relation (Parent, RelationId).

Ces deux fonctions permettent la création et la suppression d'une relation dans le document. L'opération d'insertion retourne l'identifiant interne d'une structure RelationId qui représente la relation et sa traduction sous forme de relations d'instants (voir la section V.2.4).

Parent : désigne l'élément composé de la structure logique sur lequel on veut insérer la relation.

Intervalle1, Intervalle2 : sont les identifiants symboliques des intervalles liés par cette nouvelle relation.

TypeRelation : indique quel est le type de relation temporelle liant les deux intervalles (représentation interne des relations).

Paramètre_Temporel : indique la valeur du délai lié dans la relation temporelle en question (ce délai est null si la relation ne fait pas intervenir de délai).

L'intérêt de garder en mémoire la structure RelationId est double : d'une part, dans le cas d'incohérence, cette structure permet de retrouver la relation d'intervalle incriminée et permet donc de retourner le bon message d'erreur à l'utilisateur. D'autre part, étant donné que le format de stockage des documents est basé sur les intervalles, cette structure permet de retrouver aisément la forme finale du document à partie de l'arbre abstrait.

Dans la suite de cette section, nous présentons le traitement qui est effectué lors de l'appel des fonctions présentées. Comme les relations n–aires n'interviennent qu'au niveau du format du document, et que leur traitement se ramène à des relations binaires, nous nous limiterons à exposer ce dernier cas de figure.

V.2.4 Traduction en relations d'instants

La traduction de chaque relation en relations d'instants est réalisée au moyen d'une fonction qui s'appuie sur un tableau interne. En remarquant que chaque relation binaire (a R b) fait intervenir deux intervalles, chacun étant formé par deux instants temporels, nous obtenons une configuration particulière pour chaque relation à partir des quatre instants résultants. Cette configuration est donnée dans le tableau Fig. 5.6 qui indique de façon précise les relations entre les instants de début noté a^- (resp. b^-) et les instants de fin notés a^+(resp. b^+) des intervalles a et b.

Les relations d'instants données dans le tableau sont organisées selon une suite de relations i1 r1 i2 r2 i3 r3 i4. Chaque suite correspond à une entrée du tableau et représente la conjonction de relations d'instants : (i1 r1 i2), (i2 r2 i3) et (i3 r3 i4) où chaque instant i correspond à un sommet existant ou à créer dans le graphe de contraintes, c'est à dire (a^-, a^+, b^-, ou b^+).

Les entrées du tableau qui représentent les instants et les relations d'instants ont été disposées conformément à leur ordre d'apparition temporel dans la relation d'intervalle (ordre de précédence $a^- < a^+$ et $b^- < b^+$).

Les relations d'instants (i1 r i2) que nous utilisons sont les suivantes :

- ? : indique l'absence de relation spécifique entre les deux instants,
- < : indique la précédence temporelle entre les deux instants,
- = : indique la simultanéité temporelle (l'égalité),
- $\rightarrow$: indique que l'occurrence de l'instant i1 provoque l'occurrence de l'instant i2.
- $\leftrightarrow$: indique que l'occurrence de l'instant i1 provoque l'occurrence de l'instant i2 ou vice–versa.
- $\hookleftarrow$: indique que l'occurrence de l'instant i1 est retardé jusqu'a l'occurrence de l'instant i2 ou vice–versa (point de rendez–vous).

Relation	**i1**	**r1**	**i2**	**r2**	**i3**	**r3**	**i4**
a finishes b	b^-	<	a^-	<	a^+	=	b^+
a starts b	a^-	=	b^-	<	a^+	<	b^+
a overlaps b	a^-	<	b^-	<	a^+	<	b^+
a during b	b^-	<	a^-	<	a^+	<	b^+
a meets b	a^-	<	a^+	=	b^-	<	b^+
a before b	a^-	<	a^+	<	b^-	<	b^+
a equals b	a^-	=	b^-	<	a^+	=	b^+
parmax(a, b)	a^-	=	b^-	<	a^+	$\hookleftarrow$	b^+
parmaster(a_m, b)	a_m^-	=	b^-	<	a_m^+	$\rightarrow$	b^+
parmin(a, b)	a^-	=	b^-	<	a^+	$\leftrightarrow$	b^+

Fig. 5.6 : Tableau de traduction en relations d'instants

Les relations '=', '$\rightarrow$', '$\leftrightarrow$' et '$\hookleftarrow$' entre deux instants i1 et i2 entraînent leur fusion sur un même sommet du graphe G. La relation '<' entraîne la création d'un arc entre les sommets i et j. Dans la suite de ce chapitre, on désignera un sommet par i ou j et on désignera l'arc reliant i à j par (i, j).

Dans l'application Madeus, la redéfinition ou l'ajout de nouvelles définitions de relations temporelles se limite à la simple mise à jour de cette table. Cela permet, à tout

moment, de modifier les relations temporelles supportées et l'interface graphique d'édition (palette de relations) sans avoir à apporter de grands changements aux traitements effectués par l'application.

V.3 Gestion des contraintes dans Madeus

Le graphe décrit précédemment est construit et modifié tout au long d'une session d'édition. Cette construction se fait progressivement suite aux différentes opérations effectuées par l'auteur. Les opérations qui ont des implications directes sur l'organisation temporelle d'un document sont l'insertion de nouveaux éléments multimédia et l'insertion ou la suppression de nouvelles relations ou de nouvelles contraintes sur les durées des différents éléments.

Les mises à jour du document ont généralement une portée locale. Cette portée est délimitée par les instants de début et de fin associés aux éléments. Les modifications introduites dans la structure temporelle susceptibles d'avoir lieu après chaque type d'opération d'édition sont détaillées dans la section V.3.1. Ce sont ces opérations qui permettent la construction incrémentale du graphe par des mises à jour locales, et c'est à partir de ces mises à jour qu'on peut déterminer la portion du graphe qui doit être vérifiée.

V.3.1 Manipulation du réseau de contraintes

Dans cette section, nous présentons les solutions apportées au problème d'insertion et de suppression de relations temporelles du document. Ces opérations induisent respectivement une restriction ou un relâchement des contraintes portées par le graphe.

V.3.1.1 Ajout de relations

L'insertion d'une nouvelle relation (A R B) dans un document apporte des changements de sa structure temporelle. Ces changements se traduisent par la modification de la topologie du graphe à partir des trois opérations élémentaires suivantes :

1. L'apparition explicite des sommets (instants) de début et de fin des intervalles A et B dans le graphe s'ils n'y figuraient pas déjà (cas de nouveaux éléments).
2. L'insertion, entre les sommets du graphe d'instants, d'arcs valués par les intervalles représentant les durées des délais et des éléments du document.
3. La fusion de certains sommets du graphe (cf. V.2.4).

Ces différentes opérations entraînent la création ou la suppression de nouvelles chaînes dans le graphe et/ou la modification de la liste des chaînes existantes. Des vérifications sont effectuées à deux niveaux :

- Le premier niveau de vérification est constitué par un premier filtrage des incohérences (cohérence *a priori*). Il est effectué indépendamment du reste du document et porte sur les conditions de cohérence présentées dans les Fig. 5.4 et Fig. 5.5.
- Le deuxième niveau de vérification est effectué à l'échelle du graphe tout entier. Ce niveau comporte les vérifications qualitatives, quantitatives, causales et indéterministes qui sont détaillées dans la section V.3.2.

Nous présentons d'abord comment l'insertion d'une nouvelle relation influe sur la structure du graphe. Cette insertion entraîne le deuxième niveau de vérification qui, prenant en compte le graphe du document, est plus générale que la première. L'algorithme décrivant le schéma général d'une opération d'insertion est décrit par le pseudo–code de la Fig. 5.7. Durant l'opération d'insertion, les différents traitements (lignes 3–4 de l'algorithme Fig. 5.7) peuvent mettre en évidence ces incohérences.

```
Vérifier_et_ajouter_relation (RelationId)
{
1. Si (Relation_cohérente_a_priori (Relation_Id))
2.      Alors  Créer_sommets(RelationId);
3.             Relier_sommets_au_graph(RelationId);
4.             Propager_contraintes();
5       Sinon  /* incohérence détectée */
6.             Retourner_erreur(RelationId);
}
```

Fig. 5.7 : Algorithme d'insertion de nouvelles relations

La fonction Créer_sommets() effectue la création de nouveaux sommets pour les intervalles A et B de la relation, s'ils ne sont pas déjà créés. Ces sommets sont les représentants de l'élément A et B dans le graphe de contraintes. Comme ce graphe est structuré en chaînes, les sommets des intervalles A et B sont d'abord regroupés dans des chaînes (une pour chaque intervalle Fig. 5.8) avant d'être insérés dans le graphe par la fonction Relier_sommets_au_graphe().

La fonction Relier_sommets_au_graphe() modifie la structure de graphe en y apportant les trois types de mise à jour cités auparavant : la création de sommets, l'insertion d'arcs ou la fusion de sommets existants du graphe. Ces opérations affectent la

configuration des chaînes existantes et/ou en créent de nouvelles. Les algorithmes de vérification s'appuient sur ces chaînes pour déterminer la cohérence de la relation insérée. Pour cette raison, les procédures mettant en œuvre ces trois opérations retournent la liste des chaînes modifiées.

Fusion de sommets

Dans ce qui suit, nous détaillons l'opération de fusion de sommets qui présente la plus grande complexité. Pour comprendre cet algorithme, nous allons dérouler pas à pas un exemple d'enchaînements d'insertion de relations.

L'exemple de la Fig. 5.8, donne une illustration graphique du fonctionnement de l'algorithme d'insertion de relations pendant une session d'édition. Au début le document est vide et l'auteur insère une relation A before (t1) B. Cette opération a pour effet l'apparition dans le graphe des arcs représentant A et B ainsi que le délai t1, sous forme d'une chaîne Ch1. Ensuite, l'insertion de la relation C meets D a pour conséquence l'apparition d'une chaîne parallèle Ch2, et ainsi de suite. L'insertion de la relation A finishes C dans notre exemple a engendré la fusion de deux sommets, produisant ainsi les quatre chaînes (Ch1, .., Ch4).

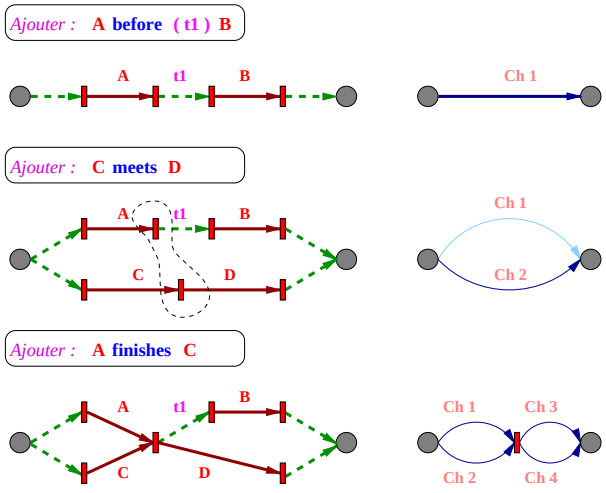


Fig. 5.8 : Gestion des chaînes temporelles

L'opération de fusion de deux sommets, qui intervient lorsqu'il y a une relation d'instant de type '=', '↔', '→' ou '↵' entre eux, est gérée par l'algorithme présenté dans la Fig. 5.9. Cet algorithme prend en entrée deux sommets s1 et s2 à fusionner. Il commence d'abord par éliminer les délais temporels qui peuvent éventuellement relier chacun de ces sommets au début ou à la fin de l'élément englobant (lignes 3–6 de l'algorithme). Ensuite, une phase supplémentaire permet de diviser les chaînes correspondant

aux sommets s1 et s2 en deux chaînes chacune (lignes 8–11). Celles–ci sont ensuite fusionnées sur un seul sommet s qui regroupe toutes les chaînes entrantes et sortantes de s1 et s2 (ligne 12). L'algorithme enregistre sur le sommet s la relation d'instant qui fait intervenir les extrémités des chaînes des sommets s1 et s2 (lignes 13–22) : la raison de la fusion. Enfin, si le résultat des modifications entraîne la création de deux chaînes, il les concatène pour en produire une seule (lignes 23–27).

```
Chaînes ← Fusionner_sommets ( s1, s2, Relation_instant )
 1. /* Enlever les intervalles de délai reliant les sommets */
 2. /* s1 et s2 aux instants source et puit de l'englobant  */
 3. Enlever_arc_depuis_début(s1);
 4. Enlever_arc_depuis_début(s2);
 5. Enlever_arc_depuis_fin(s1);
 6. Enlever_arc_depuis_fin(s2);
 7. /* Phase initiale de la fusion de deux sommets */
 8. si (Degré_entrant(s1) = Degré_sortant(s1) = 1)
 9.    Diviser_chaîne_en_deux(s1);
10. si (Degré_entrant(s2) = Degré_sortant(s2) = 1)
11.    Diviser_chaîne_en_deux(s2);
12. s ← Fusionner_sommets(s1, s2);
13. pour (Relation_instant) {
14.  '=':
15.  Ajouter_relation_au_sommet(s,(=,Chaîne(s1),Chaîne(s2)))
16.  '→':
17.  Ajouter_relation_au_sommet(s,(→,Chaîne(s1),Chaîne(s2)))
18.  '↔':
19.  Ajouter_relation_au_sommet(s,(↔,Chaîne(s1),Chaîne(s2)))
20.  '↵':
21.  Ajouter_relation_au_sommet(s,(↵,Chaîne(s1),Chaîne(s2)))
22. }
23. si (Degré_entrant(s) = Degré_sortant(s) = 1) {
24.     Chaîne1 = Chaîne_de_sommet_de_fin(s);
25.     Chaîne2 = Chaîne_de_sommet_de_début(s);
26.     Concaténer_chaînes(Chaîne1, Chaîne2);
27. }
28. Retourner (Toutes_les_chaînes_modifiées)
```

Fig. 5.9 : Algorithme d'insertion de nouvelles relations

En plus de sa fonction de construction du graphe, cet algorithme permet d'identifier les contraintes numériques à vérifier. Il s'agit des contraintes liées aux intervalles des nouvelles chaînes créées et/ou modifiées du graphe. Nous pouvons ainsi limiter le traitement de la cohérence aux seules chaînes en question. Dans l'exemple de la Fig. 5.8, cela revient à s'assurer que les contraintes Ch1 = Ch2 et Ch3 = Ch4 peuvent être vérifiées. Pour cette raison, l'algorithme retourne la liste des chaînes créées ou modifiées afin de les passer aux algorithmes de vérification détaillés dans la section V.3.2.

V.3.1.2 Retrait de relations

Dans la structure du graphe d'instants, le retrait d'une relation nécessite le traitement inverse de celui de l'insertion, à part que cette opération ne peut entraîner d'incohérences. Par conséquent, il n'est pas nécessaire de refaire une phase de vérification. Il suffit de défaire les modifications introduites dans le graphe par l'insertion de la relation retirée. Cela revient à effectuer trois opérations :

- Supprimer la relation du graphe d'intervalles du document.
- Supprimer les modifications topologiques que les relations d'instants correspondantes ont introduites dans le graphe.
- Relâcher les contraintes numériques que la relation a pu apporter aux autres chaînes du graphe (opération décrite en V.3.2.3).

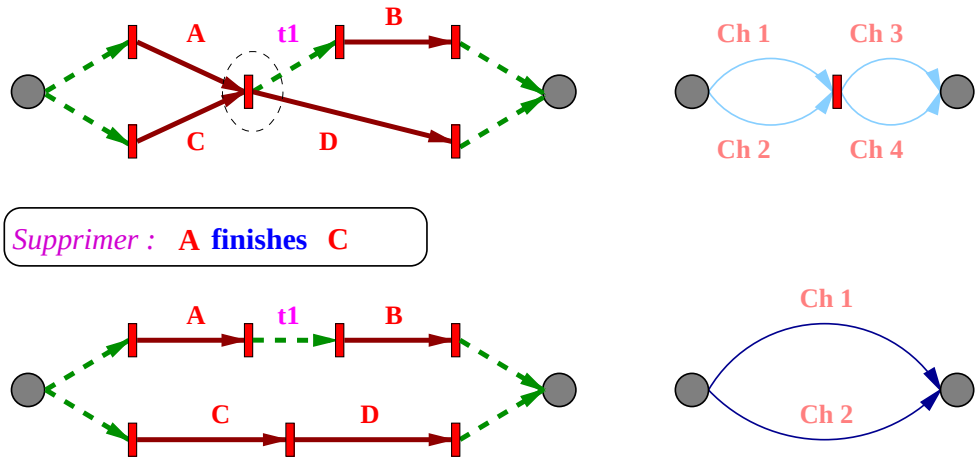


Fig. 5.10 : Mise à jour du graphe d'instants suite à une suppression de relation

Les deux premières opérations consistent à effectuer le traitement inverse de celui réalisé par l'algorithme de la Fig. 5.9. Signalons que les intervalles qui interviennent dans chaque relation supprimée ne disparaissent effectivement du graphe que s'ils ne figurent dans aucune autre relation du document (c'est le cas par exemple de l'intervalle A dans Fig. 5.10). Pour obtenir cette information, un compteur de références est utilisé, qui indique le nombre de fois qu'un intervalle figure dans une relation. Ce compteur est incrémenté/décrémenté à chaque insertion/suppression de relation qui met en jeu l'intervalle. Dès que ce compteur atteint la valeur nulle, l'intervalle en question et éventuellement les sommets du graphe correspondants sont supprimés.

La suppression d'un élément du document revient aussi à la suppression de relations du document. En effet, cette suppression entraîne celles de toutes les relations dans lesquelles figure l'intervalle correspondant. Ce qui, comme nous venons de le montrer, a aussi pour effet de supprimer du graphe l'intervalle correspondant ainsi que de relâcher les contraintes numériques qu'il a pu apporter.

V.3.2 Vérification de la cohérence

La modification d'un document multimédia par l'insertion ou la suppression d'éléments ou de relations ajoute de nouvelles contraintes dans le graphe. Ces contraintes peuvent conduire à une incohérence car les paramètres temporels des éléments et des relations (durées des éléments et des délais) doivent vérifier certains invariants liés à la progression du temps (causalité entre les différents événements) et à la coïncidence dans le temps des instants de fin des chaînes temporelles concurrentes (aspect quantitatif).

Supposons qu'une partie d'un document soit composée de trois éléments A, B et C (voir. Fig. 5.11−a), et que ces éléments soient liés par les relations (A *meets* B) et (B *meets* C). L'adjonction d'une nouvelle spécification C *overlaps* A rend le scénario incohérent. Ce cas correspond à une incohérence de type qualitatif car elle ne dépend pas des durées des éléments mais de la sémantique même des relations mises en jeu. En revanche, un scénario qui peut être cohérent du point de vue qualitatif peut être quantitativement incohérent. Par exemple (voir Fig. 5.11−b), si l'auteur spécifie que (A *meets* B) et (A *starts* C), l'adjonction de la relation (C *overlaps* B) rend le scénario incohérent pour des durées de A, B et C valant respectivement 30, 20 et 20 secondes. Finalement, dans l'exemple de la Fig. 5.11−c, on reprend le scénario précédent, mais avec un élément C dont l'intervalle de durée est de type incontrôlable de valeur [20, 40]. Ce cas correspond à une incohérence si, à l'exécution, la durée de l'élément C prend une valeur dans l'intervalle [20,30], par contre, le même scénario sera cohérent si la valeur de cette durée est dans

l'intervalle [30,40]. Ce cas est considéré comme une incohérence car cette contrainte ne peut pas être garantie dans tous les cas (cf. V.3.2.4).

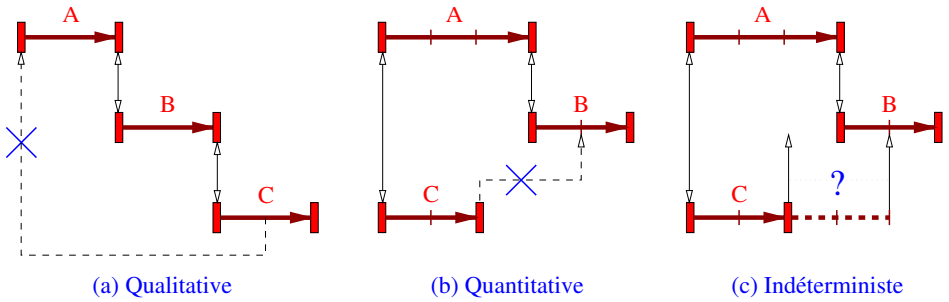


Fig. 5.11 : Les trois cas d'incohérences

Dans des scénarios plus complexes, les incohérences sont moins faciles à détecter que pour les trois exemples précédents, à cause des effets indirects des contraintes. Dans l'approche structurée des documents multimédia, la structure logique arborescente du document réduit considérablement la portée de ces contraintes, et par là même celle de la vérification. En effet, les éléments de base se trouvant au niveau des feuilles de l'arbre, une gestion ascendante de la cohérence par propagation des contraintes fournit un mécanisme efficace adapté au processus incrémental de l'édition.

Dans la suite de cette section, nous présentons les solutions mises en œuvre dans Madeus pour la vérification des quatre types de cohérence suivants :

- Cohérence qualitative.
- Cohérence causale.
- Cohérence quantitative.
- Cohérence indéterministe : contrôlabilité [111].

Les relations causales, prises indépendamment, n'introduisent pas d'incohérences. Par contre, leur insertion dans un scénario comportant des contraintes numériques peut le rendre incohérent. Dans la suite de ce chapitre, nous appelons la vérification de la cohérence causale le processus qui consiste à détecter ce type de situations.

V.3.2.1 Cohérence qualitative

La première phase de vérification qui est réalisée lors de l'insertion d'une relation temporelle est celle qui correspond au cas de la Fig. 5.11–a. Dans cet exemple, la traduction de la relation d'intervalle en relations d'instants sur la structure du graphe a pour effet l'introduction d'un cycle. Ce cycle est formé par les intervalles A, B et l'intervalle correspondant au délai t de la relation C *overlaps* A.

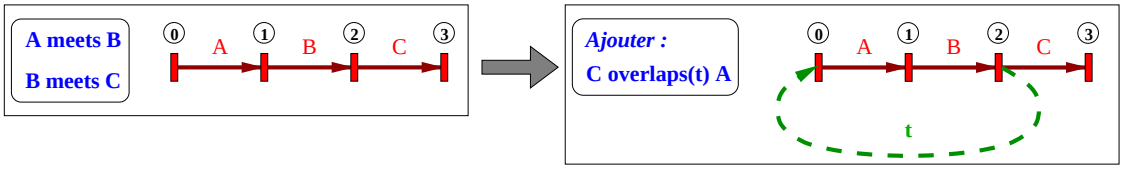


Fig. 5.12 : Exemple d'incohérence qualitative

Pour chaque relation d'intervalle introduite dans le document, la vérification de la cohérence qualitative consiste donc à vérifier que l'insertion de délais ainsi que la fusion des sommets qu'elle entraîne sur le graphe, n'introduisent pas de cycles. L'algorithme qui réalise cette fonction est décrit dans la Fig. 5.13. Notons qu'il n'est pas nécessaire de modifier le graphe pour réaliser cette opération (vérification *a priori*).

