Université Joseph Fourier
U.F.R
Informatique & Mathématiques Appliquées

Institut National Polytechnique de Grenoble
ENSIMAG

I.M.A.G

DEA Informatique :

Système et Communication

Projet présenté par:

Laurent TARDIF

Visualisation du scénario temporel d'un document multimédia

Effectué dans le projet Opéra – INRIA

Date:

Juin 1997

Membres du Jury:

Joëlle Coutaz
Brigitte Plateau
Andrzej Duda
Muriel Jourdan

Remerciements

Je remercie tout d'abord Joëlle COUTAZ, Brigitte PLATEAU et Andrzej DUDA pour avoir bien voulu faire partie de ce jury.

Je remercie Cécile ROISIN et Vincent QUINT pour m'avoir accueilli au sein du projet Opéra où j'ai effectué mon stage.

Je tiens à exprimer ma profonde gratitude à Muriel JOURDAN responsable de mon stage pour m'avoir accueilli et pour son dévouement constant à m'apporter l'aide dont j'avais besoin et les remarques qu'elle a apportées à mon travail.

Je remercie chaleureusement Nabil LAYAÏDA, sans qui ce travail n'aurait jamais pu avoir lieu et Laurent CARCONE pour sa bonne humeur communicative.

Je remercie Irène VATTON et Cécile ROISIN pour leur patience lors de la relecture de mon rapport.

Je remercie enfin José, Christophe, Jean-Christophe, toute l'équipe d'Opéra et les membres du W3C pour leur sympathie.

Et enfin, je souhaite un prompt rétablissement à notre cher ami Loay SABRY.

Chapitre I

Edition et visualisation du scénario temporel d'un document multimédia

Chapitre II

Etat de l'art sur la visualisation de données

Chapitre III

Visualisation du scénario temporel au sein de Madeus

Introduction

1 Présentation générale

Aujourd'hui, les documents électroniques intègrent une dimension temporelle en plus des dimensions spatiale (présentation graphique du document), logique (décomposition en parties,...) et sémantique (hypertexte). L'édition de documents comportant des objets multimédia (vidéo, audio, images, etc) organisés dans la dimension temporelle est une tâche complexe. En effet, la nature dynamique des objets manipulés (tels que la vidéo et l'audio) ainsi que la définition de leur organisation temporelle empêche la fusion des phases d'édition et de présentation. Le principe statique du WYSIWYG (What You See Is What You Get) fort utile lors de l'édition d'un document classique ne peut donc s'appliquer à la création d'un document multimédia.

Edition de documents multimédia

De ce fait, les outils d'édition actuels de ce type de documents ressemblent plus à des outils de programmation, par l'utilisation de langages de script pour décrire l'organisation temporelle des objets. Néanmoins, il serait dommage de limiter les auteurs potentiels de documents multimédia aux seuls experts informaticiens. Un des objectifs fixés par le projet Opéra est la création d'un éditeur multimédia puissant ayant une interface graphique conviviale, utilisable par des auteurs non–informaticiens.

La seule dimension d'un document multimédia qui nous intéresse dans ce mémoire est la dimension temporelle dans laquelle est définie la façon dont s'organisent les objets dans le temps(1). On appelle **scénario temporel** d'un document multimédia la description de cette organisation. La description de ce scénario (textuelle, graphique) fait partie de la **phase d'édition** du document. Lors de la **phase de présentation** les objets sont placés dans le temps de manière à respecter les consignes du scénario temporel.

Un système auteur pour la composition de documents multimédia se compose de deux sous–systèmes : le système d'édition et le système de présentation. Le rôle du **système d'édition** consiste à alléger la charge de

(1) Les autres domaines ont fait l'objet d'important travaux depuis de nombreuse années ([17],[11]), contrairement à la dimension temporelle plus récemment pris en compte.

l'auteur lors de la phase d'édition (interface d'édition, détection des erreurs, aide à la conception, etc.) tandis que le système de présentation a pour fonction d'orchestrer l'exécution du document (affichage, déplacement, démarrage et interruption des objets, etc.). Madeus, nom donné au prototype développé au sein du projet Opéra, est un système conçu selon ce principe.

Edition de documents multimédia avec le prototype Madeus

L'approche choisie dans Madeus[31] pour la description du scénario temporel d'un document multimédia est fondée sur l'utilisation de contraintes qui expriment le placement temporel relatif des objets. L'idée sous–jacente est qu'un auteur puisse dire : "Cette vidéo A se joue avant cette autre vidéo B. Un texte C se joue en même temps que la présentation de la vidéo B. La Vidéo A démarre en même temps qu'un son D et D doit forcément être terminé lorsque C démarre". Ce qui se traduit par les contraintes suivantes:

- `Vidéo A avant Vidéo B`
- `Texte C égale Vidéo B`
- `Vidéo A démarre Audio D`
- `Audio D avant Texte C`

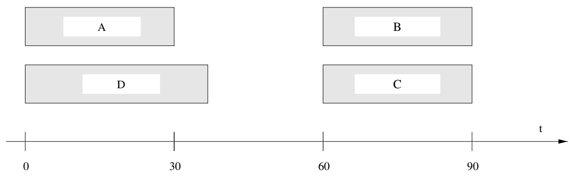


Figure I.1 : Une exécution possible du scénario précédent

La Figure I.1 présente un placement possible des objets à partir de ce scénario en supposant que leur durée respective est 30s, 30s, 30s et 40s.

La spécification du scénario avec des contraintes souples comme *avant, après, pendant* permet la définition de plusieurs placements temporels à partir d'un seul scénario. En effet, lorsque l'auteur spécifie "A avant B", il ne précise aucune valeur pour l'intervalle entre la fin de A et le début de B. C'est le système lors de la présentation du document qui effectuera le **placement temporel global** des objets en fonction des **informations locales** (contraintes entre deux objets ou groupes d'objets). Le rôle de l'auteur est donc de spécifier uniquement les informations locales, laissant au système la tâche fastidieuse de mettre à jour les informations globales.