Dans cet algorithme, les instants i et j considérés sont ceux qui interviennent dans la relation `RelationId`, c'est–à–dire, si `RelationId` = (A R B), i, j ∈ { A^-, A^+,B^-, B^+}. `Intervalle(i)` est une fonction qui rend l'intervalle auquel appartient l'instant i : `Intervalle(i)` = A / i = A^+ ou i = A^-. `Sommet_du_graphe(i)` est une fonction qui retourne le sommet correspondant à i dans le graphe d'instants.

```
Boolean ← Cohérence_Qualitative (RelationId)
{
 1.  Pour chaque relation d'instants (i r j) ∈ RelationId
 2.      si (Intervalle(i) ≠ Intervalle(j))
 3.           alors s1 = Sommet_du_graphe(i)
 4.                 s2 = Sommet_du_graphe(j)
 5.                 si (Chemin_existe(s2, s1))
 6.                   alors retourne (Faux) /*Cycle détecté*/
 7.                   sinon retourne (Vrai)
}
```

Fig. 5.13 : Algorithme de vérification de cohérence qualitative

Dans le cas de l'exemple précédent (Fig. 5.12), la relation (C *overlaps* A) induit les relations d'instants : $C^- < A^- < C^+ < A^+$. Pour la relation i r j = $C^- < A^-$, s1 et s2 correspondent aux sommets du graphe représentant les instants i et j (fonction `Sommet_du_graphe()`), on a avec $A^- < C^-$, un cycle sur la même chaîne.

Pour empêcher la création de cycles, on vérifie s'il n'existe pas déjà un chemin dans le graphe entre les sommets que l'on doit relier lors de l'adjonction de la nouvelle relation (fonction `Chemin_existe` appelée dans Fig. 5.13). Cet algorithme est basé sur deux aspects du graphe qui permettent d'accélérer la détection de chemin :

1. Le maintien d'un tri topologique entre les sommets du graphe temporel G (comme sur la Fig. 5.12). Ce tri permet d'obtenir une fonction d'ordre partiel `rang` de I $\times$ I $\rightarrow$ ℵ tel que : $\forall$ s1, s2 $\in$ I, s1 < s2 $\Rightarrow$ `rang`(s1) < `rang`(s2) [60].
2. La structuration du graphe sous forme de chaînes temporelles.

L'ordre topologique et l'application de l'algorithme aux chaînes et non pas aux arcs permet de réduire considérablement le temps de recherche de chemins dans le graphe.

```
Booléen ← Chemin_existe (s1, s2)
{
 1.     /* les deux sommets sont sur la même chaîne temporelle */
 2.     si (Chaîne(s1) = Chaîne(s2))
 3.          retourne Vrai;
 4.     /* s'ils ne sont pas sur la même chaîne temporelle */
 5.     si rang(s1) < rang(s2)
 6.           sA = s1; sB = s2;
 7.     sinon sA = s2; sB = s1;
 8.     si (Degré_entrant(s1) = Degré_sortant(s1) = 1)
 9.           ch1 = Chaîne_du_sommet(s1)
 10.          sA  = Sommet_de_fin_de_la_chaîne(ch1)
 11.    si (Degré_entrant(s2) = Degré_sortant(s2) = 1)
 12.          ch2 = Chaîne_de_sommet(s2)
 13.          sB  = Sommet_de_début_de_la_chaîne(ch2)
 14.    si (sA = sB)
 15           retourne Vrai /* cycle trouvé */
 16.    sinon si rang(sA) ≥ rang(sB)
 17.             retourne Faux /* pas de cycles */
 18.          sinon
 19.             retourne Recherche_Chemin(sA, sB);
}
```

Fig. 5.14 : Algorithme de recherche de chemin

L'algorithme vérifie d'abord si les deux sommets se trouvent sur la même chaîne (2–3). Dans ce cas, il retourne immédiatement vrai car un chemin a été trouvé (c'est le chemin détecté entre A^- et C^- de l'exemple Fig. 5.12). Dans le cas contraire, la recherche d'un chemin est appliquée au niveau du graphe tout entier. Pour deux sommets s1 et s2 placés sur deux chaînes chaîne1 et chaîne2, tels que rang(s1) < rang (s2), la recherche consiste d'abord à comparer le rang des sommets extrémités des chaînes chaîne1 et chaîne2 respectivement. Pour la chaîne1, il s'agit du sommet de fin (sA dans l'algorithme) et pour la chaîne2 il s'agit du sommet de début (sB). Dans le cas d'égalité, on vérifie d'abord si les deux sommets ne correspondent pas au même sommet du graphe, auquel cas un chemin existe. Si le rang de sA est supérieur ou égal à celui de sB, il ne peut exister de chemin. Le dernier cas nécessite une recherche exhaustive sur le graphe (fonction `Recherche_Chemin`). Cette recherche est améliorée dans Madeus par deux mesures :

- Les pas de l'algorithme sont limités aux sommets extrémités des chaînes (début et fin).
- Le graphe est exploré par une recherche en largeur d'abord à partir de l'un des deux sommets.

L'algorithme présenté ici est très bien adapté à la vérification de scénarios multimédia grâce à son exploitation maximale de la localité temporelle des relations [38]. Dans le cas des documents multimédia, cette localité traduit le fait que les relations introduites par l'auteur concernent, souvent, des éléments qui se situent dans un certain voisinage temporel, en particulier des chaînes adjacentes. Notre algorithme tire ainsi profit de ce voisinage pour vérifier la cohérence qualitative de façon plus efficace.

V.3.2.2 Cohérence causale

Dans cette partie, nous abordons la vérification des contraintes causales du graphe de contraintes. Cette opération est effectuée à chaque fois que l'auteur insère une relation, binaire ou n–aire, de type Parmax, Parmin ou Parmaster. Cela se traduit par l'insertion, dans le graphe de contraintes, d'une relation causale à base d'instants entre deux ou plusieurs extrémités de fin de chaînes (cf. tableau de traduction de la Fig. 5.6).

La démarche adoptée dans Madeus consiste à réduire ces chaînes de façon à obtenir une seule chaîne qui représente cette portion du document dans le graphe. L'objectif est de ramener le processus de vérification au cas quantitatif ou indéterministe. Afin d'illustrer notre démarche, nous exposons des exemples de portions de graphes comportant des relations causales. Les graphes présentés sont constitués de trois chaînes (une par intervalle) A, B et C de durée $[l_A, u_A]$, $[l_B, u_B]$ et $[l_C, u_C]$ respectivement. Ces chaînes sont

liées selon les cas illustrés dans la Fig. 5.15 par les relations Parmin (A, B, C), Parmaster (B_m, A, C) et Parmax(A, B, C).

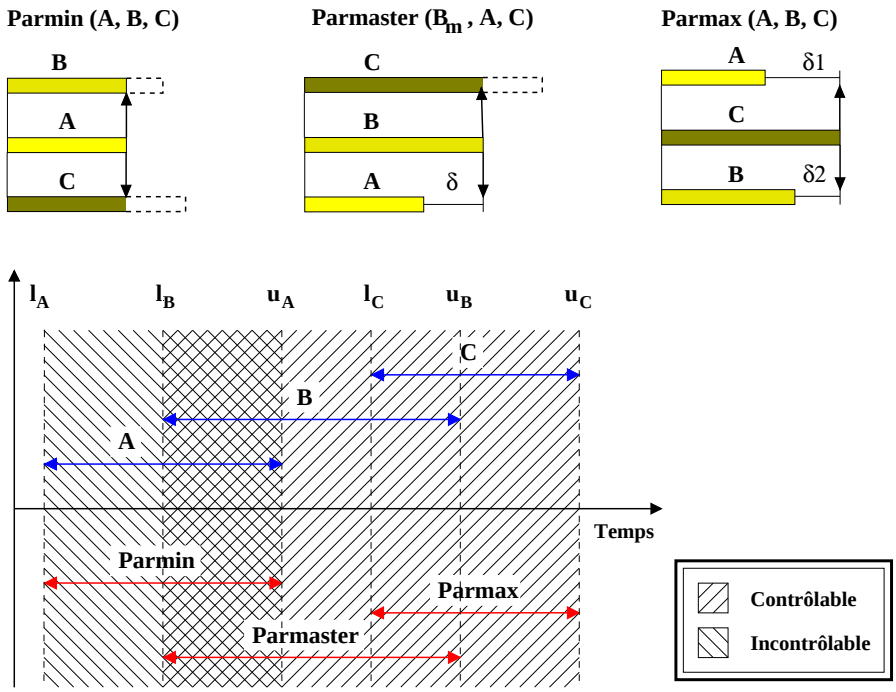


Fig. 5.15 : Vérification de la cohérence causale

L'illustration graphique de la Fig. 5.15 représente une des dispositions temporelles possibles pour chacune des trois relations en fonction des positions relatives des bornes des chaînes A, B et C. Nous avons en plus supposé l'incontrôlabilité de la chaîne A et la contrôlabilité des chaînes B et C (cf. les hachures dans la Fig. 5.15). La difficulté de gérer toutes les dispositions possibles est liée aux deux facteurs suivants :

- l'insertion des relations d'intervalles dans le graphe d'instants n'est pas unique à cause des intervalles incontrôlables comme nous l'avons évoqué dans la section V.2.2.2. Cette difficulté traduit la nature disjonctive des relations Parmaster et Parmax qui, en fonction des chaînes A, B et C, peut conduire à plusieurs transformations possibles. En particulier, si une ou plusieurs des chaînes A, B ou C est de type incontrôlable, la transformation ne peut pas être connue *a priori* (i.e à l'édition).
- la définition exacte des bornes et du type de la chaîne résultante est aussi une opération complexe, car les régions possibles (cf. Fig. 5.15), dénotent à la fois des zones de choix possibles (les intervalles contrôlables), des zones

d'indéterminisme (les intervalles incontrôlables) ou des zones combinaison des deux (zones doublement hachurées dans la Fig. 5.15).

Pour la relation Parmaster, la chaîne résultante correspond toujours à celle de B, qu'elle soit contrôlable ou non. Les chaînes A et C sont, dans ce cas, sous le « contrôle » de l'intervalle A. Par conséquent, A et C n'ont aucun effet sur le graphe de contraintes, nous les appellerons les chaînes « dominées ». Pour les relations Parmin et Parmax, la chaîne résultante est obtenue à partir des formules données dans la Fig. 5.5. Cet intervalle est considéré incontrôlable si l'intervalle résultant couvre une zone qui contient de l'indéterminisme, il sera contrôlable sinon. Cette solution nous ramène à un graphe ne comportant que des contraintes contrôlables ou incontrôlables. Nous avons ainsi préféré éviter les intervalles mi–contrôlables et mi–incontrôlables en considérant uniquement l'un des deux cas extrêmes. C'est dans un souci de garder un représentation et un traitement homogène du graphe de contraintes que nous avons pris ce choix. La vérification de la causalité est ainsi ramenée à la vérification quantitative et indéterministe.

V.3.2.3 Cohérence quantitative

Dans cette partie, nous développons les aspects liés à la vérification des contraintes numériques du graphe temporel. Cette opération est effectuée à chaque fois que l'auteur insère une nouvelle relation dans le document. Comme nous l'avons montré dans la section V.3.1, chacune de ces modifications se traduit par des insertions de délais temporels ou encore par la fusion de sommets existants du graphe. Ces opérations permettent d'identifier l'ensemble des chaînes qu'il faut vérifier du point de vue qualitatif. C'est cette liste de chaînes modifiées qui est passée à l'algorithme de vérification présenté ici. Cet algorithme est basé sur la propagation des contraintes numériques en utilisant les opérations de composition et d'intersection suivantes :

Pour chaque sommet i, j et k du graphe, D_{ij} représente la contrainte de la chaîne de sommets extrémités i et j :

$D_{ij} \otimes D_{jk} = (i : [a, b] : j) \otimes (j : [c, d] : k) = (i : [a+c, b+d] : k)$

$D_{ij} \cap D'_{ij} = (i : [a, b] : j) \cap (i : [c, d] : j) = (i : [\max(a, c), \min(b, d)] : j)$

L'algorithme que nous utilisons dans Madeus est fondé sur une version incrémentale [19] de l'algorithme DPC (*Directional Path Consistency*)[77]. Cet algorithme est plus performant que PC–2 [74]. Grâce au tri topologique du graphe, il permet de détecter toutes les incohérences quantitatives [77] en renforçant une forme restreinte de cohérence sur le graphe : la DPC–Cohérence :

Définition 3 (DPC−Cohérence) : Soit un graphe G = (I, A, E) un réseau de contraintes numériques trié topologiquement. Le réseau G est DPC−cohérent si la condition suivante est satisfaite :

$\forall$ i, j, k $\in$ I, (rang(i) < rang(k)) $\wedge$ (rang(j) < rang(k)) $\Rightarrow$ $D_{ij} \subseteq D_{ik} \otimes D_{kj}$

Avant d'aborder le déroulement détaillé de cet algorithme, nous commençons d'abord par une illustration graphique de son principe de fonctionnement, en particulier, par le parcours du graphe effectué à chaque insertion ou modification d'une chaîne temporelle.

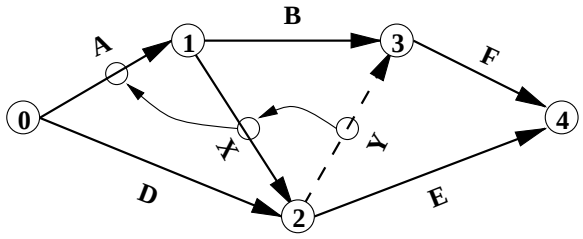


Fig. 5.16 : Parcours de l'algorithme DPC−incrémental

Considérons un graphe composé de sept chaînes A, B, D, E, F, X et Y (voir la Fig. 5.16), et soit Y la chaîne qui, après l'insertion d'une relation dans le document, a été modifiée resp. créée dans le graphe et qui doit être vérifiée quantitativement. En prenant en compte l'ordre topologique du graphe, le principe de l'algorithme consiste à vérifier ou renforcer la propriété de DPC−Cohérence en examinant ou modifiant, si c'est possible, la chaîne X. Si celle−ci est modifiée, l'algorithme examine la chaîne A et ainsi de suite à l'échelle du graphe tout entier.

L'algorithme permet de prendre en compte une ou plusieurs chaînes à la fois. Il suffit de l'initialiser avec la/les chaînes en question. Pour toute chaîne Ch à vérifier, on désignera par Ch^-et Ch^+ les sommets de début resp. de fin de Ch. L'algorithme exploite les aspects suivants du graphe temporel afin de détecter rapidement les incohérences quantitatives :

- Pour tout triplet de sommets du graphe (i, j, k) extrémités de chaîne tels que rang(i) < rang(k) et rang(j) < rang(k) et rang(Ch^+) < rang(k), les chaînes correspondantes à ces sommets ne sont pas modifiées par cette nouvelle contrainte ($D_{ij} \subseteq D_{ik} \otimes D_{kj}$). Ceci signifie que les sommets k du graphe peuvent être ignorés par cette opération de vérification.

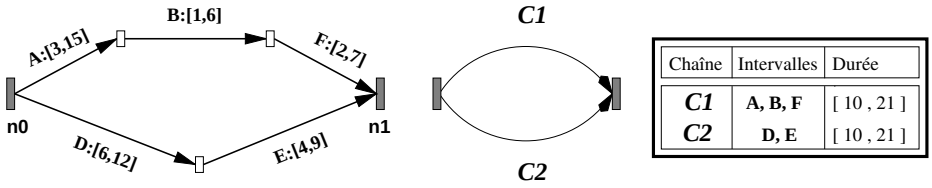
- Une contrainte qui est modifiée sur une chaîne Ch ne peut affecter que les contraintes entre le sommet Ch^- et d'autres sommets k tels que rang(k) < rang(Ch^+). Donc, il suffit de limiter la vérification aux sommets du graphe qui vérifient cette condition.

```
Booléen ← DPC_incrémental
                (GCh=(ICh,ACh,ECh), Chaînes_modifiées : C)
{
1. Liste =  ∅ ;
2. pour chaque Ch ∈ C {
3.  Liste = Liste ∪ {Sommet_de_fin_de_la_chaîne(Ch)}
4.  Ch.Modifié = Vrai;
5. }
6. /* liste contient tous les sommets de fin des chaînes
7.    modifiées */
8. tant que Liste ≠ ∅ {
9.   k ← Sommet_de_plus_haut_rang(Liste);
10.   Liste ← Liste − {k};
11.   Pour chaque arc a1 = (i, j) | rang(j) < rang(k) et
12.      a2 = (i, k), a3 = (j, k) ∈ ACh
13.      si a2.Modifié ou a3.Modifié alors
14.     {
15.       Temp ← Dij ;
16.       Dij ← Dij ∩ ( Dik ⊗ Dkj);
17.       ACh ← ACh ∪ {(i, j)};
18.       si Dij = ∅ alors retourne (Faux)
19.       si Dij ≠ Temp alors {
20.      /* la chaîne X est modifiée, il faut répercuter ses
21.         modifications sur les autres chaînes */
22.             List ← List ∪ {j};
23.             a1.Modifié = Vrai;
24.     }
25. }
26.retourne (Vrai);
 }
```

Fig. 5.17 : Algorithme de vérification de cohérence quantitative

Cet algorithme utilise une structure de données Liste pour stocker les sommets qui doivent être vérifiés. Ces sommets seront traités selon leur ordre topologique décroissant dans le graphe. Pour cette raison, à chaque fois qu'un nouveau sommet doit être ré–examiné, il est inséré dans la structure Liste en respectant cet ordre. Pour chaque chaîne insérée dans le graphe et qui doit être vérifiée, la Liste est initialisée avec le sommet de fin de cette chaîne. Pour un sommet k, il suffit de considérer des paires de sommets (i, j) tels que l'une des chaînes de i à k ou de j à k a été préalablement modifiée. Les chaînes modifiées sont marquées moyennant un champ (Modifié) qui permettra à l'algorithme d'identifier celles en cours de propagation.

Nous illustrons maintenant le fonctionnement de cet algorithme sur l'exemple de la Fig. 5.16 avec les valeurs de durées suivantes A:[3,15], B:[1,6], F:[2,7], D:[6,12] et E:[4,9]. Nous partons d'une configuration initiale (sans les intervalles X et Y) où ces intervalles sont disposés sur deux chaînes parallèles C1 et C2, où C1 est composée de la séquence {A, B, C} et C2 de la séquence {D, E} (voir Fig. 5.18).

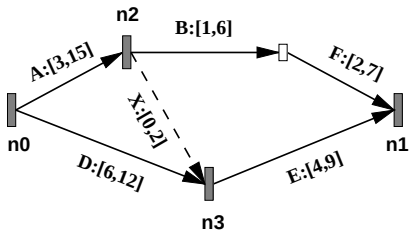


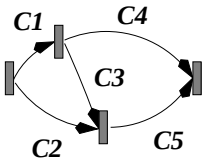
Chaîne	Intervalles	Durée
C1	**A, B, F**	[10 , 21]
C2	**D, E**	[10 , 21]

Fig. 5.18 : Algorithme de cohérence quantitative : état initial

Après l'insertion de la relation A before [0,2] E nous obtenons une nouvelle topologie du graphe qui conduit à l'insertion d'un délai. Cette relation se traduit sur le graphe par l'insertion d'un délai X entre le sommet n2 et n3. Cette insertion modifie la configuration des chaînes existantes du graphe en créant une nouvelle chaîne C3 (celle correspondant au délai X), divise la chaîne C1 en deux nouvelles chaînes C1 et C4 (voir Fig. 5.19) ainsi que la chaîne C2 en deux nouvelles chaînes C2 et C5. Comme cette relation n'introduit pas d'incohérence qualitative, nous appliquons maintenant l'algorithme de vérification quantitative pour nous assurer de la validité du nouveau scénario.

Après traduction en relations d'instants et insertion dans le graphe, les chaînes touchées par cette modification sont (C1, C2, C3, C4, C5) (Fig. 5.19). Il suffit alors d'appliquer l'algorithme en prenant initialement les chaînes en question. Notons qu'étant donné que la structure Liste contient des sommets, il suffit d'insérer une seule fois un sommet de fin qui appartient à plusieurs chaînes modifiées.





Chaîne	Intervalles	Durée
C1	**A**	[4 , 12]
C2	**D**	[6 , 12]
C3	**X**	[0 , 2]
C4	**B, F**	[3 , 13]
C5	**E**	[4 , 9]

Fig. 5.19 : Deuxième phase de l'algorithme

L'application de l'algorithme à l'exemple précédent se déroule alors de la façon suivante (les différents pas sont préfixés par le numéro de ligne de l'algorithme) :

C = {C1, C2, C3, C4, C5}

(5) Liste = $\{C1^+, C2^+, C4^+\}$

(9) k = C4$^+$

(10) Liste = $\{C1^+, C2^+\}$

(11) a1 = C3

(15) Temp = [0, 2]

(16) C3.durée = [0, 2] ∩ ([3, 13] ⊗ [−9, −4]) = [0, 2] (non modifié)

(9) k = C2$^+$

(10) Liste = $\{C1^+\}$

(11) a1=C1

(15) Temp = [3, 15]

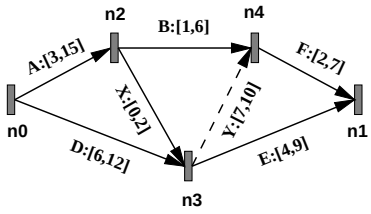
(16) C1.durée = [3, 15] ∩ ([6, 12] ⊗ [−2, 0]) = [4, 12] (modifié)

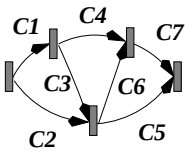
(23) C1.Modifié = Vrai

(9) k = C1$^+$

(10) Liste = ∅

(26) L'insertion de la chaîne est donc quantitativement cohérente.





Chaîne	Intervalles	Durée
C1	A	[4 , 12]
C2	D	[6 , 12]
C3	X	Φ
C4	B	[3 , 13]
C5	E	[4 , 9]
C6	Y	[7 , 7]
C7	F	[2 , 7]

Fig. 5.20 : Troisième phase de l'algorithme

Considérons maintenant l'insertion d'une nouvelle relation : D before [7, 10] F dans le scénario temporel de la Fig. 5.20. Cette relation se traduit sur le graphe par l'insertion d'un délai Y entre le sommet n3 et n4. Cette insertion modifie la configuration des chaînes existantes du graphe en créant une nouvelle chaîne (celle correspondant au délai Y) et en divisant la chaîne C4 de la Fig. 5.19 en deux nouvelles chaînes C4 et C7 (voir Fig. 5.20).

Initialement la liste Liste est composée des sommets de fin de chaînes qui viennent d'être modifiées et de la chaîne créée (celle correspondant au délai Y) :

C = {C4, C5, C6, C7}

(5) Liste = $\{C4^+, C5^+\}$

(9) k = $C5^+$

(10) Liste = $\{ C4^+\}$

(11) a1 = C6

(15) Temp = [7, 10]

(16) C6.durée = [7, 10] ∩ ([4, 9] ⊗ [−7, −2]) = [7, 7] (modifié)

(23) C6.modifié = Vrai

(9) k = $C4^+$

(10) Liste = ∅

(11) a1 = C3

(15) Temp = [0, 2]

(16) C3.durée = [0, 2] ∩ ([1, 6] ⊗ [−10, −7]) = ∅

(18) Cette nouvelle modification est donc quantitativement incohérente.

Dans ce dernier cas, dès que l'algorithme détecte une incohérence, Madeus remet à jour l'état du graphe tel qu'il était initialement (avant ces modifications), ce qui revient à la suppression de la nouvelle chaîne et au reétiquetage des chaînes avec les valeurs qu'elles avaient avant la modification.

Nous avons vu à travers ces exemples le fonctionnement de cet algorithme qui permet la gestion des contraintes numériques d'un scénario temporel. Les avantages qu'il offre sont multiples :

- L'incrémentalité : l'algorithme propage les contraintes de façon incrémentale sur le graphe en exploitant la localité des opérations de modification de sa topologie (fusion et insertion de nouvelles chaînes).
- La performance : la complexité de cet algorithme est linéaire $O(n\,k^2)$ [19] comparée à la complexité cubique : $O(n^3)$ de PC−2 (la valeur de la constante k représente de degré moyen d'un sommet du graphe [77]).