Cette approche de spécification du scénario temporel à base de contraintes est aussi celle choisie par d'autres projets tels que ISIS [43] et HPAS [51].

```
Video STARTS Audio
Video PARMIN Audio
Video OVERLAPS (  1.20 ) BACK
BACK EQUALS FORW
BACK EQUALS ENTER
Delai2 MEETS Video2
Delai2 BEFORE (  0.50 ) logo
Delai STARTS Video2
Delai PARMIN Video2
```

Figure I.2 : Le scénario sous forme textuelle

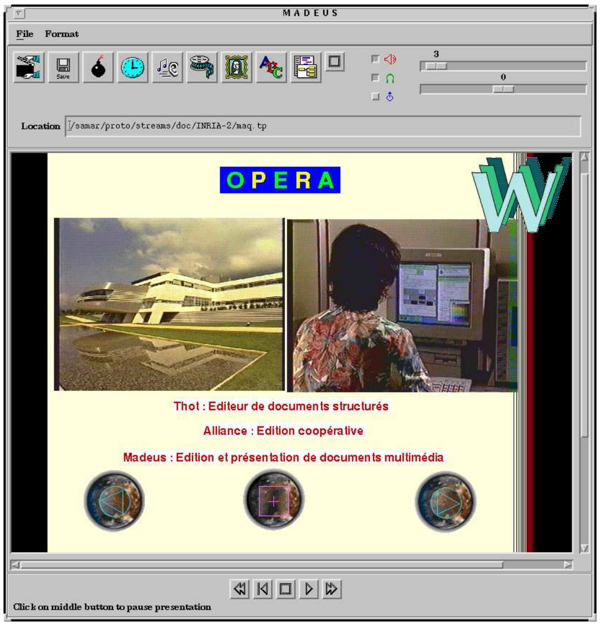


Figure I.3 : Interface de présentation de Madeus

Actuellement dans Madeus il est possible de spécifier textuellement le scénario temporel d'un document comme sur la Figure I.2. A partir de cette spécification textuelle le système est capable de présenter le document en positionnant les objets dans le temps de manière à respecter les contraintes du scénario. La Figure I.3 présente l'interface de présentation de Madeus.

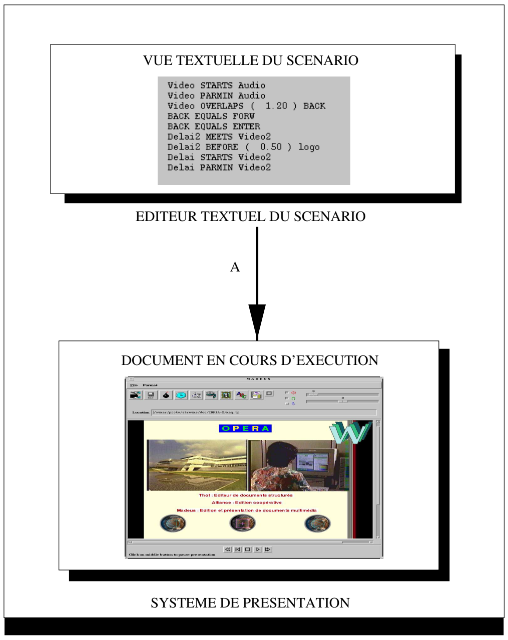


Figure I.4 : Processus de création actuel dans Madeus

Dans la Figure I.4, nous présentons le processus de création d'un document multimédia dans Madeus. L'auteur dispose du source textuel, qui est une représentation statique d'informations locales, et parallèlement, en demandant la présentation du document (flèche A de la Figure I.4), il peut voir l'un des placements temporels possibles à partir de ces informations locales. Il peut alors éventuellement modifier son scénario puis demander une nouvelle présentation du document. L'inconvénient de ce processus de création du document (édition textuelle du scénario et présentation du document) est double. Lors de la présentation du document :

- L'auteur ne dispose que d'une vue instantanée du placement temporel des objets les uns par rapport aux autres : il n'y a pas de trace du passé, ni possibilité de voir ce qui va se produire par la suite. Or ces informations de perception du scénario sont très utiles à l'auteur en phase de construction du scénario.
- Le système ne visualise qu'une des solutions de placement possibles à partir d'un scénario. Nous verrons plus tard que le calcul de cette solution est partiellement dynamique car il peut dépendre de données connues uniquement au cours de la présentation. Par conséquent, deux présentations successives du même document peuvent conduire à des placements différents d'objets. Il serait donc intéressant que l'auteur,

lors de la phase d'édition, puisse visualiser l'ensemble des solutions qui peuvent se produire au moment de la présentation de son document.

2 Objectifs du stage

Afin de faciliter la tâche de l'auteur lors de la construction du document, nous nous sommes intéressés dans ce stage à la **visualisation de l'ensemble des placements temporels possibles des objets** à partir d'un scénario.

Par la suite nous appellerons la vue ainsi obtenue, **la vue graphique du scénario.** Il ne s'agit donc pas simplement d'offrir un équivalent graphique à la forme textuelle du scénario, comme cela est fait dans les autres systèmes fondés sur une approche à base de contraintes, mais de restituer à l'auteur des informations globales déduites du scénario. En reprenant l'exemple précédent, on cherche à concevoir une vue qui permette à l'auteur de visualiser les quatre objets A, B, C et D sur des axes temporels avec la possibilité de déplacer ces objets sur ces axes, en fonction des contraintes qui les régissent, de manière à pouvoir appréhender les différents placements possibles.

Ainsi, le processus de création d'un document multimédia au sein de Madeus (Figure I.4) doit être étendu pour intégrer la vue graphique du scénario temporel que nous venons de présenter (Figure I.5). L'auteur pourra s'aider du système de visualisation (flèche B de la Figure I.5) pour affiner son scénario avant de le présenter (flèche A de la Figure I.5) et éventuellement le modifier. Il pourra s'aider de la vue graphique pour comprendre cette exécution (flèche C de la Figure I.5).

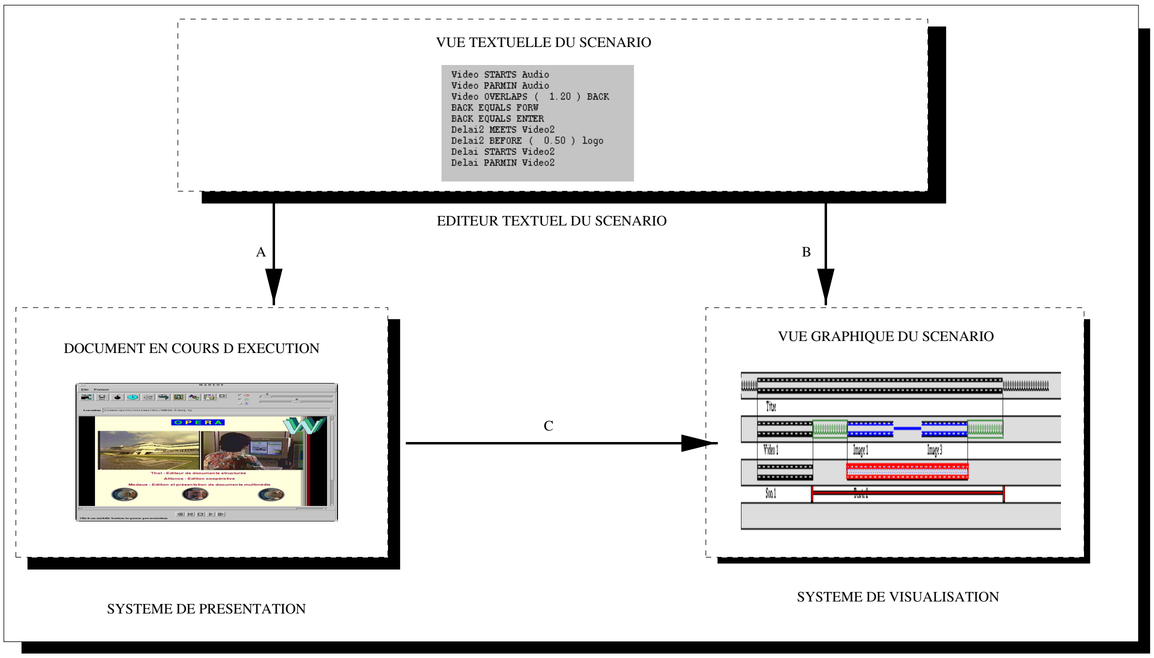


Figure I.5 : Processus de création souhaité dans Madeus

Cet effort porté sur la visualisation du scénario temporel d'un document multimédia ne constitue qu'une première étape dans le développement d'une interface graphique interactive d'édition du scénario au sein de Madeus. L'objectif à long terme consiste en effet à offrir à l'auteur un moyen de spécifier des scénarios aussi complexes que possible par l'intermédiaire d'une interface graphique aussi conviviale que possible.

3 Démarche de travail

Le travail qu'il m'a été demandé de réaliser consistait à identifier les difficultés liées à la réalisation de cette vue graphique du scénario au sein de Madeus et à proposer une première solution de visualisation.

Ce travail soulève clairement une problématique de visualisation d'information dans un domaine particulier qui est l'édition de documents multimédia. Nous avons donc jugé utile, afin d'offrir une réponse la plus complète possible, d'effectuer une synthèse des travaux intéressants menés dans ces deux domaines. Le chapitre I présente différents systèmes d'édition et de présentation de documents multimédia et le chapitre II, lui, présentera différentes techniques issues du domaine de la visualisation d'information qui seront utiles à la résolution de notre problème.

Au regard de ces synthèses, nous présentons dans le chapitre III les besoins et les difficultés soulevés par notre problématique de visualisation au sein de Madeus. Nous présentons ensuite la solution que nous proposons pour satisfaire certains de ces besoins, ainsi que les problèmes algorithmiques soulevés par son implémentation.

Chapitre I

Edition et visualisation du scénario temporel d'un document multimédia

I.1 Introduction

L'édition de documents multimédia pose de nouveaux problèmes par rapport à l'édition de documents "classiques" (ne comportant que du texte et des figures). Par exemple on ne manipule plus directement des plans ou des pages mais un document continu (sans nécessairement des ruptures de plans). On manipule aussi des données dynamiques comme le son, la vidéo, etc.

Depuis les premières études dans ce domaine [22], on a assisté à une évolution de l'approche de l'édition de documents multimédia. Une classification de cette approche a été proposée par D. Bulterman [6]. Il classe ces approches en trois générations successives.

La première génération de systèmes d'édition dépendait fortement de l'environnement et des capacités audio et vidéo de la machine hôte. De plus, la description des documents, pour profiter pleinement des capacités de la machine, conduisait à des documents non portables [24]. La seconde génération de systèmes d'édition a cherché à résoudre ce problème de portabilité des documents. Le modèle de documents, lui, restait un modèle page par page comme pour les documents textuels classiques. La troisième génération a confirmé l'intérêt pour la portabilité des documents et a vu l'apparition de documents continus.

C'est à la troisième génération d'éditeurs qu'appartiennent ceux que nous présentons dans ce chapitre. Ces éditeurs sont fondés sur différentes approches (parfois cumulées) pour la description du scénario:

- utilisation d'un langage impératif ;
- placement des objets sur des axes temporels absolus ;

- description d'un support opérationnel (graphes, arbres, réseaux de Pétri);
- utilisation de contraintes.

Chacune de ces approches soulève des problèmes de visualisation du scénario qui lui est propre. Nous avons choisi de présenter les trois premières approches par l'intermédiaire d'un représentant clé (de par sa notoriété commerciale ou son impact dans le domaine de la recherche). La dernière approche (scénario à base de contraintes) fait l'objet d'une étude plus détaillée car c'est à celle–ci qu'appartient Madeus, le système d'édition/présentation sujet de notre étude. Pour chaque présentation d'un système, nous donnons les principes de construction d'un scénario et nous analysons comment le problème de visualisation du scénario se pose et est résolu dans ce cas particulier. Les avantages et inconvénients de chaque approche sur le plan de la puissance d'expression ne sont pas discutés dans ce mémoire car nous avons pris le parti de ne considérer que le problème de visualisation.

Avant de commencer cette présentation nous avons choisi de présenter différents critères qui vont nous permettre d'analyser les vues graphiques de scénario proposées par les différents systèmes. Ces critères nous seront aussi utiles tout au long du rapport.

I.2 Les critères ergonomiques pris en compte

Pour étudier les différents systèmes, nous nous sommes basés sur les critères définis dans [19]. Parmi ceux–ci nous en avons retenu cinq qui semblent pertinents à notre sujet. Tous ne vont pas apparaître dans les analyses des systèmes qui suivent, cependant ils seront utiles lors de la présentation de notre solution de visualisation (Chapitre III).