Suppression de relations

Le parcours de l'algorithme peut être réutilisé pour la suppression de relations temporelles. Cette opération, dont nous nous limitons à expliquer le principe de fonctionnement, revient à la suppression de chaînes du graphe et au recalcul des valeurs de durée d'une partie du graphe. La suppression de chaînes nécessite donc deux étapes :

1. Identifier les chaînes temporelles qui ont pu subir, et donc porter de façon directe ou indirecte, des contraintes numériques apportées par la chaîne enlevée/modifiée.
2. Retrouver un état du graphe où la contrainte numérique subie par chaque chaîne identifiée prenne en compte cette suppression (c'est la réhabilitation de contraintes car celles qu'on cherche à retrouver sont nécessairement moins restrictives).

Comme l'algorithme DPC_incrémental propage les contraintes dans une seule direction, il suffit d'exploiter son parcours pour identifier les chaînes temporelles à réhabiliter, puisque c'est ce même parcours qui a pu leur apporter des restrictions numériques, c'est−à−dire une modification de la valeur de durée initiale. Malheureusement, ce parcours ne permet pas d'identifier avec précision les sources des restrictions numériques [7], car comme le montre l'exemple de la Fig. 5.21, la suppression de la chaîne Y n'indique pas si les contraintes portées par la chaîne X et par la chaîne A ont été, à l'origine, provoquées par la chaîne Y ou par la chaîne E (voir le parcours possible de l'algorithme DPC_incrémental). Ceci nous oblige à remettre en cause en cascade toutes les sources potentielles de ces restrictions, ce qui a pour effet de supprimer X et A ensuite de les

réinsérer avec leurs valeurs de départ ; la suppression revient donc à une opération d'insertion.

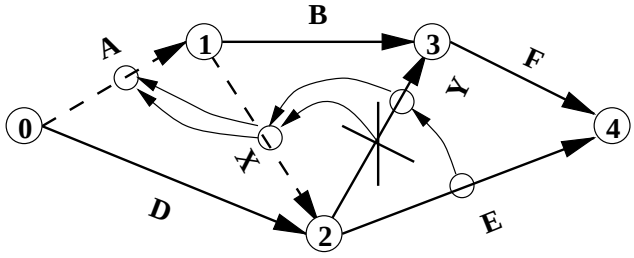


Fig. 5.21 : Effet de la suppression d'une chaîne du graphe

Cette solution, qui peut paraître peu performante, n'est en réalité pas critique. Car l'opération de suppression d'une relation ou d'un élément n'intervient qu'à l'édition, et donc de façon interactive et limitée en nombre, contrairement au processus de construction du graphe (l'insertion d'éléments et de relations) pour des fins de présentation, qui porte sur le document tout entier.

En conclusion, l'algorithme proposé, contrairement à PC−2, permet la vérification quantitative sur le graphe, sans maintenir systématiquement le graphe minimal de contraintes. Cela permet de détecter plus rapidement les incohérences quantitatives. De plus l'algorithme peut être appliqué sur la totalité du graphe, au moment de son chargement initial, tout en gardant de bonnes performances [77]; il est donc aussi bien adapté à l'édition qu'à la présentation.

V.3.2.4 Cohérence indéterministe : contrôlabilité

Ayant considéré le problème de la cohérence pour les intervalles contrôlables, nous abordons maintenant les problèmes liés à la prise en compte des intervalles indéterministes. La spécification temporelle présentée jusqu'ici traite le problème de cohérence pour les instants dont l'occurrence à l'exécution est entièrement sous le contrôle du système de présentation (les dates de présentation des différents éléments multimédia sont modélisés par des intervalles contrôlables). Le principe de vérification quantitative utilisé s'appuie sur la propagation de contraintes qui réduit la durée de ces intervalles afin de garantir l'existence d'une solution. Or pour les éléments modélisés par des intervalles incontrôlables, les durées correspondantes ne peuvent pas être modifiées car elles dépendent de facteurs externes au système de contraintes (interaction de l'utilisateur, exécution de programmes, délais variables dus aux accès réseau, etc). Prendre en compte des intervalles incontrôlables amène donc à considérer des contraintes qui sortent littéralement du cadre formel des STP [110] [35].

L'objectif de la vérification de cohérence indéterministe est d'assurer statiquement que le scénario trouvera une solution à l'exécution. Dans ce qui suit, nous allons simplement décrire de façon informelle la nature des vérifications que nous effectuons dans quelques cas bien identifiés. Une solution plus globale à ce problème n'a pas encore été apportée et nécessite des études plus poussées [111].

Notion de point d'observation

Durant la phase de présentation d'un document, les différents événements, ordonnancés à partir du graphe, sont produits dans l'ordre topologique croissant depuis l'instant de début du graphe jusqu'à sa fin. À tout instant, le graphe du document est partitionné en deux sous−graphes (voir Fig. 5.22) : le premier représente un passé « connu », car tous les événements ont déjà eu lieu, et le second représente un futur qui demeure, à ce moment, quantitativement « incertain », à cause justement des durées des intervalles incontrôlables.

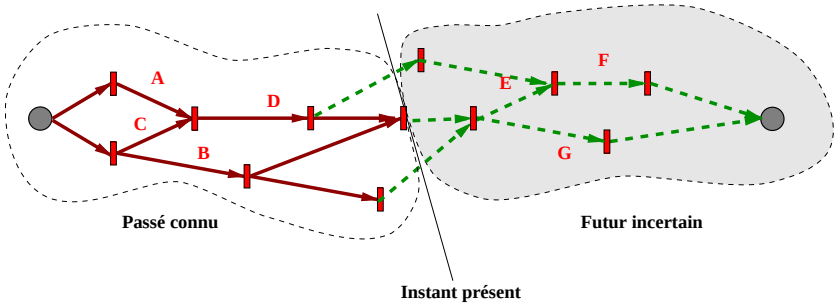


Fig. 5.22 : Schéma de progression du temps dans un scénario

Alors que les dates de fin des intervalles contrôlable sont prévisibles, celles des intervalles incontrôlables ne le sont pas et la durée de ce type d'intervalles ne peut donc être déterminée qu'à l'instant même où l'événement de fin se produit : ces instants définissent ainsi des **points d'observation** privilégiés à partir desquels il va falloir mesurer l'impact sur le scénario en cours d'exécution, en particulier sur les contraintes numériques qu'il contient.

Principe de l'algorithme

Notre approche consiste alors à examiner chaque point d'observation pour déterminer statiquement si nous pouvons assurer la cohérence du scénario quelles que soient les valeurs prises par les intervalles incontrôlables qui précèdent ce point (on vérifie donc la contrôlabilité) [64]. Cela revient à examiner l'état de la présentation pour identifier les

éléments contrôlables futurs sur lesquels nous pouvons agir. La difficulté du problème est principalement due à deux facteurs :

1. L'**irréversibilité du temps** : une fois que la valeur effective d'un intervalle incontrôlable a été observée, on ne peut plus revenir sur les durées attribuées aux intervalles contrôlables qui, au moment de cette observation, font déjà partie du passé.
2. La nature **fortement combinatoire** du problème : comme nous nous trouvons encore dans la phase d'édition, offrir des garanties sur la contrôlabilité du scénario revient non seulement à prédire toutes les situations d'incertitude (pour la présentation) qui peuvent se cumuler, mais en plus, vérifier s'il est possible de les résoudre.

Pour ce faire, nous nous fondons sur une vérification au cas par cas qui s'appuie sur la structure du graphe. L'objectif est de traiter ces différents cas selon leur ordre de difficulté et de ramener le graphe à une représentation qui comporte uniquement des intervalles contrôlables. Nous avons identifié trois cas que nous pouvons traiter.

Cas 1

Le premier cas correspond aux situations dans lesquelles la présence d'un intervalle incontrôlable n'est pas gênante : cet intervalle n'affecte pas les autres contraintes spécifiées par l'auteur. Un tel cas est illustré dans la Fig. 5.23.

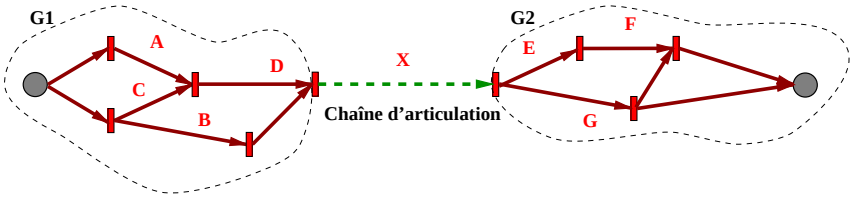


Fig. 5.23 : Exemple du Cas1

Dans cet exemple, le graphe peut être subdivisé en deux sous−graphes G1 et G2 qui ne sont reliés que par une chaîne contenant l'intervalle indéterministe (Chaîne d'articulation). Ce genre de cas se présente dans des scénarios où, par exemple, le passage entre deux parties est effectué par l'activation d'un élément de type bouton, de durée incontrôlable. Dans une telle configuration, il est clair que quelle que soit la valeur effective de durée prise par cet intervalle la cohérence du graphe n'est pas remise en question.

Cas 2

Le deuxième cas correspond à des situations dans lesquelles l'intervalle incontrôlable se trouve sur une chaîne dont l'extrémité droite correspond à un point d'articulation. Autrement dit, cet intervalle ne remet pas en cause les autres contraintes spécifiées par l'auteur, à l'exception des contraintes de coïncidence temporelle, telles que celles des chaînes B et D avec la chaîne incontrôlable X dans la Fig. 5.24.

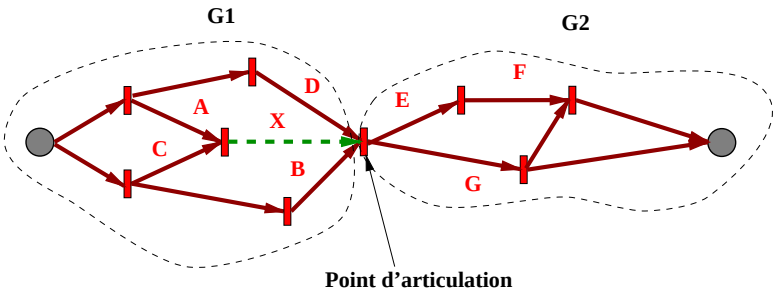


Fig. 5.24 : Exemple du Cas 2

Dans cet exemple, le graphe peut être subdivisé en deux sous−graphes G1 et G2 qui ne sont reliés que par l'instant de fin de la chaîne contenant l'intervalle indéterministe X (le point d'articulation est X^+). Ce type de situation ne peut être cohérent que si les chaînes B et D sont flexibles par la fin, comme le sont des éléments discrets (image ou texte). Dans ce cas, la terminaison de la chaîne X « provoque » la terminaison des deux autres chaînes contenant B et D. Nous nous ramenons donc au cas 1 moyennant cette simple vérification.

Cas 3

Le dernier cas consiste à vérifier, pour chaque instant permettant d'observer la durée d'un intervalle indéterministe, s'il est possible de le compenser. La compensation est reposée sur l'utilisation de la flexibilité des éléments contrôlables dont les instants de début ou de fin se trouvent dans le futur par rapport à cet instant. Ces instants sont appelés les **points de recouvrement**. Soit par exemple une chaîne constituée de deux intervalles A et B (voir Fig. 5.25−a). A est un intervalle incontrôlable $A = \mathbf{[l_A, u_A]}_i$ et $\omega = \mathbf{(u_A - l_A)}$ représente son montant d'indéterminisme. B est un intervalle contrôlable $B = \mathbf{[l_B, u_B]}_c$ et $\delta = \mathbf{(u_B - l_B)}$ représente son montant de flexibilité. Si la condition $\delta \geq \omega$ est satisfaite, alors l'indéterminisme peut être compensé dynamiquement de façon à rendre toute la chaîne contrôlable. Il suffit dans ce cas d'utiliser la flexibilité δ de façon à rendre la durée de l'intervalle A constante et égale à $\mathbf{u_A}$, ce qui a pour effet d'« éliminer » l'effet de cet indéterminisme. On obtient ainsi une chaîne ayant un comportement contrôlable

vis–à–vis du reste du graphe et de durée totale $[\mathbf{u_A+l_B}, \mathbf{u_A+u_B}]_i$. La cohérence quantitative de cette chaîne dans le graphe sera donc équivalente à sa contrôlabilité.

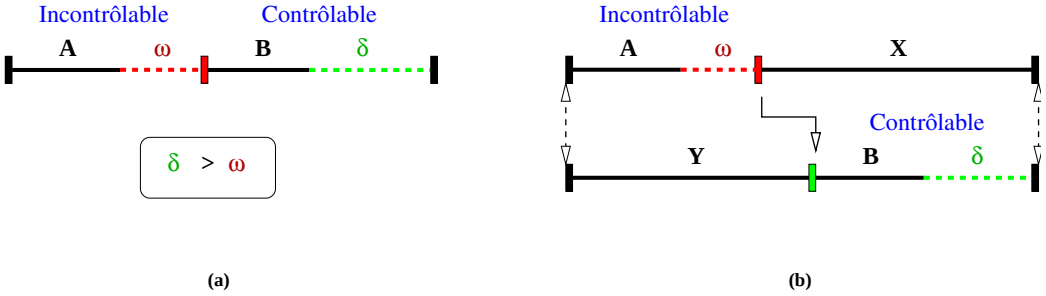


Fig. 5.25 : Gestion de la cohérence pour les éléments incontrôlables

Si l'intervalle incontrôlable ne peut être compensé au niveau de sa chaîne [64], la démarche proposée dans Madeus consiste à vérifier si une exploitation de la flexibilité dans les chaînes concurrentes peut permettre de réajuster le scénario. Par exemple, dans le scénario de la Fig. 5.25–b, deux chaînes parallèles {A, X} et {Y, B} doivent être de durées égales. Pour vérifier que cette contrainte peut être satisfaite, il suffit de s'assurer que les instants de début et de fin de chaînes ont lieu à la même date. Là encore, il est facile de voir que si la flexibilité δ de la chaîne concurrente {Y, B} est supérieure à l'indéterminsme ω de la chaîne {A, X} (condition 1), et si la date de l'instant d'observation A^+ a lieu avant le point de recouvrement B^-, alors la simultanéité des instants B^+ et X^+ peut être assurée. Cette portion du graphe est donc contrôlable. La méthode proposée entraîne, pour le graphe de contraintes, l'adjonction d'une durée indéterministe ω sur l'élément englobant. Elle entraîne aussi la suppression de la flexibilité utilisée des chaînes concurrentes. Dans l'exemple de la Fig. 5.22, cela se traduit par un allongement potentiel de toutes les chaînes concurrentes d'une durée ω, ce qui revient à décaler la partie G2 du graphe (c'est le sous graphe qui représente le futur par rapport à l'instant de fin de la chaîne indéterministe).

Le processus de vérification proposé dans cette section ne couvre pas tous les cas. L'approche adoptée dans Madeus se limite à vérifier localement la possibilité de « réparer » le scénario à partir de l'instant de fin des intervalles incontrôlables, puisque les garanties qu'on souhaite offrir à l'auteur au moment de l'édition, en terme de contrôlabilité, correspondent à celles que nous pouvons effectivement tenir au moment de la présentation [64].

La vérification de la contrôlabilité est un problème fondamental qui reste encore aujourd'hui un sujet de recherche ouvert. Des méthodes « préventives » ont fait l'objet d'une

étude dans le cadre de la planification robotique [110][109][31]. Cette étude s'oriente maintenant vers des méthodes de recherches exhaustives similaires à celles employées dans le domaine du temps réel [111].

V.3.3 Formatage temporel statique ou recherche de solution

La recherche de solution pour un scénario temporel, appelée aussi formatage temporel par analogie avec le formatage spatial, consiste à affecter des valeurs aux différents intervalles du graphe. Cette opération, contrairement aux précédentes, n'est nécessaire que lorsque l'utilisateur lance la présentation du document. Elle consiste à calculer d'abord les contraintes minimales des différentes chaînes, puisque ces dernières représentent les seules valeurs susceptibles de figurer dans la solution finale, ensuite le calcul de cette solution appelée document formaté.

Dans cette section, nous présentons donc deux étapes qui permettent d'aboutir à un document temporellement formaté : l'algorithme qui permet de calculer le réseau de contraintes minimal, et le processus de calcul d'une solution définitive, c'est–à–dire le formatage.

V.3.3.1 Réseau de contraintes minimal

Le calcul du réseau de contraintes minimal consiste à éliminer les valeurs d'intervalles des chaînes du graphe G qui ne figurent dans aucune solution.

L'algorithme du plus court chemin (*all pairs shortest paths*) présenté dans le Chapitre III permet de calculer le graphe minimal. Cet algorithme, de complexité cubique $O(n^3)$ en temps et en espace, est général car il ne tient pas compte de l'existence d'un ordre topologique entre les sommets. Or le graphe utilisé dans Madeus, étant topologiquement trié, admet une solution beaucoup plus efficace. C'est un algorithme basé sur ce tri que nous proposons ici.

Pour pouvoir appliquer cet algorithme, le graphe d'instants G est utilisé comme un graphe de distance G_d . La contrainte minimale entre deux instants i et j, introduite pour le graphe de distance G_d présenté dans la Chapitre III, est définie par l'intervalle [l, u], où –l représente la longueur du plus court chemin entre j et i, et u représente la longueur du plus court chemin de i à j.

L'algorithme présenté dans la Fig. 5.26 est une variante de l'algorithme de Bellman–Ford [77][53] basé sur l'ordre topologique du graphe. Pour deux sommets i et j du graphe tels que rang(i) < rang(j), l'ordre topologique permet de limiter la portée du calcul des plus courts chemins en propageant d'abord le calcul du sommet i en direction du

sommet j. Cela permet de définir la borne de durée minimale entre i et j. On procède ensuite dans l'ordre inverse pour déterminer la borne de durée maximale.

```
(Entier, Entier) ←
          Plus_court_chemin(G_Ch = (I_Ch, A_Ch, E_Ch), i, j)
{
1. Pour chaque x ∈ I_Ch {
2.   x.Distance = ∞
3. }
4. i.Distance = 0;
5. Pour chaque y ∈ I_Ch | 0 ≤ rang(i) ≤ rang(y) pris
      dans l'ordre topologique croissant
6.    Pour chaque chaîne C = (x, y) ∈ A_Ch et D_xy = [l, u]
7.      y.Distance = min(y.Distance, x.Distance +u)
8. Pour chaque x ∈ I_Ch | 0 ≤ rang(x) ≤ rang(j) pris
   dans l'ordre topologique décroissant
9.    Pour chaque chaîne C = (x, y) ∈ A_Ch et D_xy = [l, u]
10.     x.Distance = min(x.Distance, y.Distance − l)
11. retourne (i.Distance, j.Distance)
}
```

Fig. 5.26 : Algorithme de calcul du plus court chemin

Pour calculer le graphe minimal correspondant à un élément composé du document, il suffit d'appliquer l'algorithme deux fois : une fois entre son instant de début (sommet source) et son instant de fin (sommet puit), et une nouvelle fois entre son instant de fin et son instant de début. Cela permet en plus de déterminer les bornes de durée de toutes les chaînes qu'il contient. L'exemple de la Fig. 5.27 reprend un état cohérent du graphe présenté en V.3.2.3. et qui correspond dans ce cas à un élément composé X. L'application de l'algorithme sur ce graphe permet d'aboutir au graphe minimal représenté en Fig. 5.27.

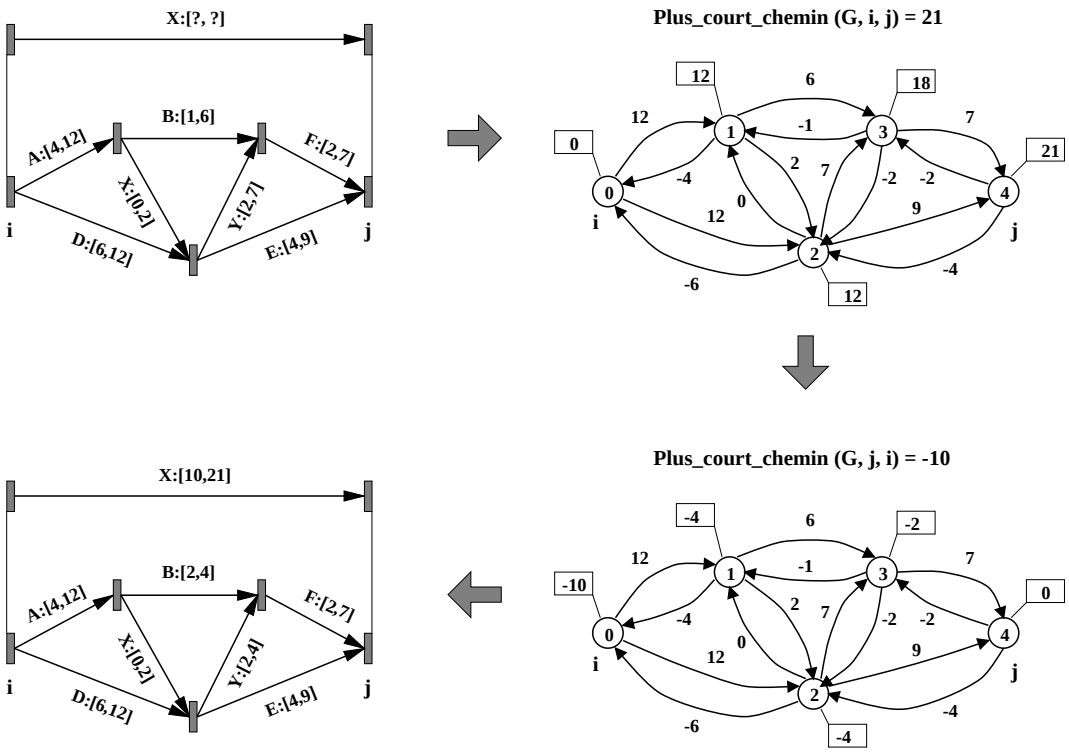


Fig. 5.27 : Calcul d'un graphe minimal

La complexité de cet algorithme est linéaire par rapport au nombre de sommets n et d'arcs e contenus dans le graphe : O(n+e). Il est donc beaucoup plus rapide que l'algorithme *all pairs shortest paths* présenté dans le Chapitre III ainsi que ses variantes comme [53]. Donc, si on considère le coût de la vérification de cohérence quantitative cumulé au coût de calcul du graphe minimal, nous obtenons une complexité totale linéaire en fonction du nombre de sommets n et du nombre d'arcs e : 2 * O(n+e) + O(n k^2) ≅ O(n+e).

V.3.3.2 Formatage temporel

Dans Madeus, l'opération de formatage temporel consiste à appliquer un algorithme à deux phases pour calculer deux solutions pour un document (à partir du début du graphe puis à partir de sa fin). On peut ainsi mieux répartir les contraintes qui pèsent sur les différentes chaînes et éviter que certains éléments multimédia ne subissent des déviations trop importantes par rapport à leurs valeurs optimales.

Chaque phase de l'algorithme consiste à dater les différents sommets du graphe en partant de son sommet source, respectivement puits. Dans la première phase, cette datation est effectuée dans l'ordre topologique croissant (lignes 1−22 de l'algorithme de la

Fig. 5.29) et permet de définir une valeur de durée (C.durée1) pour chaque chaîne. La deuxième phase est effectuée dans l'ordre décroissant et permet de définir la valeur C.durée2. La Fig. 5.28 représente l'application de cet algorithme à l'exemple de la Fig. 5.27.

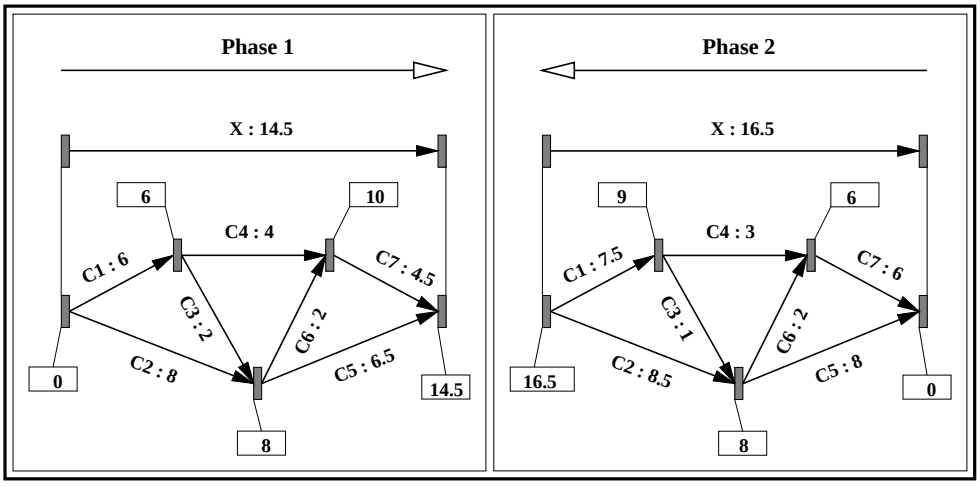


Fig. 5.28 : Formatage temporel d'un scénario

L'algorithme prend en compte les durées préférables des différentes chaînes (lignes 8, 12 et 16). De plus, à chaque étape du calcul de la date d'un sommet du graphe, l'algorithme s'assure que la date retenue respecte bien les contraintes de chacune des chaînes (lignes 13, 14, et 17).

La solution finale est alors obtenue par la valeur moyenne des valeurs issues du calcul de chaque phase. Cette nouvelle moyenne constitue elle–même une solution car les contraintes du graphe forment un système linéaire.

```
 Formater (G_Ch=(I_Ch, A_Ch, E_Ch))
  {
1. Liste = I − { source };
2. Distance(source) = 0;
3. Tant que Liste ≠ ∅ {
4.  n ← Sommet_de_rang_Minimum(Liste);
5.  Liste ← Liste − { n }
6.  Si (Degré_Entrant(n) == 1) {
7.    Pour la chaîne C | C+ = n
8.      Distance(n) = Distance(C−) + C.optimale;
```

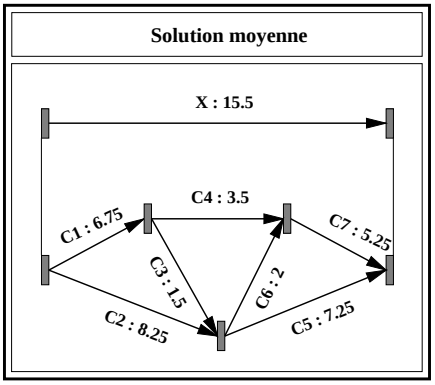
```
 9.  } sinon {
10.    somme = 0; min_Max = ∞ ; max_Min = 0 ;
11.    Pour chaque chaîne C =[l, u] | C+ = n {
12.      somme = somme + Distance(C−) + C.optimale;
13.      min_Max = min( min_Max, Distance(C−) + u);
14.      max_Min = max( max_Min, Distance(C−) + l);
15.    }
16.    Moyenne = somme / Degré_Entrant (n);
17     Moyenne = min ( min_Max, max (Max_min, Moyenne));
18.    Pour chaque chaîne C | C+ = n {
19.     C.durée = Moyenne − Distance(C−);
20.   }
21.   Distance (n) = Moyenne;
22.  }
23. /* calcul de C.durée2 par datation dans l'ordre topologique
       décroissant */
24. ...

25..  Pour chaque chaîne C {
26.     C.effective = (C.durée1 + C.durée2 )/2
27.  }
   }
```

Fig. 5.29 : Algorithme de formatage temporel

L'algorithme de la Fig. 5.29 permet le formatage d'un document en temps linéaire : O(n).

Les valeurs obtenues sont ensuite propagées à l'échelle des chaînes aux éléments multimédia qu'elles contiennent. Ce calcul de valeurs tient compte du type de média de chaque élément. Par exemple, si on considère une chaîne formée simplement par un élément vidéo suivi d'un délai temporel, on commence en priorité par le calcul de la durée de l'élément vidéo aux dépens de l'élément délai. Dans Madeus, cette notion est étendue à l'ensemble des média en attribuant à chacun un ordre de priorités. Ces priorités sont définies dans l'ordre suivant : Audio, Vidéo, Animations, Texte, Délais. On privilégie ainsi les éléments continus, ensuite les éléments discrets et enfin les éléments vides de contenu :

les délais. Après cette phase, le document est sous sa forme exécutable et l'information portée par le graphe peut directement être exploitée dans le processus de présentation.



Chaînes	Valeur optimale	Phase 1	Phase 2	Solution moyenne
C1	6	6	7.5	6.75
C2	10	8	8.5	8.25
C3	1	2	1	1.5
C4	3	4	3	3.5
C5	5	6.5	8	7.25
C6	4	2	2	2
C7	6	4.5	6	5.25

Fig. 5.30 : Document de l'exemple de la Fig. 5.28 formaté

L'établissement d'une politique automatique de répartition des contraintes entre les différents média reste une opération délicate [14][41][55]. Car elle peut engendrer des situations qui ne correspondent pas nécessairement aux souhaits de l'auteur. Ce problème n'est cependant pas nouveau dans le domaine des documents; il peut être comparé au comportement des formateurs spatiaux de texte comme LaTex [61] et Thot [88].

Signalons finalement qu'une partie du formatage est effectué dynamiquement, à l'exécution. Car la prise en compte des durées effectives des intervalles incontrôlable nécessite **un processus réactif** de réajustement du graphe (cf. V.3.2.4). Ceci ne remet pas en cause l'opération de formatage, car c'est à partir des durées calculées durant cette phase que le réajustement du graphe pendant la phase de présentation est réalisée (voir le section V.3.2).

V.3.4 Évaluation

Nous présentons maintenant quelques mesures de performances du gestionnaire temporel de Madeus (Fig. 5.31). Ces mesures sont effectuées à partir d'un ensemble de documents réels qui ont été produits et présentés avec l'outil. Ces mesures montrent que l'outil permet d'obtenir de bonnes performances dans le processus de construction et de vérification du graphe. Pour chaque document, la mesure porte sur le temps *CPU* pris pour construire la forme exécutable d'un document. Cela correspond, lors de l'utilisation de Madeus en tant qu'outil de présentation, à la succession des opérations de chargement, de vérification et de formatage. C'est ce temps que perçoit l'utilisateur lorsqu'il active un document, d'où l'importance de ces mesures.

Document	UR	Inria	Scène	Hubble	Ancres
Nombre d'éléments	167	98	42	21	11
Nombre de relations	137	93	36	20	8
Nombre de sommets	189	110	52	33	12
Nombre d'arcs	271	155	80	43	18
Nombre de chaînes	137	70	57	23	13
Niveaux hiérarchiques	6	4	3	2	2
Temp CPU (ms)	**114.52**	**69.54**	**30.87**	**15.46**	**7.6**

Fig. 5.31 : Temps de traitement des documents Madeus sur Sun Ultra Sparc

Les algorithmes sont écrits en langage C et les mesures sont effectuées sur une machine Sun−Ultra−Sparc (sous le système Unix Solaris 2.5). Chaque mesure a été effectuée 500 fois au cours du fonctionnement normal de l'application et du système. Ces chiffres témoignent d'une dépendance quasi−linéaire entre la taille du document en terme de nombre d'éléments et son temps de traitement. Ces dépendances s'expliquent surtout par le fait que le nombre de chaînes parallèles à un instant donné de la présentation n'excède pas la dizaine.

V.4 Conclusion

Dans ce chapitre, nous avons présenté le modèle temporel mis en œuvre dans Madeus. Ce modèle fondé sur les contraintes permet de manipuler de façon flexible la dimension temporelle des documents multimédia. De plus, les modifications du scénario sont prises en compte de façon incrémentale, ce qui permet d'avoir à tout instant (au formatage près) une forme cohérente et donc exécutable du document.

Le graphe extrait de la représentation peut ainsi être directement utilisé pour ordonnancer le document. C'est cette phase que nous décrivons dans le chapitre suivant.

Chapitre VI

Système de présentation de Madeus

VI.1 Introduction

Le système de présentation est une partie importante de l'architecture générale de l'application d'édition de documents multimédia Madeus. Ce système prend en charge la représentation interne d'un document qui est produite par le gestionnaire temporel. Cette représentation, structurée sous forme d'une hiérarchie d'**objets**(2) et d'un graphe d'exécution, est utilisée pour restituer dynamiquement à l'utilisateur le contenu du document à travers les différents dispositifs de la machine, comme l'écran graphique et le haut–parleur.

La construction d'un tel système de présentation est une opération particulièrement délicate. En effet, la disposition temporelle des objets spécifiée par l'auteur n'est qu'une vue idéale conçue par rapport à une forme du temps absolue. Or, cette vue implique la possibilité de réaliser plusieurs actions en un temps nul (hypothèse de synchronisme des événements [9]), ce qui n'est pas réaliste sur des plates–formes de traitement séquentiel :

- Les différents traitements des données multimédia, comme la décompression ou les transformations graphiques, nécessitent souvent des ressources système importantes. La charge imposée provoque des déviations temporelles qui restent inévitables dans les systèmes d'exploitation actuels comme *UNIX* ou *Windows* [99]. Ces systèmes n'offrent aucune garantie sur les temps de traitement et, pour cette raison, ils sont qualifiés de *best effort* : ils font pour le mieux.
- L'accès distant aux documents, rendu nécessaire par le volume très important de données comme la vidéo, l'audio et les images n'est pas instantané. Ce type d'accès fait intervenir un nombre élevé d'opérations d'entrée/sortie et provoque des délais supplémentaires dûs à la latence des réseaux qui s'ajoutent à ceux du

(2) Dans ce chapitre, nous désignerons par le terme objet les éléments d'un document multimédia manipulés dans le contexte du système de présentation.

système. La progression de la présentation devient donc tributaire du débit du flot de données qui conditionne la durée finale des objets multimédia.

La démarche qui a été adoptée dans la conception du système de présentation de Madeus consiste à limiter autant que possible l'adjonction de nouvelles sources de déviations temporelles. Les traitements de présentation appliqués aux objets du document ont été élaborés avec soin afin de satisfaire au mieux la spécification idéale de l'auteur.

Ce chapitre est organisé en trois parties. La première est dédiée à la présentation de notre approche de conception. Elle présente les principales composantes autour desquelles est structuré notre système de présentation. La deuxième partie présente la mise en œuvre de ces composantes : l'ordonnancement temporel d'une présentation, les mécanismes de synchronisation qui ont été développés (section VI.3) et le gestionnaire de présentation réalisé pour la restitution d'une présentation (section VI.4). La dernière partie tire un premier bilan de l'utilisation du système.

VI.2 Architecture du système de présentation

Le système Madeus manipule des documents sources écrits en *ASCII* et utilisant un marquage descriptif, conformément au format pivot décrit dans le chapitre IV. Lors du chargement d'un document, cette représentation est prise en charge par le gestionnaire temporel. Au moment où l'utilisateur souhaite visualiser le document, le gestionnaire produit le graphe d'exécution (graphe de contraintes résultat du formatage statique). Ce graphe, ainsi que l'arbre abstrait, rendent compte des différentes dimensions du document.

La présentation d'un document résulte de la combinaison de traitements réalisés à partir de l'arbre abstrait et du graphe d'exécution. Ces traitements permettent la présentation du document. Il s'agit principalement d'assurer les fonctions suivantes :

- La gestion de la synchronisation temporelle intra– et inter–objets et la prise en compte de l'indéterminisme.
- La gestion des interactions avec l'utilisateur.
- L'accès aux données multimédia sur le disque ou sur un serveur distant.
- La gestion de l'exécution d'une présentation.
- La gestion de la restitution sur les périphériques de la machine.

Pour garantir l'extensibilité et pour faciliter la portabilité du système sur différentes plate–formes, ces fonctions ont été mises en œuvre dans Madeus sous forme d'un ensemble de modules.

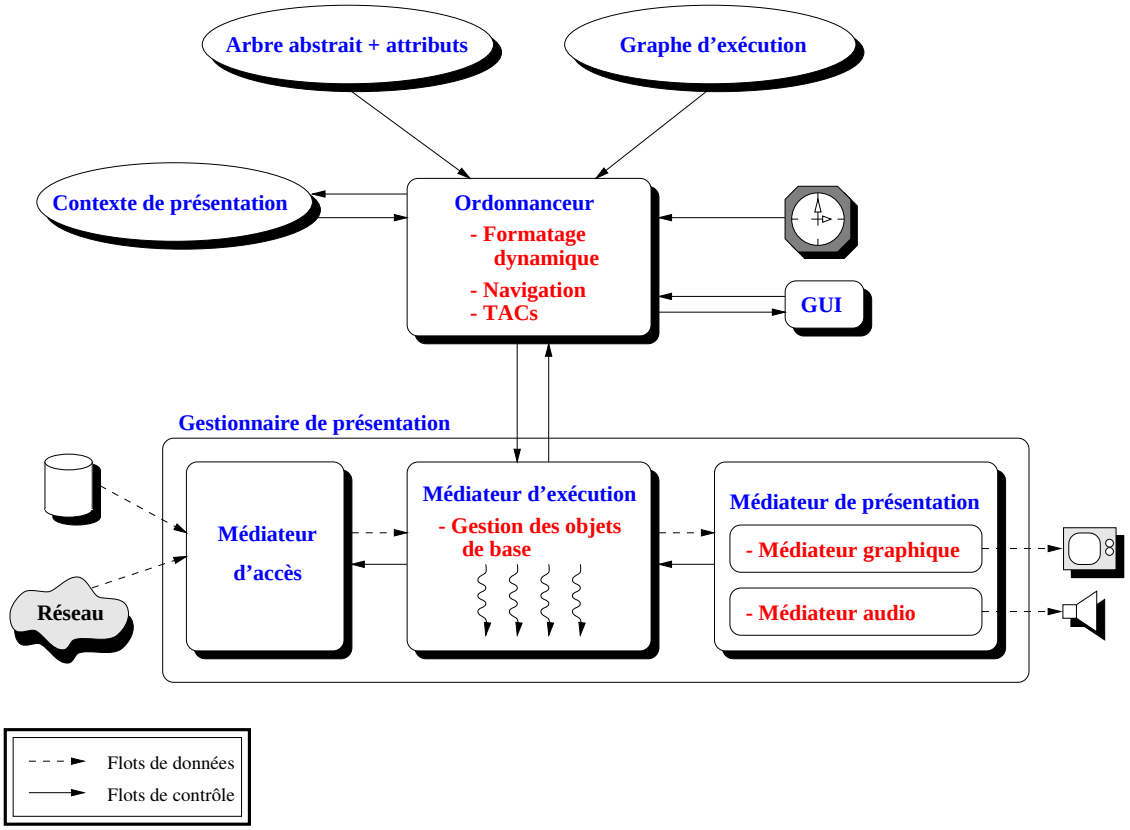


Fig. 6.1 : Architecture du système de présentation de Madeus

VI.2.1 Organisation du système

Le système de présentation est organisé autour de trois modules principaux. Chaque module est dédié au traitement d'un aspect particulier de la présentation et interagit dynamiquement avec les autres modules (cf. Fig. 6.1). Ces modules sont :

L'interface de présentation (GUI)

L'interface de présentation regroupe les moyens de visualisation et d'interaction offerts à l'utilisateur par l'intermédiaire de l'interface graphique. Elle lui permet, d'une part, d'agir dynamiquement au cours de la présentation à travers l'activation de liens ou de boutons d'interactions, et d'autre part, de contrôler la progression temporelle. Cette partie a été décrite dans la section IV.2.5.

L'ordonnanceur

L'ordonnanceur se situe au cœur du système de présentation multimédia. Après le chargement d'un document et la production du graphe d'exécution, le démarrage effectif est pris en charge par l'ordonnanceur. C'est ce module qui coordonne les

différentes activités de présentation du document et assure sa progression temporelle. Cette progression est réalisée par une **horloge** associée au document. L'horloge produit et reçoit les différents événements de présentation en s'appuyant sur deux représentations du temps :

- Le temps relatif au document et aux différents objets décrits par le graphe d'exécution.
- Le temps physique mesuré par rapport à l'horloge de la machine (le temps « réel »).

Chaque événement de l'horloge peut déclencher de nouvelles activités de présentation ou l'arrêt de celles qui sont en cours. Le rôle de l'ordonnanceur consiste à accorder dynamiquement ces deux formes de temps : c'est la supervision de la présentation.

Le gestionnaire de présentation

Le gestionnaire de présentation met en correspondance la progression temporelle du document et les opérations qui permettent de restituer son contenu à l'utilisateur. Alors qu'au niveau de l'ordonnanceur la progression temporelle est simplement perçue à travers des horloges (démarrage, terminaison, etc.), au niveau du gestionnaire elle correspond à des traitements spécifiques sur les média de base (affichage d'une image, animation d'un texte, etc.). Le gestionnaire de présentation permet aussi l'allocation des ressources comme les accès aux données, les entités d'exécution du système et les ressources graphiques et audio. Il est conçu de façon à renforcer l'indépendance de Madeus vis–à–vis de toute plate–forme spécifique.

VI.2.2 Déroulement d'une présentation

Les modules présentés ci–dessus effectuent des traitements qui se situent à différents niveaux de l'exécution et qui s'enchaînent au fur et à mesure du déroulement de la présentation : lancement de la présentation d'un document, d'un objet de base, lecture d'un échantillon de données, etc. La progression d'une présentation multimédia résulte donc des interactions entre les différents modules. On distingue alors deux types d'information qui circulent entre ces modules : les flots de données et les flots de contrôle.

Flots de données

Ces flots représentent des données qui circulent au niveau du gestionnaire de présentation. Ils sont de deux types :

- les flots entrants qui concernent les données acquises localement sur disque ou de façon distante à travers le réseau (voir le médiateur d'accès dans la Fig. 6.1),
- Les flots sortants qui concernent les données restituées sur les périphériques de sortie de la machine (écran, haut−parleur).

Flots de contrôle

Ces flots sont représentés au sein de l'application par un ensemble d'**événements**. Ces événements sont de trois types :

- Les *événements d'interaction* avec l'utilisateur. Ils concernent l'activation de boutons ou de liens hypermédia.
- Les *événements de présentation*. Ils concernent l'activation et la terminaison d'objets multimédia.
- Les *événements de contrôle* de la progression temporelle du document. Ils concernent, par exemple, l'arrêt de la présentation ou son redémarrage (*Temporal Access Control*)[73].

La présentation d'un document fait intervenir une combinaison de ces deux types de flots. Par exemple, l'opération d'arrêt d'une présentation *(flot de contrôle)* se propage à travers les différents modules jusqu'au médiateur d'accès *(flot de données)*. De la même façon, l'arrivée du dernier échantillon d'une vidéo *(flot de données)* déclenche l'événement de terminaison qui lui est associé *(flot de contrôle)*.

VI.3 Ordonnancement d'une présentation dans Madeus

L'ordonnancement d'une présentation consiste à mettre en correspondance l'image abstraite de la présentation (arbre abstrait et graphe d'exécution) et sa réalisation concrète au sein du système. Ce traitement regroupe l'ensemble des tâches suivantes :

- **La gestion des contextes de présentation** permet de maintenir, à chaque instant, l'état courant de la présentation d'un document : objets en cours, valeurs des attributs dynamiques, temps écoulé depuis le début de la présentation d'un objet ou depuis le début du document, etc. (cf. section VI.3.1).
- **La synchronisation inter−objets** permet d'une part de gérer l'enchaînement des événements de présentation, et d'autre part d'ajuster le graphe d'exécution pour prendre en compte l'indéterminisme (cf. section VI.3.2).
- **La synchronisation intra−objets** permet de cadencer la présentation au moyen des événements de contrôle (TAC) à travers des horloges (cf. section VI.3.3).

- **La gestion de la navigation** met en œuvre la navigation au moyen des liens hypermédia temporisés et gère l'historique. La navigation nécessite des traitements qui portent sur la totalité d'un document (cf. section VI.3.4).

L'ordonnancement d'une présentation applique ces différents traitements en s'appuyant sur la structure du graphe. L'ordonnancement est donc constitué d'une boucle fermée qui produit et récupère des événements et déclenche les traitements correspondants. À tout instant, ces traitements dépendent de l'état courant de la présentation du document.

VI.3.1 Contexte d'une présentation

L'état d'un document multimédia en cours de présentation est défini par la valeur des attributs de présentation de chaque objet à chaque instant. Cet état comprend de plus des informations qui identifient la progression temporelle de la présentation.

On définit sous le terme de **coupure temporelle** l'image de la présentation à un instant donné. Elle correspond à l'état d'avancement de tous les objets à cet instant. À partir du graphe d'exécution, on peut déterminer une coupure temporelle pour chaque instant de début ou de fin d'un objet (voir la Fig. 6.2). Ces coupures sont utilisées pour la reconstitution du contexte de la présentation lors de la navigation hypermédia (changements de contexte).

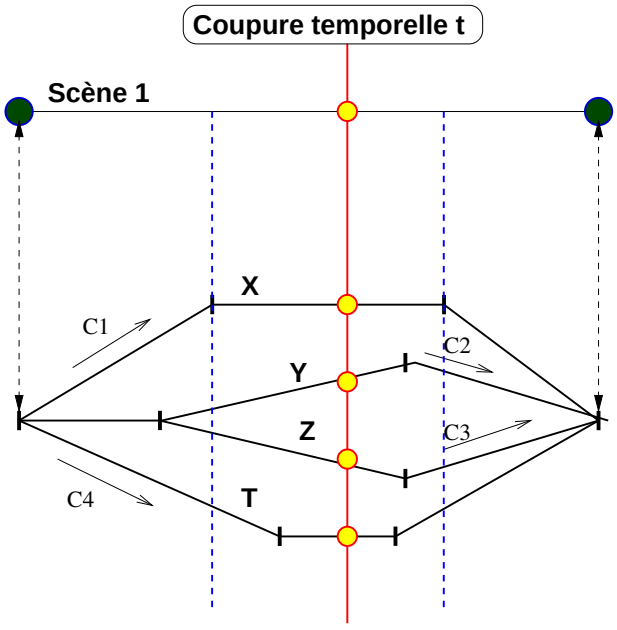


Fig. 6.2 : Coupure temporelle d'une présentation multimédia

Ainsi, à chaque objet du document, qu'il soit composé ou de base, est associé un contexte permettant d'indiquer l'état de sa présentation. Cet état est défini par les informations suivantes :

- Une *horloge* qui mesure sa progression temporelle. Ces horloges sont calculées par rapport aux horloges des objets englobants selon la structure logique et par rapport à celle de la machine.
- Des pointeurs vers les nœuds du graphe qui correspondent aux instants de début et de fin de l'objet. À partir de ces pointeurs, l'ordonnanceur peut retrouver les actions de synchronisation à effectuer lors de la terminaison de chaque objet.
- Un ensemble d'attributs de présentation, comme la taille, la position géométrique, les attributs dynamiques, etc.

Le lien entre la structure temporelle d'un document et les attributs de présentation est maintenu dans une structure de données appelée **contexte de présentation**. Le contexte de présentation de Madeus est une structure **hiérarchique** qui associe, à chaque objet actif composé (cf. Fig. 6.3), toutes les activités en cours qu'il contient. Ces activités sont décrites par la liste des chaînes actives du graphe d'exécution. Ces chaînes pointent sur l'objet en cours de présentation. Le contexte est mis à jour dynamiquement au rythme de la progression de la présentation, conformément à la topologie du graphe.