1. **Observabilité**. Elle définit la capacité pour l'utilisateur d'évaluer (de déterminer) l'état interne du système par l'intermédiaire de commandes passives (zoom, défilement...). Par exemple, si l'état interne du système est défini par un ensemble de variables, il faut donner la possibilité à l'utilisateur de visualiser les valeurs de celles–ci. L'important est donc que l'utilisateur **puisse percevoir** l'état interne du système.
2. **Insistance**. Elle définit la capacité du système à forcer la perception de l'état du système. En reprenant le cas précédent, une solution pour satisfaire le critère d'insistance et de forcer l'affichage des variables.

L'important est donc que l'utilisateur **doive percevoir** l'état interne du système.

3. **Honnêteté**. Elle définit la capacité du système à rendre observable son état sous une forme qui engendre une **interprétation correcte** de la part de l'utilisateur. Si nous reprenons le cas où l'état du système est défini par un ensemble de variables, cela revient à montrer une représentation de ces variables qui permet à l'utilisateur de ne pas faire de fausses interprétations.
4. **Multiplicité de la représentation**. Elle définit la capacité du système à fournir plusieurs représentations d'un même concept. C'est un moyen de satisfaire le critère d'insistance.
5. **Prévisibilité**. Elle définit la capacité pour l'utilisateur de percevoir pour un état donné l'effet d'une action.

Nous avons aussi défini deux critères supplémentaires qui nous semblent pertinents dans notre contexte :

1. **Cohérence entre les vues**. Elle définit la capacité du système à maintenir la cohérence entre les différentes vues d'un concept. C'est un raffinement des critères d'honnêteté et de multiplicité de la représentation.
2. **Passage à l'échelle**. Il définit la capacité du système à maintenir une vue interprétable d'un concept quelle que soit l'évolution de sa taille. C'est un raffinement du critère d'observabilité.

En plus de ces critères de visualisation, si l'on voulait offrir des fonctions d'édition directe du scénario il faudrait considerer des critères tels que :

- **Atteignabilité**. Elle définit la capacité du système à permettre à l'utilisateur d'atteindre le but désiré depuis l'état actuel du système.
- **Interaction à plusieurs fils**. Elle définit la capacité du système à permettre à l'utilisateur de réaliser plusieurs tâches de manière entrelacée.
- **Curabilité**. Elle définit la capacité pour l'utilisateur de corriger une situation non désirée.

I.3 Utilisation d'un langage impératif : l'exemple d'IconAuthor

IconAuthor [25] est un système de programmation qui permet de réaliser des présentations multimédia. Contrairement à d'autres langages de scripts tels que Lingo [14] ou ToolBook [35], IconAuthor a la particularité d'offrir à ses utilisateurs une interface graphique de son langage de scripts, fondée sur l'utilisation d'icônes (Figure 1.1). Chaque icône représente une fonction ou une instruction de contrôle qui peut être utilisée par l'auteur. IconAuthor fait partie des langages de programmation visuels.

Un langage de programmation visuel est un langage dont le lexique est constitué d'éléments graphiques au lieu des éléments terminaux alphabétiques. De plus sa syntaxe, en plus d'exprimer des règles de composition lexicale, traduit des relations topologiques et géométriques. Une classification des langages de programmation faite par Myers [36] permet de classer les langages de programmations visuels en quatre catégories :

- les diagrammes ;
- les icônes ;
- les formes tabulaires ;
- les systèmes démonstratifs.

De manière évidente IconAuthor appartient à la deuxième classe ainsi définie.

La construction du scénario temporel se fait par l'écriture d'un programme impératif qui contient à la fois des primitives classiques (conditionnelle, boucle,...) et des instructions plus spécifiques aux documents multimédia (scrolling text, display picture,...) (Figure 1.1).

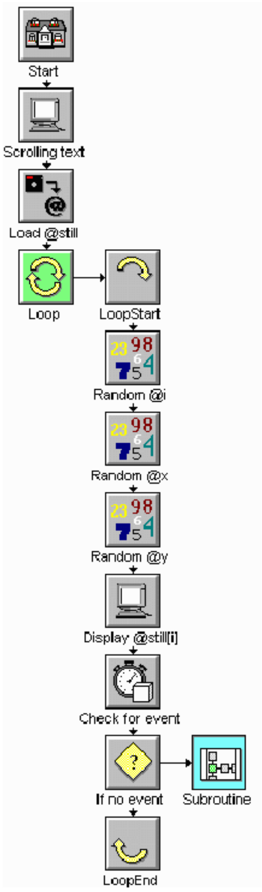


Figure 1.1 : L'édition d'IconAuthor

Le problème de visualisation du scénario tel qu'il est posé dans IconAuthor ressemble plus à un problème de visualisation des exécutions possibles d'un programme qu'à un problème de placement des objets sur un axe temporel. On voit qu'il est difficile d'imaginer le document associé à un script. Sans vouloir être exhaustif, les critiques qui nous semblent les plus importantes sont les suivantes:

- Avec cette approche, la structure du document et ses relations temporelles se dispersent au sein de l'algorithmique des scripts. L'observabilité de la structure temporelle du document est donc relativement faible.
- Il n'y a pas de lien entre le scénario et la présentation du document. La cohérence entre les vues est donc inexistante.
- Le passage à l'échelle pose un problème. En effet plus les documents deviennent complexes, plus il est difficile de les appréhender.

I.4 Utilisation d'un axe temporel absolu: l'exemple de Director

Macromedia Director [14] est aujourd'hui le système commercial le plus utilisé pour la réalisation de présentations multimédia.

Director n'est pas exclusivement un système directement fait pour créer des documents multimédia, il est aussi un système de création d'applications manipulant des données multimédia.

I.4.1 Principe de construction du scénario temporel

La construction d'un document à l'aide de Director s'effectue en trois étapes:

- Définition des objets de base (ou réutilisation de ceux existants).
- Placement des objets "à la main" dans le time–line, c'est–à–dire que l'auteur doit explicitement indiquer les moments de présence d'un objet à l'écran. C'est un placement absolu des objets sur un axe temporel qui se fait sur une vue comme celle montrée Figure 1.2. L'auteur peut naviguer dans son document à l'aide de "scroll–bar" (barre de défilement).