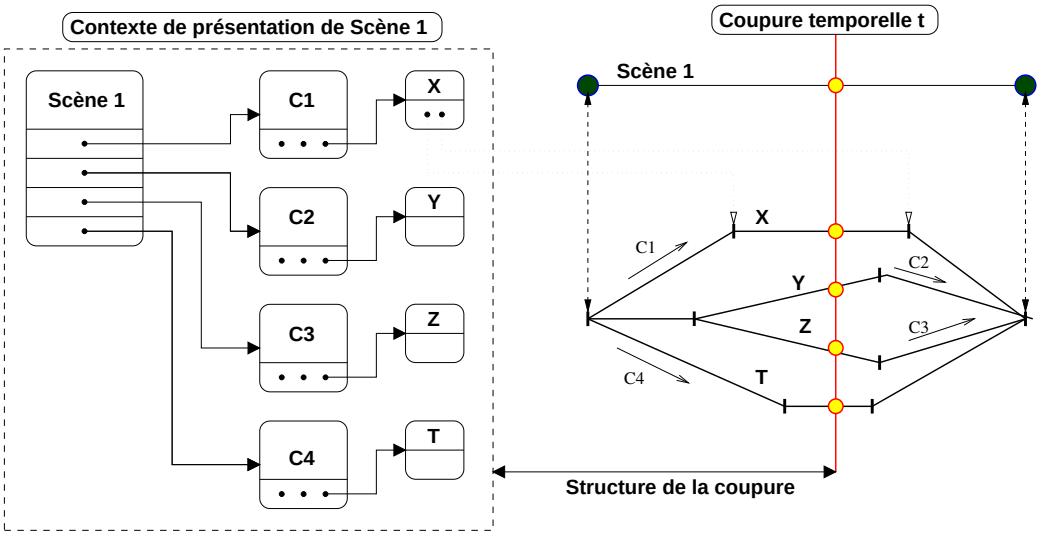


Fig. 6.3 : Contexte de présentation

Les valeurs des attributs de présentation sont également mises à jour dynamiquement. Ce processus de mise à jour permet d'assurer la transmission des différents attributs entre les objets composés, ainsi que l'application des attributs de style au niveau des objets de base (mouvement de fenêtre).

Par ailleurs, le contexte de présentation est la structure de base utilisée pour la synchronisation inter– et intra–objets.

VI.3.2 Synchronisation inter–objets

La synchronisation inter–objets est appliquée à chaque occurrence d'un événement de présentation. On distingue deux types d'événements :

- La terminaison d'un objet contrôlable : l'occurrence de cet événement entraîne soit la terminaison, soit l'activation d'autres objets. Ce traitement est réalisé par l'**algorithme de synchronisation** décrit ci–dessous (section VI.3.2.2).
- La terminaison d'un objet incontrôlable : l'occurrence de cet événement nécessite, dans un premier temps, d'adapter le graphe d'exécution à partir de la durée observée de l'objet. Ce traitement est réalisé par l'**algorithme de formatage dynamique** présenté section VI.3.2.3. Suite à cette opération, il est également nécessaire de gérer la terminaison ou l'activation d'autres objets. Cela revient à appliquer l'algorithme de synchronisation.

VI.3.2.1 Structures de synchronisation

La synchronisation inter–objets est réalisée au moyen des nœuds du graphe temporel. Ces nœuds contiennent des informations de synchronisation décrivant de façon précise les relations qui existent entre les différents événements. Ces relations indiquent à l'ordonnanceur les actions qu'il doit entreprendre à la réception d'un événement.

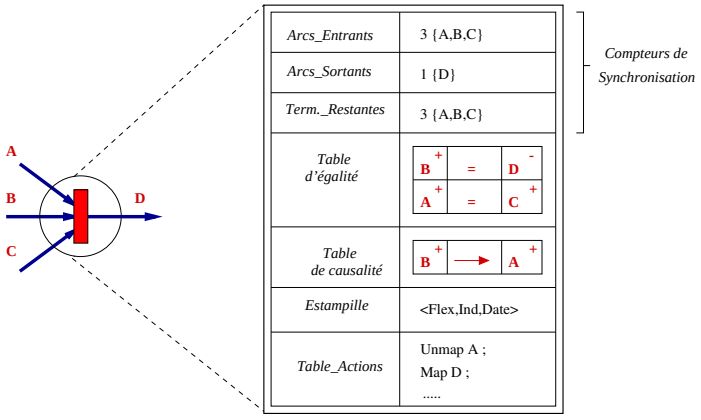


Fig. 6.4 : Structure de présentation d'un nœud temporel

La structure d'un nœud est décrite dans la Fig. 6.4. Elle comprend les informations suivantes :

- Le nombre d'arcs entrants dans le nœud.
- Le nombre d'arcs sortants du nœud.
- Un compteur de synchronisation (Terminaisons_Restantes) qui reflète, à chaque instant de la présentation, le nombre d'arcs entrants ayant terminé leur présentation. Ce compteur est initialisé au chargement du document et à chaque fois que le nœud est franchi.

Les conditions de franchissement d'un nœud sont définies à partir de deux tables. Ces tables indiquent la nature de la synchronisation entre les objets du document. Chacune joue un rôle de synchronisation particulier :

- **Table d'égalité :** cette table contient des paires de liens vers les instants de début ou de fin d'objets qui doivent coïncider temporellement.
- **Table de causalité :** cette table contient des règles de causalité décrites par des paires d'instants de fin d'objets (e1, e2). Chaque entrée de la table signifie que l'occurrence de l'événement e1 provoque l'événement e2.

La structure d'un nœud est complétée par une liste d'actions spatio–temporelles. Chaque action est composée d'un opérateur spatial décrivant la nature de l'opération à effectuer et d'un opérande. L'opérande est une référence interne à un objet simple ou composé. Parmi ces opérations, on trouve les actions *Map* et *Unmap* permettant de réaliser l'apparition ou la disparition d'un objet de base ou composé à l'écran. Ces actions sont réalisées dans l'ordre suivant : les actions *Unmap* d'abord, les actions *Map* ensuite, car les premières concernent des objets dont la présentation se termine et les secondes les objets dont la présentation commence par rapport à l'instant représenté par le nœud. On libère ainsi les espaces occupés sur l'écran graphique avant de les ré–allouer à d'autres objets. D'autres opérations peuvent être attachées à cette liste d'actions. En particulier, des scripts peuvent être activés au franchissement du nœud permettant ainsi la coordination du système de présentation avec des outils externes.

Le graphe de la Fig. 6.4 représente un seul niveau d'une hiérarchie logique : chaque arc pouvant représenter un objet composé et donc un nouveau graphe de synchronisation (*hypergraphe*). Ainsi la synchronisation d'un document multimédia revient à la synchronisation de tous les objets qu'il contient.

VI.3.2.2 Algorithme de synchronisation

L'algorithme de la Fig. 6.5 utilise de façon directe les tables présentées dans la section précédente. Il met en œuvre la synchronisation globale du document à partir des différents événements de présentation. À chaque occurrence d'un événement, il applique un traitement qui consiste à propager l'effet de causalité entre les événements de fin, contenus dans

la Table_Causalité. Par exemple, si l'événement e1 est associé à l'événement e2, l'occurrence de e1 provoque l'occurrence de e2. Si à son tour e2 est lié avec d'autres événements, ils sont provoqués de façon récursive (lignes 8–13). Le compteur Terminaisons_Restantes est alors décrémenté d'autant d'unités que d'événements produits. Dans les autres cas, l'ordonnanceur décrémente simplement le compteur Terminaisons_Restantes d'une seule unité : cas des événements définis dans la Table_Égalité (égalité entre instants, lignes 15–16).

Après cette phase, l'ordonnanceur vérifie la condition de franchissement d'un nœud. Elle correspond à une mise à zéro du compteur Terminaisons_Restantes (ligne 18). Dès qu'un nœud atteint cette valeur, l'ordonnanceur met à jour la liste des chaînes actives du contexte (ligne 19) et démarre les horloges des objets correspondants aux arcs sortant du nœud (lignes 20–22). Cette opération se traduit, au niveau du gestionnaire de présentation, par le lancement des activités de présentation correspondantes.

```
Algorithme : Ordonnanceur(événement : e)
{
1. si (e.type == Démarrer_Document)
2. Déclencher_Nœud(Document.début);
3. sinon /* traitement de la terminaison d'un objet */
4.  si (e.type == Terminaison_Objet) {
5.   pour (e.O.fin) {
6.   1) e.O.fin ∈ Table_Causalité(e.O.fin)
7.     e.O.fin.Terminaisons_Restantes --;
8.     e1 = e;
9.     tant que ∃(<e1,e2,Vrai> ∈ Table_Causalité(e.O.fin)){
10.     Termine(e2.Horloge);
11.     e2.O.fin.Terminaisons_Restantes --;
12.     <e1,e2,Faux> ← <e1,e2,Vrai>;
13.     e1 = e2;
14.    }
15.  2) e.O.fin ∈ Table_Égalité(e.O.fin)
16.    e.O.fin.Terminaisons_Restantes --;
17.  }
18.  si (e.O.fin.Terminaisons_Restantes == 0) alors {
19.    Mettre_à_jour_Contexte(e.O.fin);
20.    pour chaque Action de Table_Actions(e.O.fin)
21.      Gestionnaire de Présentation (
22.      Table_Actions[i].Opération(Table_Actions[i].Opérande));
23.    Déclencher_Nœud(e.O.fin);
24.  }
```

```
25. }
}
/* Activation des arcs sortants d'un nœud */
Fonction Déclencher_Nœud(Nœud : n) {
         Pour i=1 à n.Arcs_Sortants
            Démarre_Horloge(n.Objet_Sortant[i]);
}
```

Fig. 6.5 : Algorithme d'ordonnancement de Madeus

L'interface entre l'ordonnanceur et le gestionnaire de présentation se fait, soit directement par les appels de fonctions spatio–temporelles (Gestionnaire de Présentation), soit indirectement par la modification des valeurs d'horloges (Termine_Horloge) de l'algorithme. Cette technique de synchronisation vise à séparer les fonctions d'ordonnancement temporel des activités de présentation réalisées par le gestionnaire de présentation.

VI.3.2.3 Algorithme de formatage dynamique

La synchronisation présentée dans la section précédente prend en compte uniquement l'aspect déterministe et donc prédit des objets multimédia : les objets sont lancés avec les vitesses élaborées lors du formatage statique. Or, comme nous l'avons indiqué dans le modèle temporel, la prise en compte du comportement indéterministe est une nécessité absolue. Cette nécessité est d'autant plus grande que les systèmes d'exploitation actuels [23], les protocoles réseau et la nature de certains média sont autant de sources d'indéterminisme qu'il faut prendre en compte.

Dans Madeus, l'approche qui a été retenue est en accord avec la vérification de la contrôlabilité fournie à l'auteur (phase d'observation et phase de recouvrement). On peut relever deux types d'observations de l'indéterminisme lors d'une présentation :

- Les **valeurs effectives** des intervalles de type incontrôlable qui sont prises en compte dès la phase d'édition.
- Les **exceptions** qui correspondent à des déviations temporelles imprévues. Elles représentent les cas d'erreurs.

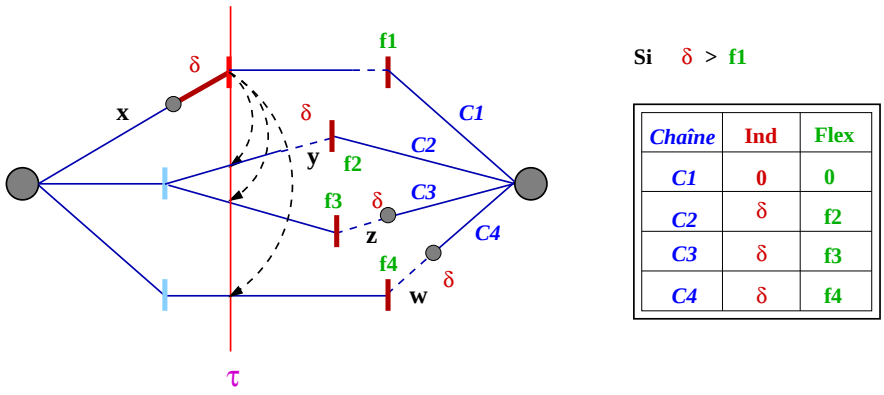
La phase d'observation permet de relever les valeurs des intervalles incontrôlables. Ces valeurs sont alors utilisées pour ajuster le scénario afin de respecter les contraintes temporelles décrites dans le graphe d'exécution. Cet ajustement est réalisé au moyen de la flexibilité contenue dans les chaînes. Il consiste à adapter les durées nominales de certains objets.

```
Algorithme : Formateur_Dynamique(événement : e)
{
1  /* Instantiation d'un intervalle incontrôlable */
2. Indéterminisme=e.O.Horloge.Valeur − e.O.Horloge.Lower_Bound;
3. si e.O.Chaîne.Flex > Indéterminisme
4.    Formater_Chaîne (e.O.Chaîne, (−1) * Indéterminisme);
5. sinon
6.  /* Compensation sur les chaînes concurrentes */
7.  pour toutes les Chaînes actives Ch ∈ Contexte(e.O.englobant)
8.    Formater_Chaîne (Ch, Indéterminisme);
}
Fonction : Formater_Chaîne (Chaîne : Ch, Entier : Ind)
{
9.  Ch.Flex = Ch.Flex − |Ind|;
10. Pour tout objet O ∈ Ch | Doc.Horloge.Valeur < O.fin
11.   si O.type == discret {
12.      O.fin = O.fin + Ind;
13.      fin de la fonction;
14.   } sinon
15.      si Doc.Horloge.Valeur < O.début {
16.         si O.Flex ≥ Ind {
17.            O.Horloge.Durée = O.Horloge.Durée + Ind;
18.            O.Flex = O.Flex − |Ind|;
19.            fin de la fonction;
20.         }
21.         sinon {
22.            O.Horloge.Durée = O.Horloge.Durée +
23.                              Signe(Ind) * O.Flex;
24.            Ind = Ind − Signe(Ind) * O.Flex;
25.            O.Flex = 0;
26.         }
27.      }
}
```

Fig. 6.6 : Algorithme de formatage dynamique de Madeus

L'algorithme de la Fig. 6.6 met en œuvre le principe d'observation et de recouvrement des intervalles incontrôlables [64]. Lorsque l'événement de fin d'un objet incontrôlable a lieu, le montant d'indéterminisme est d'abord calculé (ligne 2). Ensuite, l'ajustement du graphe est pris en compte prioritairement sur la chaîne de cet intervalle (ligne 4). Si la

flexibilité de la chaîne, depuis cet instant jusqu'à sa fin, est suffisante, l'indéterminisme est compensé en effectuant le re–formatage cette chaîne. Cette opération consiste à rétrécir la durée de la chaîne du montant d'indéterminisme observé. Dans le cas contraire (lignes 7 et 8), les chaînes concurrentes sont rallongées afin de permettre à la chaîne en question de rattraper ce délai. Par exemple, dans la Fig. 6.7, pour l'indéterminisme δ, observé sur l'intervalle x de la chaîne C1, le formatage de la chaîne n'a pu être effectué car sa flexibilité f1 est inférieure à δ. Les chaînes concurrentes C2, C3 et C4 sont alors reformatées pour être rallongées de ce même montant δ.



Chaîne	Ind	Flex
C1	0	0
C2	δ	f2
C3	δ	f3
C4	δ	f4

Fig. 6.7 : Configuration des chaînes concurrentes à l'instant τ

L'avantage de cette technique est de permettre de ré–aligner temporellement les chaînes les unes par rapport aux autres de façon non bloquante et locale. Ceci est effectué grâce au contexte de présentation qui permet, d'avoir, à tout instant, une vision globale de la progression du document. Il est ainsi possible de respecter les contraintes de coïncidence portées par les nœuds extrémités des chaînes.

Par ailleurs, le principe du formateur dynamique peut être employé pour gérer également les exceptions. Il suffit pour cela d'appliquer l'algorithme afin d'imposer le ré–alignement temporel des chaînes. Dans ce cas, on risque de ne plus respecter, momentanément, une partie des contraintes spécifiées par l'auteur.

VI.3.3 Synchronisation intra–objets

La présentation d'un objet multimédia est composée d'un ensemble d'actions élémentaires de grain variable (affichage d'une image de vidéo, remplissage du tampon interne de la carte son, décompression d'un échantillon, etc.). Dans Madeus, le contrôle de ces actions de présentation est réalisé à partir d'un mécanisme d'horloges qui masque le traitement spécifique de chaque objet [92]. Par exemple, l'arrêt de l'horloge d'un document correspond à l'arrêt de la présentation de tous les objets qu'il contient. Chaque

horloge mesure, tout au long de la durée de la présentation d'un objet, sa progression temporelle. Cette progression de l'horloge est manipulée par l'ordonnanceur afin de réaliser deux types de fonctions :

1. Offrir à l'utilisateur des fonctionnalités (TAC) de contrôle d'une présentation au moyen des actions Pause, Démarre, Redémarre, etc., fournies par l'interface graphique de Madeus.
2. Propager les tics réguliers de l'horloge au sein du gestionnaire de présentation. Ceci permet d'apparier les horloges avec les actions de présentation correspondantes au niveau de l'implantation des objets de base.

Types d'horloges

On distingue deux types d'horloges (Fig. 6.9) selon le type de média et leurs implantations dans le gestionnaire de présentation :

- **Horloges externes :** ces horloges correspondent aux objets dont les actions de présentation internes ne sont pas à la portée de l'ordonnanceur, comme les processus externes et les objets multimédia programmés « *applets* ». Après leur lancement, ces horloges produisent des événements qui indiquent simplement leur fin ou le dépassement de leur borne de durée maximale (exception). Elles regroupent tous les objets qui sont des sources potentielles d'indéterminisme.
- **Horloges internes :** ces horloges correspondent aux objets dont la synchronisation interne est entièrement contrôlée par l'ordonnanceur. Ce sont les objets multimédia discrets dont la présentation se traduit, soit par des actions portées par les nœuds du graphe (*Map*, *Unmap*), soit par les tics réguliers de l'horloge gérée par l'ordonnanceur (mouvement à l'écran). Parmi ces objets, on trouve les objets de type texte et image fixe.

Opérations sur les horloges

Les horloges du document et des différents objets sont manipulées à travers l'ensemble des opérations suivantes :

Démarre(Horloge) : démarre l'horloge interne d'un objet multimédia depuis son début.

Pause(Horloge) : suspend l'horloge interne d'un objet.

Redémarre(Horloge) : relance la progression de l'horloge d'un objet qui a été suspendue par Pause.

Change_Vitesse(Horloge, Vitesse) : modifie la cadence d'une horloge.

Termine(Horloge) a pour effet d'interrompre la progression de l'horloge.

Tic_un_Quantum(Horloge) : fait progresser l'horloge d'un quantum de temps (un tic).

Fin(Horloge) : indique si l'horloge a atteint sa durée d'échéance.

La gestion de l'horloge (voir l'algorithme de la Fig. 6.8) consiste à propager les tics (allocation de quantums) aux activités parallèles d'un objet composé (lignes 1–4). Si l'une des horloges atteint sa durée d'échéance, l'ordonnanceur est avisé pour effectuer la synchronisation inter–objets (lignes 5–6). Si l'utilisateur modifie la progression de la présentation (TAC) au niveau de l'interface utilisateur (cf. Fig. 6.9), il suffit, dans ce cas, de mettre à jour l'horloge globale du document. Cette modification se traduit par la propagation des mises à jour de chacune des horloges des objets de base actifs dans le contexte (lignes 8–9). S'il reste encore des horloges à faire progresser (ligne 10–11), l'algorithme réarme un chien de garde du système d'exploitation (*timer* d'UNIX) qui le réactive de façon régulière.

```
Algorithme : Gestion_Horloge(événement : e)
{
1.  /* attribution des quantums de temps */
2.  si (e.type == Fin_Quantum)
3.        pour tout objet O ∈ Document.Context tel que
          Fin(O.Horloge) == Faux et O.Horloge.interne {
4.           Tic_un_Quantum(O.Horloge);
5            si Fin(e.O.Horloge)
6              Ordonnanceur(e);
7   }
8.  si (e.type == TAC)
9.      e.Opération(Document.Horloge);
10. si ∃ (O ∈ Document.Context)|Fin(O.Horloge) == Faux
11.    Système_Arme_Chien_de_Garde(Quantum_Madeus);
}
```

Fig. 6.8 : Algorithme de gestion des horloges

Enfin, pour les objets ayant des horloges externes correspondant à des intervalles incontrôlables, les opérations liées à la modification de leur vitesse d'horloge n'ont pas de sens. Dans ce cas, l'ordonnanceur ignore simplement ces horloges en prenant en compte leurs dates d'échéance au plus tôt.

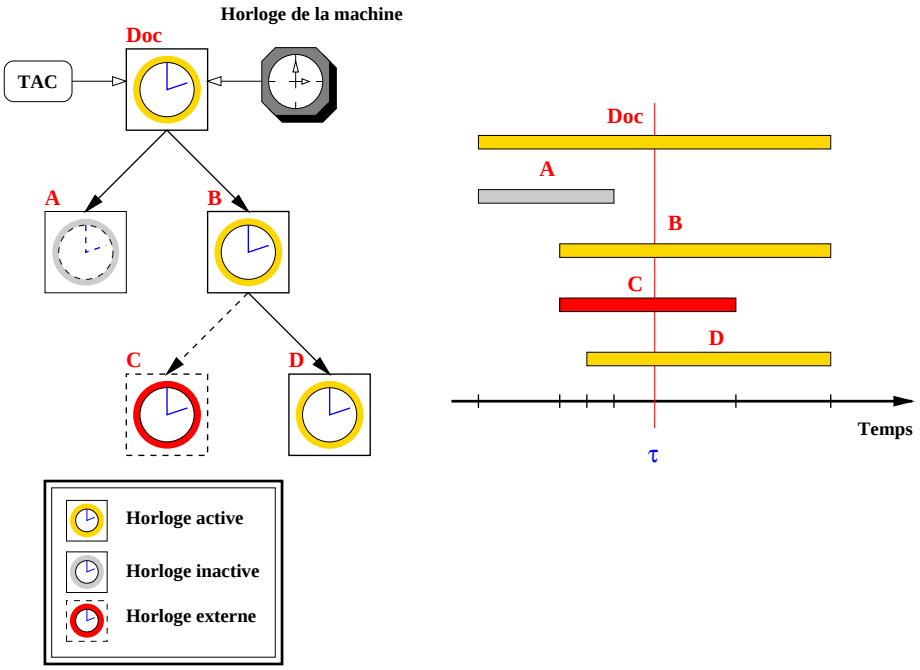


Fig. 6.9 : Hiérarchie des horloges

VI.3.4 Gestion de la navigation

Les liens sont des objets particuliers qui permettent à l'utilisateur de naviguer librement dans l'espace intra– ou inter–document. Dans les documents hypertextes statiques, les liens supportent l'interactivité dans la mesure où leur activation provoque une rupture dans la lecture séquentielle. Dans cette section, on s'intéresse plus particulièrement à l'influence de la structure temporelle sur la gestion et la mise en œuvre des liens. Une discussion beaucoup plus complète sur la nature des liens, leur typage, leur représentation, et leur mise en œuvre au sein de la structure du document statique est présentée dans [89][42].

On distingue dans Madeus deux types de liens : les liens d'**inclusion** et les liens de **référence**. Le traitement de l'activation d'un lien est effectué en fonction de son type. Ce traitement est réalisé selon le schéma suivant :

1. L'émission d'un événement d'activation de lien au niveau du gestionnaire de présentation. Celui–ci identifie l'objet référencé et demande à l'ordonnanceur d'exécuter la requête d'ouverture du lien.
2. Le traitement de cette requête entraîne soit la libération du contexte courant s'il s'agit d'un lien de référence, soit l'arrêt de la présentation en cours s'il s'agit d'une inclusion avec arrêt (cf. Fig. 6.10).

3. Le chargement de l'objet cible de la référence est alors amorcé. Cet objet est ensuite interprété et lancé à partir de son instant de début.

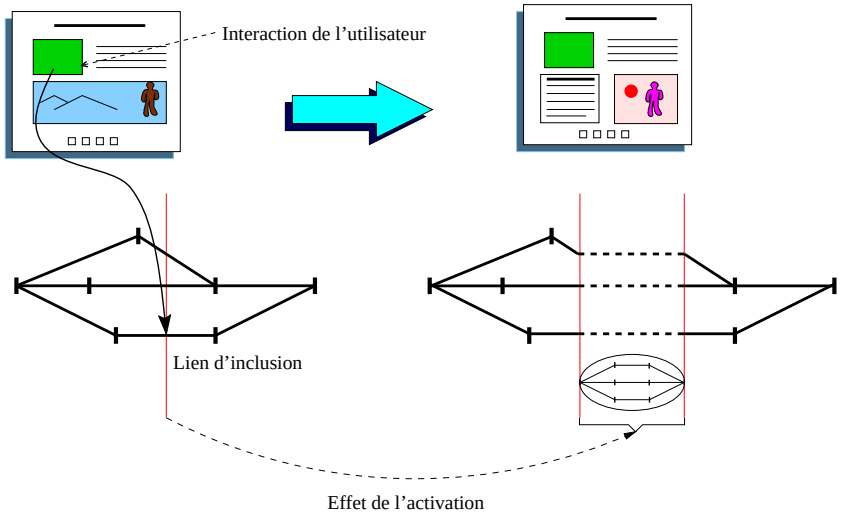


Fig. 6.10 : Effet de l'activation d'un lien d'inclusion sur le document

Dans le cas d'une référence de type inclusion, l'appel est assimilé à une intégration d'un nouveau graphe de présentation au sein du document (voir Fig. 6.10). Ce graphe est ordonnancé en exclusion mutuelle avec celui du document si l'attribut arrêt est positionné à vrai, en concurrence dans le cas contraire. Dans le premier cas la présentation du document est relancée dès la terminaison de l'inclusion et dans le second cas elle n'est pas affectée.

VI.4 Gestionnaire de présentation

Le gestionnaire de présentation effectue la médiation entre l'ordonnanceur et la représentation des objets dans l'environnement de présentation. Il est constitué de trois modules :

- **Médiateur d'accès** : Il associe à chaque objet de base un flot de données en entrée (cf. VI.4.3).
- **Médiateur d'exécution** : Il associe à chaque objet de base un flot d'exécution (cf. VI.4.4).
- **Médiateur de présentation** : Il gère les ressources de restitution sur l'écran graphique et le périphérique audio (cf. VI.4.5).

L'interaction entre l'ordonnanceur et le gestionnaire de présentation est réalisée à travers la représentation de l'exécution sous forme d'automates d'états finis [50]. Ces

automates permettent de faire correspondre aux opérations effectuées sur les horloges des traitements de plus bas niveau (des copies graphiques, des effets vidéo).

VI.4.1 Automate d'états finis

Chaque automate est composé d'un ensemble de nœuds et d'arcs. Les nœuds représentent des états de la présentation. Les arcs représentent des transitions entre ces états. Les arcs sont activés par des événements qui proviennent soit de l'ordonnanceur, soit du médiateur d'exécution (terminaison de processus externes, interactions utilisateur, etc.).

L'occurrence d'un événement se traduit soit par une transition de l'automate d'un état à un autre, soit par la modification des paramètres internes d'un objet (variables d'état).

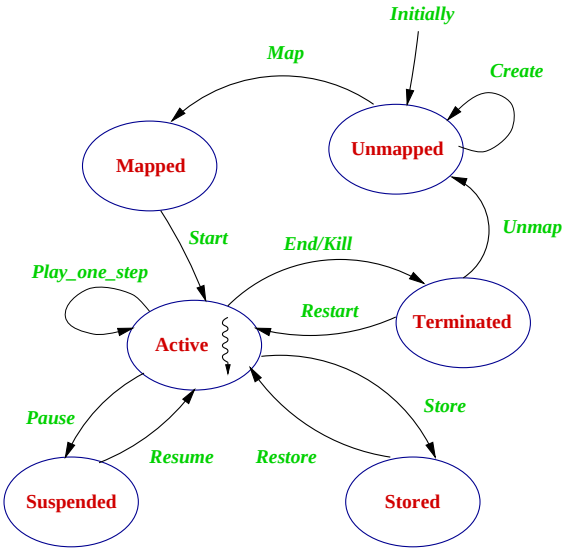


Fig. 6.11 : États abstraits d'un objet multimédia

Les principales actions modélisant une transition d'état d'un objet sont (voir Fig. 6.11) :

- **Create :** cette transition provoque l'initialisation des structures internes d'un objet et l'allocation de ses ressources de présentation.
- **Map/Unmap :** ces transitions provoquent respectivement l'affichage et l'effacement d'un objet graphique sur la fenêtre de présentation. Dans le cas des objets de type audio ces transitions sont confondues avec les transitions Start/End.
- **Start/Pause/Resume/Kill/End/Restart :** ces transitions provoquent respectivement le démarrage, l'arrêt, le redémarrage, la terminaison, la fin normale ou la relance d'un objet.

- **Play_One_Step :** cette transition provoque l'exécution d'un pas de la présentation d'un objet multimédia. Elle est provoquée par l'appel **Tic_un_Quantum** de l'ordonnanceur.
- **Store/Restore :** ces transitions provoquent la libération/ré−allocation des ressources d'un objet. Elles interviennent lors du suivi d'un lien hypermédia.

À chaque réception d'une demande de création *Create*, le médiateur met en place l'infrastructure permettant, pour chaque objet, d'exécuter sa présentation : c'est la phase de configuration d'un objet.

VI.4.2 Configuration des objets multimédia de base

La structure des objets au sein du système comprend les informations suivantes :

- Les périphériques (graphiques ou audio) auxquels les objets sont liés.
- Les flots de données qui constituent le contenu des objets de base.
- Les traitements associés à chaque objet de base au sein du système de présentation.

Les traitements liés à la présentation des objets multimédia dépendent de leur type. Pour les données texte et les interactions utilisateur, leur représentation dans le contexte de présentation est suffisante pour réaliser leur rendu à l'écran (voir l'objet A dans la Fig. 6.12). Par contre, les objets comme la vidéo, les images et l'audio sont stockés de façon externe (disque local ou serveur distant). Ces différents objets nécessitent une série de traitements avant d'aboutir à leur restitution finale sur les périphériques de la machine.

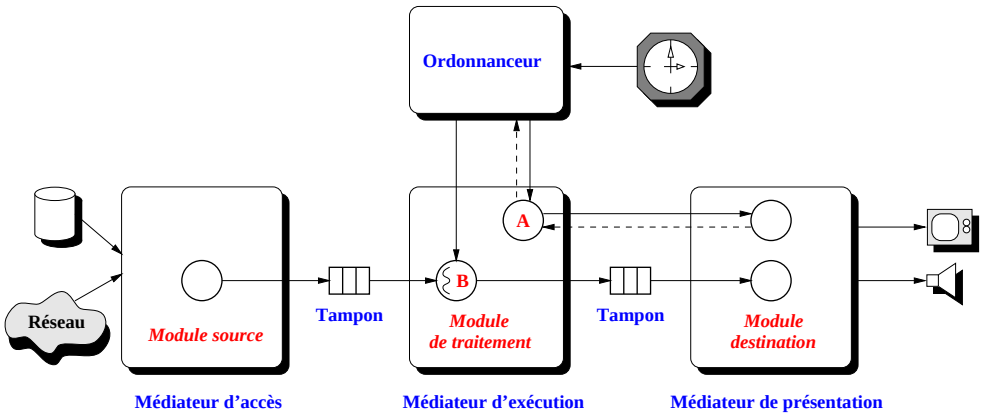


Fig. 6.12 : Configuration des objets multimédia de base

Le traitement de chaque objet de base est organisé en trois modules :

- **Module source :** il lit les données, par l'intermédiaire du médiateur d'accès, et les délivre au module de traitement.
- **Module de traitement :** il effectue les traitements spécifiques à un objet : la décompression, l'adaptation de la taille de l'image, la segmentation du texte en plusieurs lignes, etc.
- **Module destination :** il restitue les données produites par le module de traitement sur les périphériques correspondants.

Ces trois modules définissent **la configuration de traitement** pour chaque objet multimédia de base. Le transfert des données entre les différents modules est réalisé à travers des tampons selon le schéma de synchronisation producteur–consommateur [57].

VI.4.3 Médiateur d'accès

Il permet de réaliser l'accès effectif à l'information multimédia, qu'elle soit locale ou distante. Le chargement des objets suit le schéma suivant :

1. L'ouverture d'une connexion avec un serveur distant et la récupération d'un flot de données *(stream)*.
2. Ce *flot de données* est ensuite passé au module de traitement.

Le prototype actuel, fournit un support pour le protocole *HTTP* [8]. Il permet l'acquisition des flots *mpeg* audio et vidéo à travers le réseau, l'objectif étant de démontrer la viabilité de notre approche sur des documents multimédia réels. De plus, cette mise en œuvre a été menée dans la perspective d'extensions vers des protocoles temps réel comme *RTSP* [98].

VI.4.4 Médiateur d'exécution

Il fournit un ensemble de fonctions qui permettent le traitement de données provenant de l'interaction avec l'utilisateur, du médiateur d'accès et de l'ordonnanceur. Les fonctions assurées par le médiateur d'exécution sont :

- La **liaison** : elle associe à chaque objet multimédia de base le module de code qui l'implante.
- L'**héritage** des attributs de présentation : il assure la propagation des attributs de présentation.
- La **gestion des événements** : elle consiste à acheminer les événements de présentation entre les entités actives du système.

VI.4.4.1 Opération de liaison

À la construction de la représentation interne du document, la correspondance entre un objet de base et les modules de traitement offerts par l'application est réalisée par l'intermédiaire du type Mime [12]. Les types Mime supportés sont stockés dans une table interne (voir Fig. 6.13). Chaque entrée de la table est composée des deux champs suivants :

1. **Type :** est une chaîne de caractères qui indique le type de média de l'objet et son format, par exemple le type audio/mpeg.
2. **Identifiant du module de traitement :** donne la localisation du code de traitement correspondant, par exemple, le fichier binaire exécutable *madeus_mpeg*.

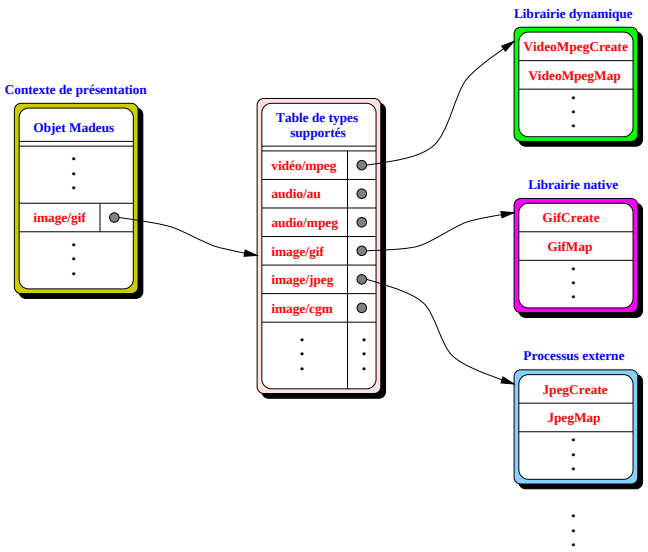


Fig. 6.13 : Opération de liaison

Les modules de traitement supportés par l'application sont de trois types : processus externe, librairie dynamique ou librairie native. Chaque module de traitement est spécialisé dans la manipulation d'un média et d'un format particulier. Les modules de traitement implantés couvrent les types de base suivants :

- Un module pour la **vidéo** aux formats *mpeg* vidéo niveau 1 et *mjpeg*. Il est réalisé sous forme d'un processus externe.
- Un module pour l'**audio** aux formats audio *mpeg* niveau 1 et AU. Il est réalisé sous forme d'un processus externe.
- Un module pour les **images** supportant les formats *xpm*, *gif*, *jpeg* et *png*. Il est réalisé sous forme d'une librairie native.

- Un module pour les **images** au format *cgm*. Il est réalisé sous forme d'une librairie dynamique et est chargé à la demande.
- Un module pour le **texte** qui supporte les fontes de *X11* et les couleurs. Il est réalisé sous forme d'une librairie native.

VI.4.4.2 Héritage des attributs de présentation

Lorsqu'un objet multimédia composé ou de base est en cours de présentation, les variables qui lui sont associées rendent compte des valeurs de ses attributs de présentation. On distingue deux types d'attributs :

- **Les attributs statiques :** ce sont les attributs dont les valeurs sont instantiées à la création d'un objet (transition *Create*).
- **Les attributs dynamiques :** ce sont des attributs dont les valeurs évoluent au cours de la présentation. Ils sont calculés à chaque transition *(Play_One_Step*) déclenchée par l'horloge.

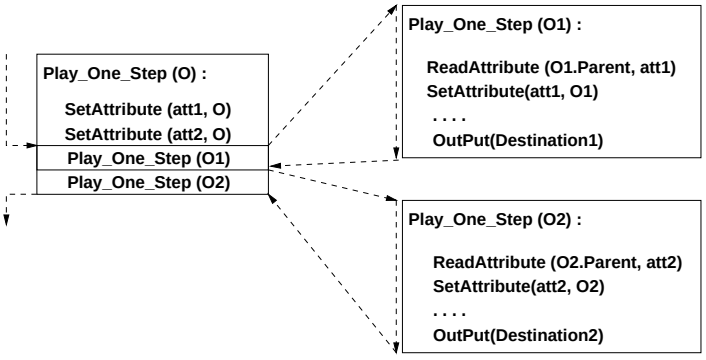


Fig. 6.14 : Héritage continu des attributs

À chaque réception d'une demande de création *Create*, le médiateur peut calculer les valeurs des attributs d'un objet de deux façons : si les valeurs sont définies de façon spécifique à cet objet, elles sont directement utilisées pour la présentation. Dans le cas contraire, ces valeurs sont d'abord héritées de l'objet englobant.

Pour les objets continus, lors de l'exécution d'un pas de présentation provenant de l'ordonnanceur, les différents attributs sont utilisés par le médiateur audio ou graphique pour définir les nouvelles caractéristiques de sortie des différents objets.

La transmission des valeurs d'attributs entre un objet père et ses fils est effectuée à chaque pas de la présentation. Lorsque l'ordonnanceur appelle la fonction *Play_One_Step* d'un objet composé (voir Fig. 6.14), celui–ci met à jour ses propres valeurs d'attributs. Lorsque l'ordonnanceur appelle la fonction *Play_One_Step* d'un objet de base, celui–ci prend en compte ces modifications en récupérant les valeurs de l'objet composé

englobant. Ceci nous amène à un schéma de propagation continue des attributs organisé en deux étapes (cf. Fig. 6.14) :

1. L'objet (de base ou composé) hérite les valeurs des attributs de l'objet parent (*ReadAttribute*) et met à jour ses propres attributs (*SetAttribute*).
2. Si l'objet est un type de base, il effectue en plus une opération de présentation élémentaire de sortie (*OutPut*) comme l'affichage à l'écran. Cette opération prend en compte toutes les mises à jour de ses attributs.

Cette gestion des attributs dépend de l'ordonnancement qui, comme nous l'avons souligné dans la section VI.3.2.2, propage les tics de l'horloge globale de façon descendante : un tic du document (*Play_One_Step*) est susceptible de changer un ou plusieurs attributs d'un objet composé qui, à son tour, les propage aux objets englobés.

VI.4.4.3 Gestion des événements

Lors de la présentation, plusieurs types d'événements peuvent se produire. Un événement peut provenir de plusieurs **sources** : interaction de l'utilisateur, opération de contrôle de la présentation ou échéance d'un délai de garde de l'horloge de la machine. Un événement déclenche un traitement au sein de l'application. Chaque événement est caractérisé par :

- Un identifiant symbolique qui correspond à un événement interne d'un objet multimédia (*Map*, *Unmap*, *Start*, etc.).
- Un identifiant de la source de l'événement.

Les différents événements de présentation sont pris en compte au sein du processus de l'application Madeus (processus *MADEUS client X11* Fig. 6.15) ainsi que dans les activités externes créées lors de la présentation (comme les processus *P1* et *P2* Fig. 6.15).

Lorsque l'utilisateur clique sur un objet au moyen de la souris, le médiateur est notifié par une fonction de rappel (*callback*) qui contient l'identifiant de l'objet ayant émis l'événement. Cet identifiant est obtenu grâce au mécanisme d'abonnement fourni par la boîte à outils *X11/Toolkit*[3].

---

(3) Fonction XtAdd_Call_Back.

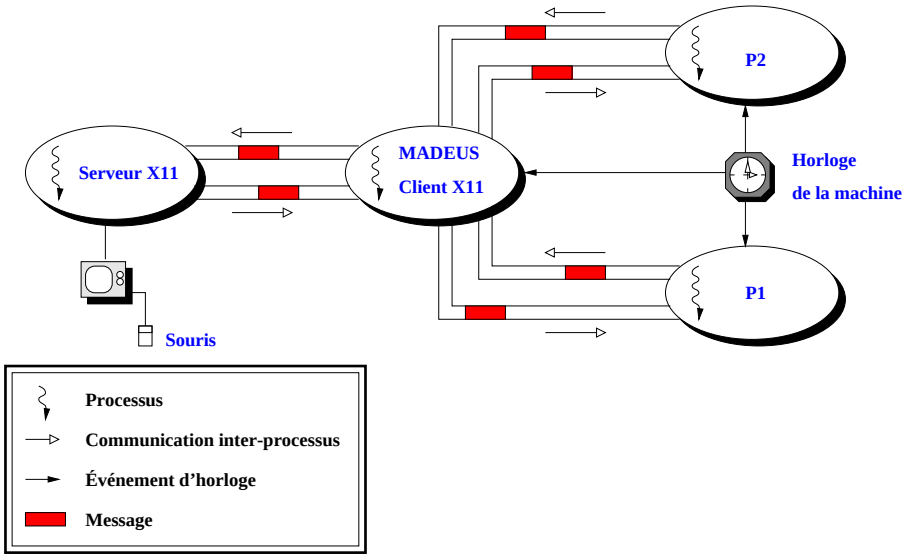


Fig. 6.15 : Gestion des événements

Dans le cas des processus externes, l'isolation(4) du contexte nécessite la gestion d'une communication bi–directionnelle permanente avec l'ordonnanceur. Cette communication est basée sur des échanges de messages entre le serveur *X11*, le processus Madeus et les processus externes (cf. Fig. 6.15).

VI.4.5 Médiateur de présentation

La restitution des données sur les périphériques de la machine est effectuée par deux médiateurs : le médiateur graphique et le médiateur audio. Ces médiateurs fonctionnent comme des serveurs qui reçoivent des demandes de création ou de suppression de modules destination attachés à chaque objet de base (cf. VI.4.2).

VI.4.5.1 Médiateur graphique

Il est dédié au traitement de l'information spatiale et au placement des différents objets sur l'écran graphique. Ce placement dépend d'un ensemble de paramètres qui sont définis à partir de la disposition relative des objets les uns par rapport aux autres [17].

Au moment de la création d'un objet, ses informations spatiales sont calculées et stockées dans une structure qui associe à chaque objet graphique une fenêtre d'affichage à l'écran. Cette opération tient compte du style attaché à l'objet comme la fonte pour le texte, la taille des caractères, la couleur de fond, etc.

Le médiateur graphique offre les possibilités suivantes :

---

(4) Dans *Unix*, les processus ne partagent pas le même espace d'adressage

- Le positionnement relatif des fenêtres à l'écran et leur déplacement dynamique (voir annotation 1 de la Fig. 6.16).
- La gestion de plans graphiques complexes comme les contours non–rectangulaires de fenêtres et les recouvrements (voir annotations 2 et 3 de la Fig. 6.16).
- Le rafraîchissement continu de l'écran lors des mouvements de fenêtres.
- La gestion des couleurs qui consiste à attribuer à chaque pixel d'une image, décrit en *RGB*, la couleur la plus proche dans une palette restreinte à 256 entrées.

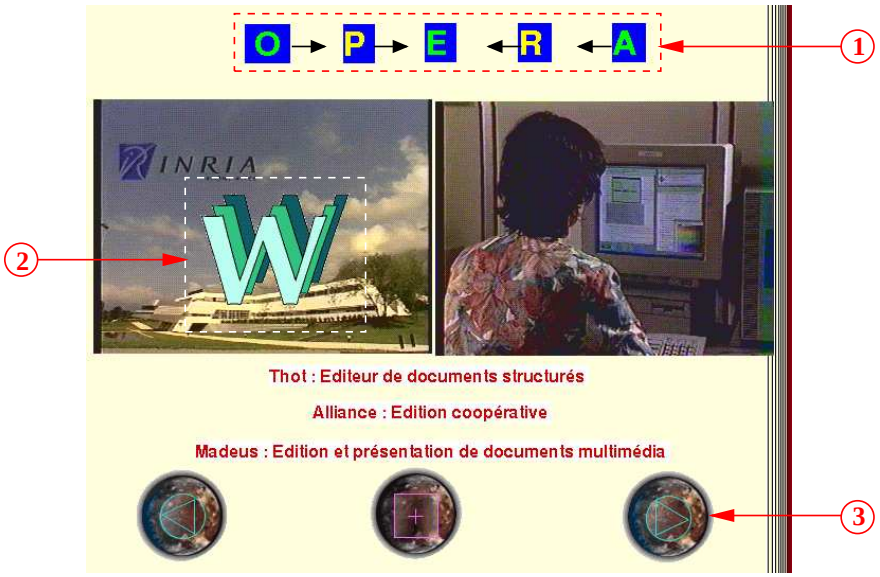


Fig. 6.16 : Médiateur graphique

VI.4.5.2 Médiateur audio

Nous décrivons le fonctionnement du médiateur audio à travers l'exemple de la Fig. 6.17. Dans cet exemple, trois objets A, B et C de type audio sont présentés en parallèle et chacun d'eux produit un flot audio constitué d'échantillons élémentaires de son.

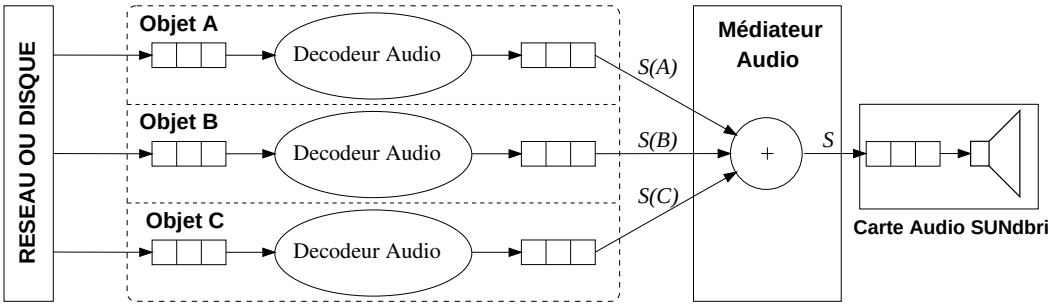


Fig. 6.17 : Périphérique logique audio

Chaque objet est implanté par un processus externe qui lit un flot compressé de données et délivre au médiateur un flot d'échantillons à travers des tampons. Chacun de ces échantillons est caractérisé par l'encodage du signal (u–law, a–law ou format linéaire), des paramètres de fréquence de saisie (8 Khz, 41 Khz, ...) et le nombre de canaux de sortie (mono, stéréo) [39].