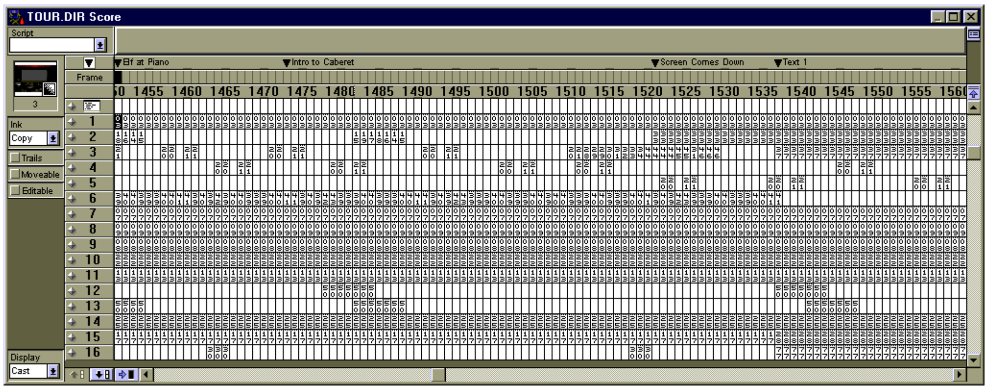


Figure 1.2 : Interface d'édition du scénario temporel de Director

- Description des interactions entre les objets et des interactions utilisateurs à l'aide d'un langage de programmation spécialisé nommé Lingo (Figure 1.3). Ces scripts peuvent, par exemple, servir à introduire des boutons d'interaction pour permettre au lecteur de naviguer dans le document. Pour cela, l'auteur dispose d'une piste temporelle spéciale sur laquelle est définie la période d'activation d'un bouton. Si le bouton est activé, un script est lancé. Cela peut avoir comme effet de stopper les éléments courants du document et de se placer à un autre point temporel. Pour identifier ces points temporels l'auteur dispose de marqueurs en forme de ↓ qu'il peut disposer au dessus de l'axe temporel, comme montré dans la Figure 1.2. Il peut alors associer des étiquettes à ces marques.

```
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" ....
```

Figure 1.3 : Description d'un document au moyen de scripts (Lingo)

I.4.2 L'interface de visualisation et d'édition du scénario dans Director

En plus de la vue "time–line", Director offre à l'auteur:

- Un "story–board", qui lui permet de voir les points clés (identifiés par des marques dans le timeline) de son scénario.

Figure 1.4 : "Story board" de Director

Dans cette vue, l'auteur a un récapitulatif de tous les points clés de son document par ordre chronologique. Le système lui permet de mettre des annotations à côté.

- Une interface de contrôle de la présentation dans le document:

Figure 1.5 : Interface de contrôle de la présentation

C'est par cette interface que l'auteur peut visualiser son document avec des facilités de manipulation: avance, avance rapide, stop, ...

- Une fenêtre d'édition des scripts

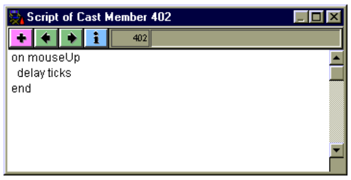


Figure 1.6 : Interface Script

- Une liste des objets graphiques présents dans le scénario.

Figure 1.7 : Liste des objets présents dans le scénario

I.4.3 Analyse de la visualisation dans Director

Le problème de visualisation du scénario dans Director se pose à la fois en termes de placement d'objets sur un axe temporel absolu, de visualisation d'un script et de liens entre ces deux vues. Director n'offre actuellement aucune vue graphique des scripts, le lien entre les scripts et l'axe temporel est fait de manière très rudimentaire par les marqueurs étiquetés. Ce n'est pas suffisant car l'auteur ne voit pas quel script fait référence à une étiquette donnée. La structure de navigation du document offerte au lecteur est donc complètement cachée à l'auteur.

Si on considère uniquement l'interface de placement temporel, l'analyse que l'on peut en faire est la suivante :

- La facilité avec laquelle on voit localement quels sont les objets qui démarrent et finissent en même temps.
- La difficulté de parcours du document et l'absence de mécanisme de zoom qui permettrait de mieux gérer les scénarios de grande taille. C'est le problème du passage à l'échelle.
- Le non respect du critère d'honnêteté car deux objets qui se succèdent sur des axes ne sont pas obligatoirement joués l'un après l'autre du fait de scripts associés qui peuvent bouleverser ces placements.
- La liaison avec les autres fenêtres est difficile (par exemple, il est malaisé de voir quels objets sont présents à un moment donné).

I.5 Approche opérationnelle : l'exemple de CMIFed

CMIFed [21] est l'un des tous premiers systèmes d'édition/présentation de documents multimédia développé par une équipe du domaine de la recherche. Il est issu d'une équipe du laboratoire CWI d'Amsterdam.

I.5.1 Construction du scénario temporel

La construction d'un scénario se fait en trois étapes:

- Définir une structure hiérarchique qui correspond à la structure du document. Choisir pour chaque nœud de la hiérarchie précédemment définie, un opérateur temporel (une approche plus complète en terme d'opérateurs a été proposée par A. Duda [13]) parmi:

  - parallèle, les deux objets commencent en même temps ;
  - séquentiel, les deux objets se jouent l'un après l'autre.

  et associer un objet de base à chaque feuille de la hiérarchie.
- Demander la construction de la "channel view" (vues des canaux) associée à cette vue hiérarchique. Dans cette vue, chaque objet est associé à un canal physique de sortie et est placé dans le temps en fonction de la durée des objets et des opérateurs de la hiérarchie. Par exemple, deux vidéo composées en séquence vont être placées l'une après l'autre sur le canal vidéo.
- Raffiner le scénario temporel par l'ajout d'arcs de synchronisation qui viennent modifier le placement temporel des objets décrits dans la vue hiérarchique. Par exemple, il est possible de décaler le démarrage de 2 objets reliés par un opérateur parallèle par l'ajout d'un arc de synchronisation entre ces deux objets. On précise alors le décalage souhaité qui peut par exemple être lié, dans le cas de deux vidéos, à la présentation d'une image particulière dans l'une des deux vidéos.