Le rôle du médiateur audio consiste à traiter ces différents signaux dans le but de réaliser les fonctions suivantes :

- Appliquer les attributs de style audio liés à l'objet sur les échantillons produits S(A), S(B) et S(C). Ces attributs sont liés au volume et à la balance pour les stations disposant de sorties stéréo. Une fois cette opération achevée, le médiateur audio additionne les différents signaux pour produire un seul signal final.
- Adapter l'échantillon final aux paramètres de sortie du périphérique audio (fréquence et nombre de canaux).
- Synchroniser le tampon interne de la carte avec le flot produit par le médiateur. Cette synchronisation est nécessaire car les cartes son disposent d'un tampon interne de taille limitée (quelques Kilo octets). Ce tampon doit être maintenu plein tant qu'il y a des échantillons disponibles en entrée.

Les attributs de style audio peuvent être modifiés dynamiquement soit par les attributs de volume des objets audio soit à travers l'interface utilisateur (contrôle du volume global). Ces modifications sont prises en compte, lors de la présentation, grâce à la gestion centralisée de la production de l'échantillon final réalisée par le médiateur audio.

VI.5 Mise en œuvre de Madeus

Le système Madeus a été développé sur des stations de travail *Sun* sous le système d'exploitation Solaris2. La réalisation a duré deux ans et se poursuit actuellement par des développements complémentaires, en particulier sur le gestionnaire de présentation, le médiateur graphique et l'interface d'édition.

Une telle application regroupe des techniques de programmation très hétérogènes : interface graphique, algorithmes et structures de données complexes, manipulation de données multimédia comme l'audio, la vidéo, les images, le texte et la programmation système et réseau.

L'outil est presque entièrement écrit en langage C et représente, dans sa version actuelle, 48 812 lignes de code, dont 30 157 développées dans le cadre de cette thèse. Une partie du prototype est réalisée avec des librairies du domaine public. Le volume de code par partie du prototype est résumé dans le tableau de la Fig. 6.18.

Partie du prototype	**Langage**	**Boîte à outils**	**Développés**	**Réutilisés**
Système d'édition	C	lex + yacc	3 107	–
Gestionnaire temporel	C	–	4 442	–
Ordonnanceur	C	–	3 169	–
Médiateur graphique et Interface utilisateur	C	X11 et Motif	4148	–
Audio	C ++	lib–mpegaudio	2442	4 221
Vidéo	C	lib–mpeg	1822	7 300
Image	C	libjpeg, libpng	5 037	7 131
Texte	C	–	698	–
Gestion d'accès	C	lib–http	1783	–
Gestion des événements	C	socket Unix	3509	–

Fig. 6.18 : Répartition du volume de code dans Madeus

Ce tableau montre une grande disproportion entre le coût du développement du système de présentation par rapport au gestionnaire temporel et le système d'édition (75 %), ce qui constitue l'une des principales charges de réalisation de cette application. Nous avons cependant cherché à limiter cette charge en réutilisant le plus possible les logiciels disponibles actuellement (colonne de droite du tableau [?]).

Le développement du système de présentation était néanmoins nécessaire dans la mesure où, sans ce module, nous n'avions pas la possibilité de restituer un document, ni

même d'expérimenter l'intégration des fonctions d'édition et de présentation au sein d'un même système, ce qui était l'un des objectifs de cette thèse.

L'interface utilisateur, l'ordonnanceur et le gestionnaire de présentation sont implantés, dans l'espace utilisateur du système. L'exécution des activités parallèles de présentation est réalisée au moyen d'un chien de garde *(timer*) pour tous les objets multimédia d'un document, à l'exception de la vidéo et de l'audio. Ces derniers sont ordonnancés par un flot d'exécution concurrent et une horloge séparée.

Dans certaines implantations de systèmes multimédia, l'utilisation de processus légers est systématique pour ordonnancer tous les objets d'un document. Ce type de synchronisation, utilisé dans CMIFed [43], est pénalisant pour plusieurs raisons. L'emploi des processus légers engendre une dé−synchronisation entre les flots parallèles de présentation à cause du grand nombre d'appels au noyau du système qui en résulte [44][5]. Par ailleurs, il est nécessaire d'ajouter, pour chaque processus léger, un niveau supplémentaire de synchronisation inter−objets qui se révèle souvent très coûteux (emploi de sémaphores). Or, dans une présentation multimédia, le quantum de temps alloué aux différents flots de présentation est défini par rapport à des actions de présentation (mouvement d'un graphique à l'écran, affichage d'un objet, etc.) et non pas par rapport à des actions atomiques d'exécution (instruction du processeur).

VI.6 Bilan du développement de Madeus

Les principaux avantages que nous retenons de cette réalisation, au niveau applicatif, sont les suivants :

- **La généricité :** l'ordonnancement des objets par des horloges, et non par des fonctions de manipulation directe des objets, permet de garder un aspect générique à l'ordonnanceur [92].
- **L'adaptabilité :** la correspondance entre les horloges des objets de base et leur représentation dans le système d'exploitation (processus, *threads*, *timers*) peut être adaptée en fonction du type de l'objet multimédia et de la plate−forme d'exécution.
- **La réutilisation du graphe :** la correspondance entre les tâches à exécuter et le graphe d'exécution est parfaite. Sa réutilisation pour l'exécution évite d'avoir à reconstituer une structure d'exécution similaire au sein du système hôte. De plus, ce graphe constitue une donnée précieuse (prédiction du futur) qui peut être exploitée pour la gestion des ressources [42].

- L'**extensibilité du système** : le mécanisme d'exécution externe et de chargement dynamique permet l'**extensibilité** du système vers de nouveaux types d'objets multimédia sans modification de l'application (sans recompilation). Cette technique permet de répondre à la diversité croissante des types d'objets multimédia de base. Elle est, de nos jours, employée de façon courante dans les navigateurs World Wide Web : *les plugins* [82].

De toutes les limitations de conception, le système graphique est la partie la plus fragile de l'application Madeus. L'environnement *X11* constitue le goulot d'étranglement car, n'ayant pas été conçu pour être sollicité de façon intensive à travers le temps, il engendre des déviations temporelles relativement importantes [94]. À titre d'exemple, le temps de chargement d'une fonte peut parfois atteindre l'ordre de la seconde ! Ceci démontre que le système graphique comme le système d'exploitation restent encore très peu adaptés à ce type d'applications et justifie les recherches actuelles qui tentent d'y apporter des solutions [99].

En conclusion, le système Madeus est opérationnel et est actuellement utilisé par un étudiant de DEA, un étudiant en thèse et un ingénieur CNAM qui l'emploient comme plate–forme pour mener des expériences dans le cadre de leurs activités de recherche. Des documents multimédia ont également été élaborés à des fins de démonstration. L'outil a notamment été présenté lors des démonstrations :

- Journées nationales du PRC GDR IA à Grenoble.
- Exposition TEC 96.

VI.7 Conclusion

Dans ce chapitre, nous avons présenté le système de présentation de Madeus à travers son architecture et ses principales fonctions. Nous avons montré comment le graphe d'exécution était exploité pour l'ordonnancement de la présentation, pour la gestion de l'indéterminisme à travers le formatage dynamique, ainsi que pour la gestion de la navigation hypermédia. Ensuite, nous avons développé la partie liée au support des présentations multimédia à travers les différents médiateurs. Ces médiateurs permettent de restituer de façon concrète le document à l'utilisateur. Nous avons également présenté les premières conclusions concernant à la fois la démarche adoptée pour la réalisation du système Madeus et l'évaluation quantitative de l'effort nécessaire pour le développement de ce type d'applications.

Chapitre VII

Conclusion

VII.1 Rappel des objectifs

Le projet Opéra étudie les problèmes posés par la conception d'un environnement éditorial coopératif permettant l'édition, la maintenance et le traitement de documents structurés et multimédia. Mon travail de thèse concerne la conception et la réalisation d'un système d'édition et de présentation multimédia qui intègre des éléments de type texte, image, audio, vidéo ainsi que les interactions de l'utilisateur. Le nouveau type de documents visé est donc multimédia temporisé et interactif.

La construction d'un document multimédia est soumise à des contraintes multiples provenant de la nature diverse des média, de leur combinaison, et de leur intégration homogène au sein des dimensions multiples d'un document. À ces contraintes s'ajoutent celles de l'organisation d'une application qui est impliquée dans plusieurs couches d'un système.

La plupart des systèmes actuels d'édition / présentation multimédia proposent des méthodes d'édition fondées sur une approche impérative (langages de *script*) ou sur la datation explicite (les *timelines*). Dans ces systèmes, la construction d'un document est un processus long, source de nombreuses erreurs et difficile à maîtriser pour un auteur non spécialisé. Les documents produits sont souvent complexes, difficiles à maintenir et peu compatibles entre eux. En outre, la représentation des documents utilisée ne permet pas de traitement automatique du document (comme le copier/coller) qui fournirait à l'auteur une assistance efficace. Ces systèmes n'exploitent pas non plus la nature « élastique » du temps pour adapter le scénario d'un document aux vraies intentions de l'auteur en fonction du contexte.

L'étude des systèmes disponibles montre la nécessité de redonner le « contrôle » aux auteurs et de les assister par la conception d'outils « intelligents » qui facilitent la création d'un scénario, détectent les incohérences et permettent d'obtenir une représentation du document plus expressive et une restitution conforme à celle que l'auteur souhaite.

VII.2 Démarche suivie et bilan scientifique

Dans cette perspective, nous avons analysé la nature des documents multimédia et nous avons proposé une décomposition de l'organisation globale d'un document en quatre structures : logique, spatiale, temporelle et hypermédia. Ensuite, nous avons passé en revue les principales approches de l'édition multimédia à travers les méthodes de construction de scénario et les standards proposés. Nous avons étudié les langages de spécification de scénarios temporels pour le multimédia, ainsi que les techniques d'analyse et de synthèse, notamment dans un domaine où ces questions occupent une place prédominante : la planification, l'ordonnancement et la satisfaction de contraintes temporelles en intelligence artificielle.

Notre objectif était d'étudier si une approche déclarative de la spécification d'un scénario temporel pouvait constituer une alternative intéressante aux solutions actuelles. Nous avons alors défini une architecture d'application qui permet de raccourcir le cycle édition et présentation en tirant profit d'une approche similaire au *Wysiwyg* dans les documents statiques.

Le premier résultat est une méthode de construction mettant en œuvre ce principe ainsi qu'un format de document de haut niveau. Ce format rend compte des différentes dimensions du document et intègre la dimension temporelle des éléments de base et leurs relations de façon homogène. Ce format constitue notre *langage temporel* qui permet la spécification d'un scénario. Inspiré de la logique purement symbolique d'Allen [3], il a été étendu pour prendre en compte l'aspect quantitatif du temps, l'indéterminisme des durées de certains objets de base et les opérateurs permettant d'exprimer la causalité.

Le deuxième résultat est la conception d'un gestionnaire temporel qui prend en charge les spécifications temporelles de l'auteur. Ce gestionnaire, au cœur de notre système, met un œuvre des mécanismes permettant la construction d'un scénario et la vérification de sa cohérence qualitative, quantitative, causale et indéterministe. Ces deux derniers points soulèvent des problèmes théoriques difficiles, à ce jour non−résolus, et pour lesquels nous avons apporté une première solution. Le gestionnaire temporel permet aussi de produire une représentation interne du document sous forme de graphe adaptée pour la présentation. Il a été conçu pour permettre une édition incrémentale tout en gardant de très bonnes performances lors de la présentation (traitement des documents en une fois).

À partir de cette représentation nous avons conçu et développé un système de présentation qui permet la restitution d'un document. Ce système, qui constitue notre troisième résultat, exploite un automate d'états finis extrait du graphe d'exécution et permet de restituer une présentation multimédia à travers les dispositifs de la machine (écran graphique et haut−parleur). Ce système permet en plus de superviser dynamiquement le

déroulement d'une présentation pour prendre en compte l'indéterminisme, les interactions utilisateur et la navigation hypermédia intra– et inter–documents.

VII.3 Bilan et évaluation de la réalisation

Cette étude n'offrait un réel intérêt que si elle était concrétisée par la réalisation d'un prototype validant les idées qui en émergeaient. La réalisation de Madeus a comporté une part importante de développement pour mettre en œuvre le modèle de document et d'édition proposé. Elle a contribué au raffinement du modèle et d'une manière globale à toute la réflexion.

L'originalité de l'outil d'édition et de présentation développé dans le cadre de cette thèse se situe à plusieurs niveaux : d'abord les outils d'expérimentation sont encore très peu nombreux et les traitements considérés font intervenir une combinaison de schémas d'exécution souvent opposés : une présentation multimédia est à la fois conduite par les flots de données (les éléments multimédia), les flots de contrôle (enchaînements, interactions utilisateur et navigation hypermédia) et par la supervision dynamique de son déroulement (prise en compte de l'indéterminisme, des contraintes dues aux accès distants, des ressources disponibles, etc.).

Le prototype Madeus est construit selon une architecture en modules et comporte une grande partie de l'infrastructure nécessaire à ce type d'application. Cette architecture a été définie de manière à faciliter l'expérimentation de divers algorithmes d'analyse, de schémas d'exécution, ainsi que de la synchronisation système de bas niveau et des accès distants. Le prototype fonctionne de manière satisfaisante, avec des performances très acceptables. Il permet déjà de construire, de vérifier et de présenter des documents d'une certaine complexité. Cependant, toutes les fonctions ne sont pas supportées et les opérations de formatage temporel et de restitution graphique peuvent être améliorées.

VII.4 Perspectives

Cette contribution apporte des réponses aux problèmes qui se posent à chaque niveau de l'édition d'un document multimédia (spécification, analyse, synthèse et exécution). Elle constitue une base qu'il faudra encore améliorer et compléter. En plus de la consolidation des propositions faites dans cette thèse, le travail qui reste à faire peut s'orienter sur quatre grandes directions : le langage de spécification, l'interface utilisateur, les outils formels pour la synchronisation et la synchronisation au niveau du système et des protocoles réseaux.

Langage de spécification

Le premier champ d'activité concerne le langage de spécification de la synchronisation temporelle. L'intérêt de continuer ce travail est de combler les carences des langages proposés dans les outils existants :

- **Format de documents :** À court terme, mon projet est de participer activement au groupe de travail du consortium World Wide Web(1) qui définit les formats et langages pour le multimédia sur le Web (*MML : Multimedia Mark−Up Language*). L'objectif est d'étendre les documents HTML pour prendre en compte la dimension temporelle des présentations multimédia. Mon action consiste à valoriser, dans ce contexte, les résultats obtenus dans cette thèse sur la représentation des documents multimédia.
- **Multi−dimensions :** il s'agit de considérer un format qui intègre les différentes dimensions du document. L'étude que nous avons menée dans cette thèse est restreinte à la dimension temporelle et n'aborde pas les problèmes liés à l'interaction entre le temps et l'espace. Il s'agit d'une part de pouvoir décrire en terme de langage cette interaction et de résoudre les éventuels problèmes qui peuvent se poser. Par exemple, des conflits d'attribution de ressources graphiques (espace d'affichage) aux différents objets tout au long d'une présentation multimédia. En effet, deux éléments présentés de façon concurrente ne doivent pas se chevaucher spatialement. De même, des ressources limitées de l'environnement de présentation peuvent conduire à l'impossibilité de présenter de façon concurrente deux objets multimédia (CPU, cartes de décompression, périphérique audio, etc.).
- **Généricité :** Seules des instances de documents indépendants ont été considérées. Les travaux antérieurs du projet Opéra et ceux développés dans d'autres laboratoires [54] pour les documents multimédia ont montré les avantages qu'on pouvait tirer d'une approche générique des structures de document. Il convient d'étudier comment cette approche peut être étendue à la structure temporelle des documents multimédia.

Interface utilisateur

Le langage temporel est actuellement très « informatique ». Le problème central sera d'en trouver une version telle qu'il soit à la fois facile à apprendre et facile à

(1) http://www.w3.org

utiliser par les concepteurs de documents non informaticiens, tout en conservant toute sa puissance. Un des problèmes importants à résoudre sera celui d'un codage spatial de relations temporelles [63], sachant que les approches à base de représentation du graphe des relations (firefly) ne répondent pas aux besoins des auteurs. Un premier travail de visualisation est de permettre d'explorer l'ensemble des solutions d'une présentation à travers la manipulation directe des objets et des contraintes temporelles qui leurs sont attachées. Il faut ensuite passer à la phase d'édition en offrant à l'auteur des moyens de création et de modification entièrement graphique. Afin de considérer l'ensemble des dimensions du document, il faut traduire de façon graphique l'organisation logique du document et d'établir le lien avec sa dimension spatiale et hypermédia. Les problèmes posés restent nombreux dans ce domaine. À titre d'exemple, on peut citer le maintien continu de la cohérence de la représentation graphique, les problèmes d'interface homme–machine comme le multi–vues, la multi–modalité, etc.

Outils formels pour la synchronisation

À plus long terme, nous pensons aborder la formalisation d'un langage temporel primitif à base d'instants intégrant les aspects qualitatif et quantitatif, les interruptions, les rendez–vous et l'indéterminisme [24]. En particulier, la gestion de l'indéterminisme peut être traitée de façon plus complète en intégrant des techniques similaires à celles employées dans la synthèse de contrôleurs dans les systèmes temps réels. Ces techniques sont fondées sur la modélisation des scénarios au moyen d'automates temporisés. Le principe consiste à représenter de façon exhaustive (les états de l'automate) l'évolution d'un scénario suite aux différents événements d'une présentation (les transitions). Il s'agit alors de définir des stratégies qui permettent d'assurer dynamiquement le respect des contraintes temporelles suite aux différents événements indéterministes. En termes d'automates, il s'agit de choisir (par des prises de décision successives) des chemins dans l'automate qui permettent d'éviter que le système n'atteigne un état dit d'échec. Un tel état correspond à l'impossibilité de remettre le scénario temporel dans un état cohérent. Cette condition correspond à la propriété de contrôlabilité d'un scénario évoquée dans le chapitre V. Le problème peut ainsi être abordé sous l'angle de la théorie des jeux où le contrôleur (l'ordonnancer) modifie, à l'exécution, le scénario pour parer aux valeurs incontrôlables. Ces valeurs étant choisies par la « l'environnement d'exécution » qui représente l'adversaire du contrôleur dans la partie de jeu.

Synchronisation système et réseaux

Notre démarche s'est principalement focalisée sur le processus de construction d'un scénario et sur sa présentation en privilégiant le point de vue de l'auteur. Le travail effectué dans Madeus concernant l'aspect système et réseaux reste encore rudimentaire étant donnée qu'il ne couvre que les fonctions de base. Le travail de recherche qui reste à accomplir dans ce domaine reste très vaste et encore peu exploré. Nous pouvons néanmoins dresser trois voies qui nous semblent très prometteuses :

- **Support d'exécution :** le support d'exécution actuellement implanté est principalement adapté à une exécution locale de documents multimédia de petite taille. Cependant, on s'aperçoit dès à présent, avec l'utilisation du prototype Madeus, des variations de la qualité d'une présentation lorsque les documents multimédia sont volumineux ou visualisés dans un contexte réparti. Le volume important de données à traiter, à transférer et à stocker induit une dérive de la présentation que les outils comme Madeus doivent prendre en compte. Il faut donc que le schéma général de l'exécution d'une présentation soit très performant pour être en accord avec des traitements de type temps–réel. En outre, il doit être extensible de façon à intégrer de nouvelles technologies comme la diffusion (server push) ainsi que l'orientation objet comme dans activeX et MHEG permettant une plus grande intégration au sein de l'outil.
- **Supervision de la présentation :** l'objectif général au niveau de l'application multimédia est de concevoir et d'optimiser le traitement des documents hypermédia. Il est clair cependant qu'une réponse pertinente au problème de la présentation de documents multimédia ne pourra être correctement apportée que si l'on prend en compte les trois parties parties qui sont impliquées :
  1. L'environnement de présentation (machine cliente).
  2. Les contraintes du réseau comme la bande passante et les latences des accès.
  3. L'environnement de stockage (machine serveur de données pour les documents, la vidéo, l'audio, les images, le texte, etc.).

  Ainsi, les environnements lecteur (les clients) ont à la fois besoin d'une meilleure supervision de la présentation (adaptation de la dimension temporelle des documents présentés) ainsi que d'avoir à leur disposition des moyens d'agir sur les serveurs pour réaliser de telles adaptations. Du

côté du serveur, le mode de représentation des données, de leur stockage et surtout de leur accès doit être en conséquence adaptatif. Dans une première étape, il est nécessaire de définir des critères pour exprimer les contraintes de ressources à l'échelle des documents multimédia interactifs, ensuite il faut concevoir un processus de supervision permettant, à partir de l'évolution de ces paramètres, de décider dynamiquement quand et comment les adaptations doivent être mises en œuvre.

- **Protocoles de communication :** TCP est certainement avec IP, l'un des protocoles qui a contribué le plus au succès et au développement d'Internet et des systèmes hypermédia en général. Cependant l'arrivée des applications "temps–réel" sur l'Internet a montré les limites de ce protocole. En effet, les applications ont des exigences en terme de latence de transmission et de support multipoint beaucoup plus importantes que les applications de type ftp ou http. Face à ces nouveaux besoins, de nouveaux protocoles de transport comme RTP, RTSP ont été proposés. Cependant, Internet ne permet pas aujourd'hui de faire de la réservation de ressources ou même d'instaurer des priorités sur les flux transmis. Dans ces conditions, le rôle des protocoles de transport est assez limité. Par conséquent, pour obtenir de bonnes performances, il est indispensable que les applications s'adaptent en temps réel ou de façon prédictive au débit du réseau. Ces adaptations peuvent être de différentes natures. L'application pourra, par exemple, changer l'algorithme de compression en fonction du débit disponible ou encore ajouter de la redondance dans son codage de façon à augmenter sa résistance aux pertes de paquets dans le réseau.

Dans les différentes évolutions du système que nous venons de citer, l'originalité du travail à réaliser consiste à tirer profit de l'aspect prédictif du graphe de contraintes. Il sera alors possible de redéfinir la notion de qualité de service à l'échelle du document tout entier contrairement aux travaux actuels [102]. En outre, les accès aux données à travers le réseau peuvent être réalisés d'une façon plus prédictive [59].

Au terme de cette étude, nous constatons que les objectifs que nous nous étions fixés étaient réalistes dans la mesure où nous avons pu concevoir et mettre en œuvre un système pour documents multimédia qui intègre à la fois l'édition et la présentation, tout en sachant que tous les problèmes liés à l'édition et à la présentation ne pouvaient pas être couverts de façon complète.

Bibliographie

[1] ADIBA M., ‘‘STROM : Structural and Temporal Object–oRiented Multimedia database system’’, *Proceedings of IEEE International Workshop on Multi−Media Database*, pp. 12–19, Blue Mountain Lake, New York, USA, août 1995.

[2] AKPOTSUI E. K. A., *Transformations de types dans les systèmes d'édition de documents structurés*, Thèse de doctorat, Institut National Polytechnique de Grenoble, octobre 1993.

[3] ALLEN J. F., ‘‘Maintaining Knowledge about Temporal Intervals’’, *Communications of the ACM*, 26(11), pp. 832–843, novembre 1983.

[4] ALLEN J. F., ‘‘Time and Time Again: The Many Ways to Represent Time’’, *International Journal of Intelligent Systems*, 6, pp. 341–355, 1991.