I.5.2 L'interface de visualisation et d'édition du scénario

CMIFed offre à l'auteur deux interfaces d'édition/visualisation:

- La vue hiérarchique (fenêtre de droite de la Figure 1.8):

  elle donne à l'utilisateur la structure du document. Les objets qui vont se dérouler en séquence sont positionnés les uns au dessus des autres, alors que les objets qui vont se jouer en parallèle sont positionnés horizontalement.

  Le système autorise la présence de niveaux de hiérarchie vides (pour faciliter la phase d'édition du scénario) et permet de grouper les objets.
- La "channel view" (fenêtre de gauche de la Figure 1.8) :

  c'est un "time–line" modifié qui montre les informations temporelles par un axe absolu, l'usage des ressources logiques et les arcs de synchronisation.

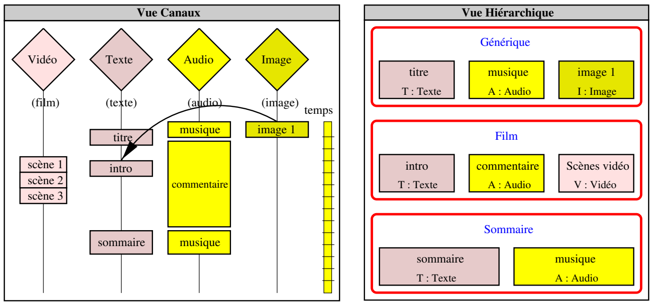


Figure 1.8 : CMIFed

I.5.3 Analyse de l'interface de visualisation

Plusieurs critiques peuvent être faites sur ce mode de visualisation:

- Complexité de la phase d'interprétation de la "channel view", lorsque le nombre d'arcs devient important (problème du passage à l'échelle). On peut aussi regretter que la décomposition hiérarchique n'est pas utilisée pour réduire la complexité de la vue temporelle (la "channel view").
- Problème d'honnêteté, l'axe de temps absolu est bouleversé par l'usage des arcs de synchronisation : ceux–ci remettent en cause les placements des objets et ces modifications ne sont pas répercutées dans les différentes vues. Par exemple, sur la Figure 1.8 deux objets sont liés par un arc de synchronisation qui peut exprimer que l'introduction doit être présentée 15 secondes après le début de la présentation de l'image. Cette information ne modifie pas le placement des objets construit à partir de la vue hiérarchique par rapport à l'axe temporel absolu. Elle n'est pas non plus répercutée dans la vue hiérarchique, ce qui fait que les deux fenêtres ne contiennent pas la même information. De plus, la vue hiérarchique nous dit que les trois composants du nœud film sont en parallèle, la "channel view", elle, nous montre vaguement que ces objets ont un instant en commun.
- Absence de mécanismes de liaisons entre les vues, qui rend la tâche de liaison complexe pour l'auteur. Cette difficulté est accentuée par le fait que l'on perd de l'information entre les deux vues (ordonnancement des objets composites).

I.6 Approches fondées sur l'utilisation de contraintes

Une idée commune aux approches fondées sur l'utilisation de contraintes est de permettre à l'auteur de ne pas fixer précisément les durées de chaque objet, mais de laisser le système les calculer en fonction des contraintes exprimées par l'auteur.

Chaque objet multimédia est caractérisé par trois valeurs:

- durée minimum ;
- durée idéale ;
- durée maximum.

C'est un moyen de tenir compte de la flexibilité des objets: leur capacité à être présentés avec des durées variables. Par exemple, un auteur qui tient à présenter une image durant la présentation d'une vidéo qui doit elle même être présentée en même temps qu'une image, déclare dans son scénario les contraintes de placement existantes entre les objets. Le système calcule le temps de présentation de chacun de manière à respecter les contraintes. On parle de "formatage temporel" par comparaison au formatage spatial.

Un des problèmes qui apparaît de manière évidente dans ces approches est la notion de cohérence du scénario temporel qui caractérise le fait qu'il existe au moins une solution de placement. Ce problème ne fait pas partie des préoccupations du stage présenté dans ce rapport.

Les contraintes portent soit sur les instants de début et fin, c'est le cas de FireFly ci–dessous présenté, soit sur les objets, c'est le cas d'ISIS (section I.6.2) et de Madeus présenté dans la section I.6.3.

I.6.1 Contraintes sur les instants : l'exemple de FireFly

FireFly [5] est un système de création, d'édition et de présentation de documents multimédia. C'est un prototype élaboré en même temps que CMIFed et issu d'un laboratoire de recherche américain. Le système permet à l'auteur de synchroniser des données multimédia et de spécifier des points d'interaction dans le document avec l'utilisateur. Le système Firefly permet de manipuler différents types de données multimédia comme le texte, le son, l'image et la vidéo mais aussi des liens hypermédia et des programmes (instructions de présentation).

Des contraintes temporelles sont associées entre les instants de début et de fin des objets. Les contraintes sont de deux types, ce sont soit des contraintes de simultanéité soit des contraintes de précédence. Dans les deux cas les relations peuvent être spécifiées de manière souple en permettant des marges maximales et minimales sur les délais de synchronisation.

L'édition du document multimédia se fait de manière progressive par adjonction incrémentale:

- d'objets multimédia ;
- de contraintes de synchronisation des événements ;
- d'associations d'événements internes aux objets spécifiés.

Dans la vue ci–dessous, qui sert à la fois d'interface de visualisation et d'édition, le système offre une visualisation du graphe de contraintes. Pour cela, les objets sont représentés par leur instants de début et fin. Les contraintes sont représentées par des arcs entre des instants.

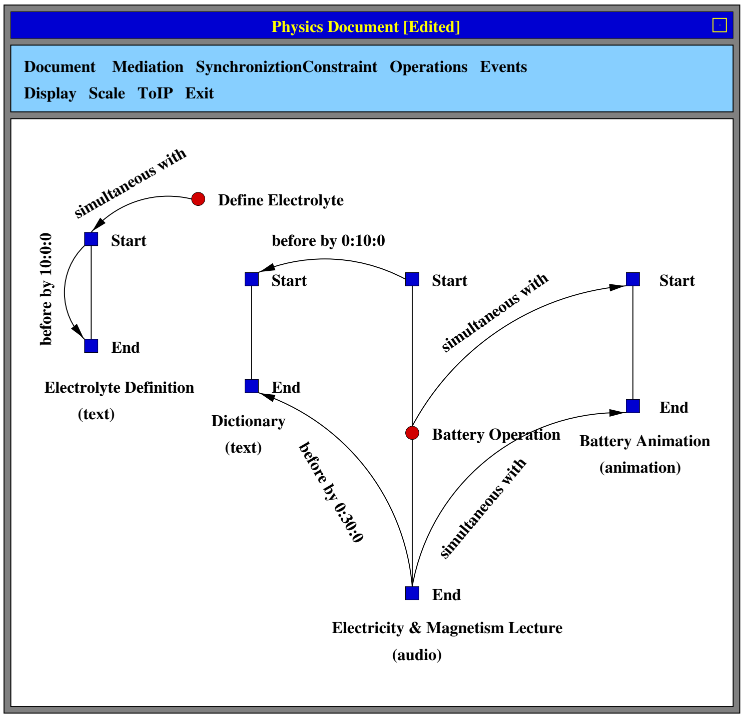


Figure 1.9 : Le système FireFly

On peut remarquer que cette vue tend à induire une mauvaise compréhension du scénario. La représentation choisie induit un certain nombre de problèmes :

- Honnêteté : la disposition du graphe induit implicitement un axe temporel de haut en bas (du fait des placements des instants de début et de fin). Dans l'exemple de la figure Figure 1.9 on a tendance à mal interpréter l'enchaînement de certains événements. Par exemple, on a tendance à croire que la "Battery Operation" et le "Dictionnary" commencent en même temps, alors qu'il y a un décalage de 10s entre les deux.

- Observabilité : à chaque objet sont associées des durées minimale, maximale et optimale, or ces informations n'apparaissent nul part.
- Passage à l'échelle : on peut aussi s'interroger sur la facilité de compréhension de cette vue lorsque le scénario devient un peu plus complexe. Ce problème est d'autant plus crucial que FireFly ne propose pas de moyen de décomposer le document de manière hiérarchique.

I.6.2 Contraintes sur les intervalles: l'exemple d'ISIS

Le système Isis [43] est un prototype de recherche développé au sein du laboratoire IBM RESEARCH CENTER.

La création d'un document multimédia est constituée de cycles de phases de validation et de composition. On relie les objets multimédia en mettant des contraintes entre eux. On obtient ainsi un ensemble de solutions.

Les contraintes pouvant être placées entre les objets sont :

- co−start(m_1 , m_2, ... , m_n): les objets m_i démarrent en même temps ;
- co−end(m_1 , m_2, ... , m_n): les objets m_i finissent en même temps;
- co−meet(m_1 , m_2, ... , m_n): les objets m_i se succèdent ;
- co−occur(m_1 , m_2, ... , m_n): les objets m_i se déroulent en même temps.

L'interface de visualisation d'ISIS (Figure 1.12) qui sert aussi d'interface d'édition permet de visualiser la flexibilité de certains objets (métaphore du ressort) et toutes les relations entre les objets. En effet, chaque relation se traduit directement par une structure graphique (Figure 1.10).

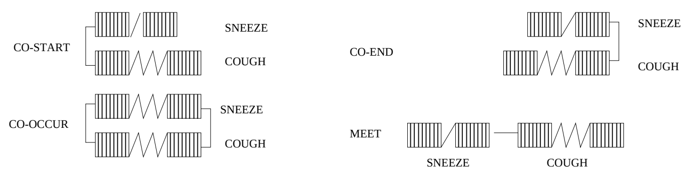


Figure 1.10 : Représentation des opérateurs dans ISIS

Co–start (Bear, Delay)
Meet (Delay, Baloo)
Co–end (Pond, Baloo)
Co–start (Pond, Waltz)
Co–end (Waltz, Bear)

NOM	DUREE MIN	DUREE OPT	DUREE MAX	DUREE EFF
Baloo	3.0	3.0	15.0	3.0
Pond	7.0	9.9	14.9	9.9
Waltz	1.0	3.0	5.0	1.0
Delay	10.0	10.0	10.0	10.0
Bear	3.0	3.0	20.0	4.1

Figure 1.11 : Un exemple de scénario simple

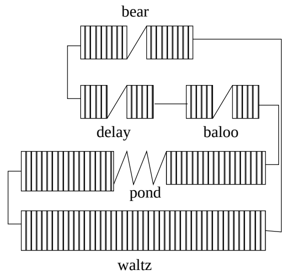


Figure 1.12 : Représentation graphique du scénario (Figure 1.11)

Dans la Figure 1.12 on visualise la solution qui satisfait au mieux les durées optimales de chaque objet. Ici, les durées affectées à chaque objet sont données dans le tableau de la Figure 1.11.

Le modèle de représentation choisi, qui essaye de faire cohabiter des informations temporelles avec des informations sur les contraintes entres les objets peut entraîner une compréhension inexacte du scénario de la part de l'auteur. Plusieurs problèmes sont mis en avant par cette représentation :

- Honnêteté : en effet, dans l'exemple de la Figure 1.12, l'auteur peut être amené à faire mentalement le lien entre la taille des objets et leur durée, ce qui n'est pas le cas, cela se voit bien sur la figure. Ainsi "Baloo" qui a une durée trois fois plus grande que celle de "Delay" a la même taille que lui sur la vue graphique. De plus, malgré le lien explicite entre "Bear" et "Waltz" on n'a pas l'impression que ces deux objets terminent en même temps. De la même façon, la représentation explicite choisie pour le "meet" fausse l'impression que l'on a du scénario (ie: les deux objets ne sont pas collés l'un derrière l'autre).

- Le mode de représentation, qui permettrait de montrer plus d'informations temporelles que ce qui est fait (par exemple dans ce cas toutes les informations temporelles sont connues) se limite en fait à être une représentation graphique du graphe de contraintes. De plus aucune manipulation n'est possible.
- Passage à l'échelle : on peut aussi ce demander ce qu'il va se passer lorsque le scénario va devenir complexe, et comment l'auteur fera pour comprendre les enchaînements de son document.