[5] ANDERSON T. E. et al., ‘‘Scheduler Activations: Effective Kernel Support for the User–level Management of Parallelism’’, *Thirteen ACM Symposium on Operating Systems Principles*, pp. 95–109, SIGOPS vol(25), Pacific Grove, CA, USA, 1991.

[6] APPLE Computer Inc. QuickTime Reference Manual, *1991*.

[7] BERLANDIER P., *Deux variations sur le thème de la consistance d'arcs : maintien et renforcement*, (Rapport 2426), Institut National de Recherches en Informatique et Automatique, décembre 1994.

[8] BERNERS–LEE T., CAILLIAU R., GROFF J. et POLLERMANN B., ‘‘World–Wide Web : The Information Universe’’, *Electronic Networking: Research, Applications and Policy*, Vol. 1(No. 2), printemps 1992.

[9] BLAIR G. S. et al., *A Hybrid Aproach to Real−Time Synchronisation in Distributed Multimedia Systems*, (No.MPG–94–21), Lancaster University Distributed Multimedia Research Group, 1994.

[1 0] BLAKOWSKI G. et STEINMETZ R., ‘‘A Media Synchronisation Survey: Reference Model, Specification, and Case Studies’’, *IEEE Journal Of Selected Ares In Communications*, 14(1), pp. 5–34, janvier 1996.

[1 1] BOURSIER P. et TAUFOUR P.A., *La technologie Multimédia*, Hermès, Paris, mars 1994.

[1 2] BORNSTEIN N. et FREED N., ‘‘MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the format of Internet Message Bodies’’, *RFC 1521*, Bellcore, Innsoft, septembre 1993.

[1 3] BRAY T. et SPERBERG C. M., *Extensible Markup Language (XML)*, (WD–xml–961114), World Wide Web Consortium, novembre 1996.

[1 4] BUCHANAN M. C. et ZELLWEGER P. T., ‘‘Automatic Temporal Layout Mechanisms’’, *Proceedings of the First ACM International Conference on Multimedia*, pp. 341–350, ACM Press, Anaheim, Californie, août 1993.

[1 5] BUFORD J. F. K., *Multimedia Systems*, Addisson–Wesley, New York, 1994.

[1 6] BULTERMAN D. C. A., ‘‘Embedding Video in Hypermedia Documents: Supporting Integration and Adaptive Control’’, *ACM Transactions on Information Systems*, pp. 1–30, octobre 1995.

[1 7] CARCONE L., *Gestion de contraintes spatiales dans Madeus : un système d'édition multimédia*, Mémoire d'ingénieur Cnam, Grenoble, à paraître, 1997.

[1 8] CHEN Z., TAN S., CAMPBELL R. et LI Y., ‘‘Real Time Video and Audio in the World Wide Web’’, *Proceedings of the Fourth International Conference on the World Wide Web*, Boston, MA, USA, décembre 1995.

[1 9] CHLEQ N., ‘‘Efficient Algorithms for Networks of Quantitative Temporal Constraints’’, *CONSTRAINT'95 : the international workshop on constraint–based reasonning, held in conjunction with FLAIRS–95*, url : http://www.sci.tamucc.edu/constraint95/home.html, 1995.

[2 0] CLARK D., ‘‘The Structuring of Systems Using Upcalls’’, *Proceedings of the 10th ACM SIGOPS Symposium on Operating System Principles*, édité par ACM Press, pp. 171–180, 1985.

[2 1] CLAVES P., *Restructuration dynamique de documents structures*, Mémoire d’ingénieur CNAM, Concervatoire Natinal des Arts et Métiers, mars 1995.

[2 2] COULSON G., *Multimedia Application Support in Open Distributed Systems*, Thèse de doctorat, Computing Department, Lancaster University, avril 1993.

[2 3] COULSON G., BLAIR G. S., DAVIES N. et WILLIAMS N., ‘‘Extensions to ANSA for Multimedia Computing’’, *Computer Networks and ISDN Systems*, (25), pp. 305–323, 1992.

[2 4] COURTIAT J.P. et DE OLIVERA R. C., ‘‘Proving Temporal Consistency in a New Multimedia Synchronisation Model’’, *Proceedings of the Fourth ACM Conference on Multimedia*, ACM Press, Boston, novembre 1996.

[2 5] DALAL M. et Al, ‘‘Negotiation for Automated Generation of Temporal Multimedia Presentations’’, *Proceedings of the Fourth ACM Conference on Multimedia*, ACM Press, Boston, novembre 1996.

[2 6] DANTZIG G. B., *Linear Programming and Extensions*, Princeton University Press, Princeton, NJ, 1962.

[2 7] DECHTER R., MEIRI I. et PEARL J., ‘‘Temporal Constraint Networks’’, *Artificial Intelligence*, (49), pp. 61–95, 1991.

[2 8] DEROSE S. J. et DURAND D. G., *Making Hypermedia Work: A User’s guide to HyTime*, Kluwer Academic Publishers, mars 1994.

[2 9] DORCEY T., ‘‘The CU–SeeMe Desktop Videoconferencing Software’’, *ConneXions − The Interoperability Report*, 9(3), pp. 42–45, mars 1995.

[3 0] DRAPEAU G. D., ‘‘Synchronization in the MAEstro Multimedia Authoring Environment’’, *Proceedings of the First ACM Conference on Multimedia*, pp. 331–340, ACM Press, Anaheim, Californie, août 1993.

[3 1] DUBOIS D., FARGIER H. et PRADE H., ‘‘The Use of Fuzzy Constraints in Job–Shop Scheduling’’, *In Proceedings of IJCAI 93 Workshop on Knowledge−Based Planning, Scheduling and Control*, Chambéry (France), 1993.

[3 2] ERFLE R., ''Specification of temporal constraints in multimedia documents using HyTime'', *Electronic Publishing*, 6(4), pp. 397–411, décembre 1993.

[3 3] FLOYD W. R., ''Algorithm 97 (shortest path)'', *Communications of the ACM*, 5(6), pp. 345, 1962.

[3 4] FREDERICK R., ''Experiences with real–time software video compression'', *Sixth International Workshop on Packet Video*, pp. 1.1–1.4, Portland, Oregon, septembre 1994.

[3 5] FARGIER H., LANG J. et SCHIEX T., ''Mixed Constraint Satisfaction: a Framework for Decision Problems under Incomplete Knowledge'', *Proceedings of AAAI Conference*, Portland, Oregon, USA, 1996.

[3 6] FURTH B., ''Multimedia Systems: An Overview'', *IEEE Multimedia*, Vol. 1(No. 1), pp. 47–59, printemps 1994.

[3 7] GHALLAB M., *Managing Temporal Knowledge: A Survey on Problems and Approaches*, (Rapport LAAS 92520), Centre National de la Recherche Scientifique, Toulouse, France, octobre 1992.

[3 8] GHALLAB M. et VIDAL T., ''Focusing on a Sub–Graph for Managing Efficiently Numerical Temporal Contraints'', *Proceedings of FLAIRS*, Melbourne Beach (FL), 1995.

[3 9] GIBBS J. S. et TSICHRITZIS D. C., *Multimedia Programming, Objects, Environments and Frameworks*, ACM Press, 1994.

[4 0] GOLDFARB C. F., *The SGML Handbook*, Oxford University Press, Oxford, 1990.

[4 1] HAMAKAWA R. et REIKIMOTO J., ''Object Composition and Playback Models for Handling Multimedia Data'', *Proceedings of the First ACM International Conference on Multimedia*, pp. 273–281, ACM Press, Anaheim, Californie, août 1993.

[4 2] HARDMAN L., BULTERMAN D. C. A. et VAN ROSSUM G., ''Links in Hypermedia: the requirement for Context'', *Hypertext'93 Proceedings*, pp. 183–191, novembre 1993.

[4 3] HARDMAN L., VAN ROSSUM G. et BULTERMAN D. C. A., ''Structured Multimedia Authoring'', *Proceedings of the First ACM International Conference on Multimedia*, pp. 283–289, ACM Press, Anaheim, Californie, août 1993.

[4 4] HÖPNER P., ''Presentation Scheduling of Multimedia Objects and Its Impact on Network and Operating System Support'', *Proceeding of the Second International Workshop on Network and Operating System Support for Digital Audio and Video*, édité par Herrtwich R. G., LNCS N. 614 Springer Verlag, Heidelberg, 1991.

[4 5] HyTime Information Technology, ''Hypermedia/Time–based Structuring Language (HyTime)'', *ISO/IEC 10743*, novembre 1992.

[4 6] IBM CORPORATION, *Audio Visual Connection User's Guide and Authoring Language Reference*, IBM Form S15F–7134–02, août 1990.

[4 7] IINO M., DAY Y. F. et GHAFOOR A., ''An Object–Oriented Model for Spatio–Temporal Synchronization of Multimedia Information'', *Proceedings of the International Conference on Multimedia Computing and Systems*, pp. 110–119, Boston, Massachusetts, 14–19 mai 1994.

[4 8] INTERNATIONAL STANDARD ORGANIZATION, *International Standard ISO/IEC 8613: Information Processing − Text and Office Systems − Office Document Architecture (ODA) and Interchange Format*, Genève/New York, 1989.

[4 9] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, *International Standard ISO/IEC 8613: HyperODA − extensions for temporal relationships, extensions for non−linear structures*, International Organization for Standardization, 1992.

[5 0] ISO/IEC 14478–1, *Information Processing Systems −− Computer Graphics and Image Processing −− Presentation Environments for multimedia Objects (PREMO)*, (1), mai 1996.

[5 1] JACOBSON V., *VAT: Visual Audio Tool Manual*, Lawrence Berkeley Laboratory, USA, février 1993.

[5 2] KAUTZ H. et LADKIN P. B., ''Integrating metric and qualitative temporal reasoning'', *In Proceedings of AAAI−91*, pp. 241–246, Anaheim, CA, 1991.

[5 3] KHACHIYAN L. G., *A Polynomial Algorithm in Linear Programming*, Soviet Methematics Doklady , 191–194(20), 1979.

[5 4] KHALFALLAH H. et KARMOUCH A., ''An architecture and a data model for integrated multimedia documents and presentation applications'', *Multimedia Systems*, 1995(3), pp. 238–250, 1995.

[5 5] KIM M. Y. et SONG J., ''Multimedia Documents with Elastic Time'', *Proceedings of the Third ACM International Conference on Multimedia*, édité par ACM Press, pp. 143–154, Polle Zellweger, San Francisco, Californie, 5–9 novembre 1995.

[5 6] KNUTH D. et PLASS M., ''Breaking Paragraphs into Lines'', *Software Practice and Experience*, 11(11), pp. 1119–1184.

[5 7] KRAKOWIAK S., *Principes des systèmes d'exploitation*, Dunod, 1987.

[5 8] KSHIRASAGAR N., ''Specification and Systhesis of a Multimedia Synchronizer'', *IEEE Proceedings of the International Conference on Multimedia Computing and Systems*, pp. 544–549, IEEE Computer Society Press, 1994.

[5 9] KWAN B. et KARMOUCH A., ''A Multimedia Agent in Distributed Broadband Environment'', *Proceedings of ICC'96*, Dallas, USA, juin 1996.

[6 0] LAMPORT L., ''Time, Clocks and the Ordering of Events in a Distributed System'', *Communications of the ACM*, Vol. 21(No. 7), pp. 558–565, juillet 1978.

[6 1] LAMPORT L., *LATEX: a document preparation system*, Addison–Wesley, New York, 1986.

[6 2] LAYAÏDA N., *Schéma de désignation extensible dans les systèmes répartis*, Rapport de DEA, Université Joseph Fourier, Juin 1993.

[6 3] LAYAÏDA N. et VIONDURY J.Y., ''Multimedia Authoring: a 3D Interactive Visualization Interface based on a Structured Document Model'', *Proceedings of the Sixth International Conference on Human−Computer Interaction*, Elsevier Science Publishers, Yokohama, Japon, juillet 1995.

[6 4] LAYAÏDA N. et SABRY–ISMAIL L., ‘‘Maintaining Temporal Consistency of Multimedia Documents Using Constraint Networks’’, *Proceedings of the International Conference on Multimedia Computing and Networking*, San José, CA, USA, janvier 1996.

[6 5] LAYAÏDA N. et SABRY–ISMAIL L., ‘‘MADEUS : un modèle de documents multimédia structurés’’, *Techniques et science informatiques*, 15(9), pp. 1227–1257, novembre 1996.

[6 6] LEGALL D., ‘‘MPEG : A Video Compression Standard for Multimedia Applications’’, *Communications of the ACM*, Vol. 34(No. 4), pp. 45–68, avril 1991.

[6 7] LEISERSON C. E. et SAXE J. B., ‘‘A mixed–integer linear programming problem which is efficiently solvable’’, *Proceedings of the 21st Annual Allerton Conference on Communication, Control, and Computing*, pp. 204–213, 1983.

[6 8] LIAO Y. Z. et WONG C. K., ‘‘An Algorithm to compact a vlsi symbolic layout with mixed constraints’’, *IEEE Transactions on Computer−Aided Design of Integrated Circuits and Systems*, CAD(2), pp. 62–69, 1993.

[6 9] LIAO W. et LI V. O. K., ‘‘Synchronization of Distributed Multimedia Systems with User Interactions’’, *Multimedia Modeling Conference MMM’96*, édité par COURTIAT J.P., DIAZ M. et SENAC P., pp. 237–252, World Scientific, Toulouse, novembre 1996.

[7 0] LIESTØL G., ‘‘Aesthetic and Rhetorical Aspects of Linking Video in Hypermedia’’, *Proceedings of the ACM European Conference on Hypermedia Technology*, pp. 217–222, ACM Press, Edimbourg, Écosse, septembre 1994.

[7 1] LITTLE T.D.C. et GHAFOOR A, ‘‘Synchronization and Storage Models for Multimedia Objects’’, *IEEE Journal on Selected Areas in Communications*, Vol. 8(3), pp. 413–426, avril 1990.

[7 2] LITTLE T. D. C. et GHAFOOR A., ‘‘Network Considerations for Distributed Multimedia Objects Composition and Communication’’, *IEEE Network Magazine*, Vol. (4), pp. 32–49, novembre 1990.

[7 3] LITTLE T. D. C. et GHAFOOR A., ''Interval–Based Conceptual Models for Time–Dependent Multimedia Data'', *IEEE Transactions on Knowledge and Data Engineering (Special Issue : Multimedia Information Systems)*, Vol. 5(4), pp. 551–563, août 1993.

[7 4] MACKWORTH A. K. et FREUDER E. C., ''The Complexity of Some Polynomial Network Consistency Algorithms for Constraint Satisfaction Problems'', *Artificial Intelligence*, 25(1), pp. 65–74, 1985.

[7 5] MACROMEDIA DIRECTOR, *User's Guide*, Macromedia Inc., 1995.

[7 6] MARKEY B. D., ''HyTime and MHEG'', *IEEE COMPCON'92 International Conference*, pp. 25–39, IEEE Computer Society Press, 1992.

[7 7] MEIRI I., *Temporal Reasoning: A Constraint Based Approach*, (R–173), Cognitive Systems Laboratory, University of California, Los Angeles, CA90024, janvier 1992.

[7 8] MINGLU L., YONGQIANQ S. et HUANYE S., ''Temporal Representation for Multimedia Systems'', *Time 96 International Conference*, Florida, USA, 19–20 mai 1996.

[7 9] Motif, *Programmer's Guide*, Open Software Foundation, 11 Cambridge Center, Cambridge, MA 02142, 1992.

[8 0] NANARD J. et NANARD M., ''Should Anchors be typed Too? An Experiment with MacWeb'', *Proceedings of the Fifth ACM Conference on Hypertext*, pp. 51–62, ACM Press, Seattle, Washington, novembre 1993.

[8 1] NEWCOMB S. R., KIPP N. A. et NEWCOMB V. T., ''the Hytime Hypermedia/Time–based Document Structuring Language'', *Communications of the ACM*, 34(11), pp. 67–83, novembre 1991.

[8 2] Netscape Communications Corporation, *Netscape Navigator : Plug–in Guide*, juillet 1996.

[8 3] OGAWA R., TANAKA E., TAGUCHI D. et HARADA K., ''Design Strategies for Scenario–based Hypermedia: Description of its Structure, Dynamics, and Style'', *Proceedings of the ACM Conference on Hypertext*, pp. 71–80, ACM Press, Milan, Italie, novembre 1992.

[8 4] OZSU M. T., SZAFRON D., EL–MADANI G. et VITTAL C., ‘‘An object–oriented multimedia database system for a news–on–demand application’’, *Multimedia Systems*, (3), pp. 182–203, 1995.

[8 5] PAPADIMITRIOU C. H. et STEIGLITZ K., *Combinatorial Optimization: Algorithms and Complexity*, Prentice–Hall, Englewood Cliffs, NJ, 1982.

[8 6] PEREZ–LUQUE M. J. et LITTLE T. D. C., ‘‘A Temporal Reference Framework for Multimedia Synchronisation’’, *IEEE Journal on Selected Areas in Communications*, 14(1), pp. 36–51, janvier 1996.

[8 7] PRICE R., ‘‘MHEG: An Introduction to the Future International Standard for Hypermedia Object Interchange’’, *Proceedings of the First ACM Conference on Multimedia*, pp. 121–128, ACM Press, Anaheim, Californie, août 1993.

[8 8] QUINT V. et VATTON I., ‘‘Grif: An Interactive System for Structured Document Manipulation’’, *Proceedings of the International conference on Text Processing and Document Manipulation*, pp. 200–213, Cambridge University Press, 1986.

[8 9] QUINT V. et VATTON I., ‘‘Combining Hypertext and Structured Documents in Grif’’, *Proceedings of ECHT’92*, édité par Lucarella D., pp. 23–32, ACM Press, Milan, décembre 1992.

[9 0] RIT J. F., *Modélisation et Propagation de Contraintes Temporelles Pour la Planification*, Thèse de doctorat, Institut National Polytechnique de Grenoble, 7 mars 1988.

[9 1] ROISIN C. et VATTON I., ‘‘Merging Logical and Physical Structures in Documents’’, *Proceedings of the Fifth International Conference on Electronic Publishing, Document Manipulation and Typography*, pp. 327–337, Wiley Publishers, Darmstadt, Allemagne, avril 1994.

[9 2] ROTHERMEL K. et HELBIG T., ‘‘Clock Hierarchies: An Abstraction for Grouping and Controling Media Streams’’, *IEEE Journal on Selected Areas in Communications*, 14(1), pp. 174–184, janvier 1996.

[9 3] RUTLEDGE L., *A HyTime Engine for Hypermedia Document Presentation*, Master en informatique, Université de Massachusetts, Lowell, 1993.

[9 4] SASINOWSKI J. E. et STROSNIDER J. K., ‘‘ARTIFACT: A Platform for Evaluating Real−Time Window Systems Designs’’, *Proceeding of the 16th IEEE Real−Time Systems Symposium*, pp. 342−352, Pisa, Italie, décembre 1995.

[9 5] SCHLOSS G. A. et WYNBLATT M. J., ‘‘Building Temporal Structures in a Layered Multimedia Data Model’’, *Proceedings of the Second ACM International Conference on Multimedia*, édité par Domanico Ferrari, pp. 271−278, ACM Press, San Francisco, California, octobre 1994.

[9 6] SENAC P., *Contribution à la modèlisation des systèmes multimédias et hypermédias*, Thèse de doctorat en informatique, Université Paul Sabatier de Toulouse, 12 juin 1996.

[9 7] SHOSTAK R., ‘‘Deciding linear inequalities by computing loop residues’’, *JACM*, (28), pp. 769−779, 1981.

[9 8] SCHULZRINNE H., ANUP R. et LANPHIER R., *Real Time Streaming Protocol (RTSP)*, (Internet−Draft), février 1997.

[9 9] SCHULZRINNE H., ‘‘Operating System Issues for Continuous Media’’, *Multimedia Systems*, 4(5), pp. 269−280, 1996.

[1 0 0] SIOCHI A., FOX E. A., HIX D., SCHWARTZ E. E. et NARASIMHAN A., ‘‘The Integrator: A Prototype for Flexible Development of Interactive Digital Multimedia Applications’’, *Interactive Multimedia*, 2(3), pp. 5−26, 1993.

[1 0 1] SONG J., DOGONATA Y. N., KIM M. Y. et TANTAWI A. N., ‘‘Modelling Timed User−Interactions in Multimedia Documents’’, *IEEE Proceedings of the International Conference on Multimedia Computing and Systems*, pp. 407−416, IEEE Computer Society Press, 1996.

[1 0 2] STAEHLI R. A., *Quality of Service Specification for Resource Management in Multimedia Systems*, Thèse de doctorat, Oregon Graduate Institute of Science and Technologie, juillet 1995.

[1 0 3] STEINMETZ R., ‘‘Human perception of jitter and media synchronisation’’, *IEEE Journal on Selected Areas in Communications*, 14(1), pp. 61–72, janvier 1996.

[1 0 4] SUN MICROSYSTEMS, *The Java Language Specification*, Sun Microsystems, 1995.

[1 0 5] TELEMEDIA INC., *Process Management in UNIX*, Computer Technology Group, Unit 4, 1990.

[1 0 6] TURLETTI T., ‘‘IVS: The INRIA Videoconferencing System’’, *ConneXions − The Interoperability Report*, 8(10), pp. 11–22, octobre 1994.

[1 0 7] VALDES R., ‘‘What’s UP at Kaleida ?’’, *Dr Dobb’s Developper Update*, Vol. 1(No. 7), septembre 1994.

[1 0 8] VANBEEK P., ‘‘Reasonning about qualitative temporal information’’, *Proceedings of AAAI*, Boston, MA, pp. 728–734, 1990.

[1 0 9] VIDAL T. et GHALLAB M., ‘‘Temporal Constraints in Planning: Free or not Free ?’’, *Constraint 95, 2nd International Workshop on Constraint−Based Reasoning*, Melbourne (Fl, USA), avril 1995.

[1 1 0] VIDAL T., *Le temps en Planification et en Ordonnacement : Vers une Gestion Complète et Efficaces de Contraintes Hétérogènes et Entachées d’Incertitude*, Thèse de doctorat, Université Paul Sabatier de Toulouse, septembre 1995.

[1 1 1] VIDAL T., *A preliminary charecterization of Dynamic Controllability in Contingent Temporal CSPs*, (Rapport interne), janvier 1997.

[1 1 2] VILAIN M. et KAUTZ H. A., ‘‘Constraint Propagation Algorithms for Temporal Reasoning’’, *Proceedings of AAAI*, pp. 377–382, Philadelphia, août 1986.

[1 1 3] WAHL T. et ROTHERMEL K., ‘‘Representing Time in Multimedia Systems’’, *IEEE Proceedings of the International Conference on Multimedia Computing and Systems*, pp. 538–543, IEEE Computer Society Press, Boston, Massachusetts, 14–19 mai 1994.

[1 1 4] WALLACE G., ‘‘The JPEG Still Picture Compression Standard’’, *Communications of the ACM*, Vol. 34(No. 4), pp. 30–44, avril 1991.

[1 1 5] WYNBLATT M. J. et SCHLOSS G. A., ''Control Layer Primitives for the Layered Multimedia Data Model'', *Proceedings of the ACM International Conference on Multimedia* , édité par ACM Press, pp. 167–177, Zellweger P., San Francisco, CA, novembre 1995.

[1 1 6] WEISS R., DUDA A. et GIFFORD D. K., ''Composition and Search with a Video Algebra'', *IEEE Multimedia Magazine*, 2(1), 1995.

[1 1 7] WEITZMAN L., *The Architecture of Information*, Thèse de doctorat, Massachussets Institute of Technology, février 1995.

[1 1 8] ZEDAN H., *REAL−TIME SYSTEMS : Theory and Applications*, Elsevier Science Publishers B. V., North–Holland, 1990.