I.6.3 Contraintes d'intervalles: l'exemple de Madeus

Madeus est un prototype développé au sein du projet opéra [31]. Les particularités de l'approche choisie dans Madeus sont:

- La différenciation de deux types d'objets:
  - Objets contrôlables: leur durée de présentation peut être librement fixée par le système entre deux bornes. C'est le cas par exemple des objets de type texte. On peut leur associer une borne minimale pour assurer la lisibilité de leur contenu et une borne maximale infinie (ce sont les objets flexibles dans Isis).
  - Objets incontrôlables: leur durée de présentation ne sera connue qu'à l'exécution du scénario car: soit elle dépend de facteurs dynamiques comme la charge de la machine ou du réseau (cas d'une vidéo), soit elle est par nature indéterministe comme une interaction utilisateur. On peut dans le premier cas donner des bornes minimales et maximales sur la durée en considérant des critères de qualité de présentation des objets.
- L'utilisation des opérateurs d'Allen ([3]) comme jeu de contraintes augmentés de trois opérateurs supplémentaires (Figure 1.14). Allen a défini l'ensemble de toutes les positions relatives que peuvent avoir deux intervalles placés sur un axe temporel. Nous en présentons sept dans la Figure 1.13, les six autres s'obtiennent en prenant les placements symétriques (le symétrique de l'opérateur "equal" est lui–même). La sémantique intuitive de ces opérateurs est donnée dans le tableau de la Figure 1.14. Madeus utilise les opérateurs d'Allen comme autant de contraintes que souhaite voir respecter l'auteur du scénario. Il peut en plus quantifier les délais libres introduits par certains opérateurs (before, during, overlpas qui deviennent before(t), during(t), overlaps(t)) et

utiliser une des trois contraintes supplémentaires : parmin, parmax, parmaster, présentées à présent:

– A parmax B, qui est la disjonction de A start B et B start A ;
– A parmin B, qui exprime l'interruption de l'objet le plus long par l'objet le plus court ;
– A parmaster$_{(A)}$ B, qui est équivalent à A Parmin B si A est plus court que B, ou à B start A si A est plus long que B. Par conséquent la durée de B est conditionnée par celle de A qui a été désigné comme le maître de la mise en parallèle.

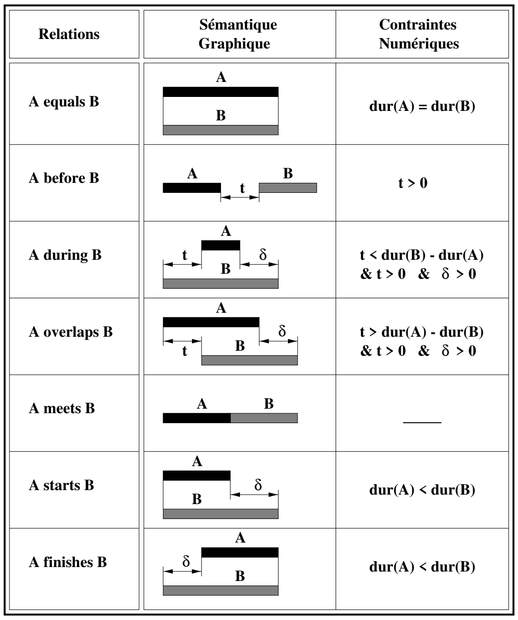


Figure 1.13 : Les opérateurs d'Allen

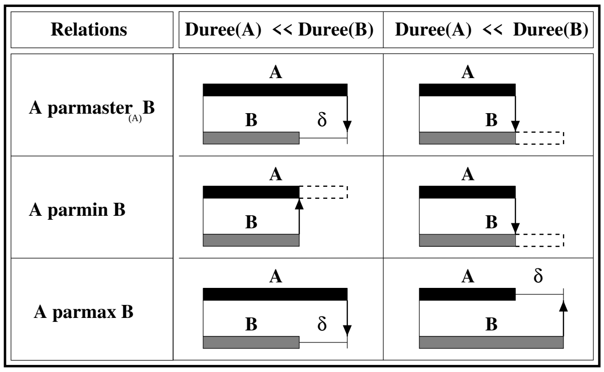


Figure 1.14 : Les opérateurs supplémentaires

- La définition d'une structure hiérarchique du document qui n'est pas comme dans CMIFed le support de la composition temporelle (ie: les nœud de l'arbre ne portent pas les opérateurs temporels). Par exemple, sur la hiérarchie de la Figure 1.15, il est possible de placer des opérateurs entre les 3 scènes ou entre les objets Son 3, Video 4, et le groupe composite, mais il est interdit de placer directement des contraintes entre Photo 1 et Son 3 ou entre Son 1 et Son 2.

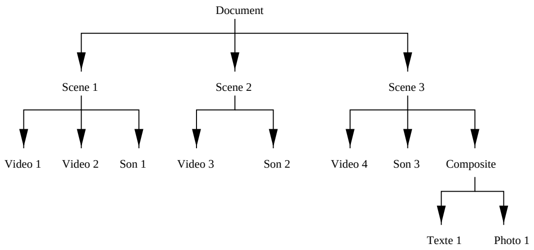


Figure 1.15 : Exemple de hiérarchie d'un document multimédia

La création d'un document multimédia se fait par la création de la structure hiérarchique et la définition de contraintes entre les objets. A tout moment l'auteur peut ajouter un objet dans la hiérarchie et/ou ajouter des contraintes.

- La modélisation des liens hypertextes par des attributs associés aux objets du document multimédia. Ces attributs contiennent le type du lien (inclusion ou hypermédia) et la destination qui doit être un objet composite. Par exemple l'objet Photo 1 peut avoir un attribut du type hypermédia qui pointe sur scène 1. Cela signifie que lors de la présentation du document, le texte 1 sera une zone activable : si on clique sur texte 1 , la scène 3 est arrêtée et la scène 1 est lancée.

  C'est liens peuvent être de différentes natures :

  – Inclusion, dans ce cas lorsque le lien est activé, un objet (qui peut être un objet composite) est inséré dans le document en cours.
  – Référence, dans ce cas lorsque le lien est activé le document en cours est stoppé, et on présente l'objet désigné par le lien. Cet objet peut être de deux types:
    + Objet du document, dans ce cas on présente un objet du document courant.
    + Document à l'extérieur du document courant, cela permet la navigation entre les documents.

L'interface de visualisation et d'édition du scénario

Comme nous l'avons dit dans l'introduction, actuellement seulement deux vues sont offertes à l'auteur. La première est celle dans laquelle s'effectue la présentation du document (Figure 1.16), la seconde n'est qu'une représentation textuelle du scénario Figure 1.17.

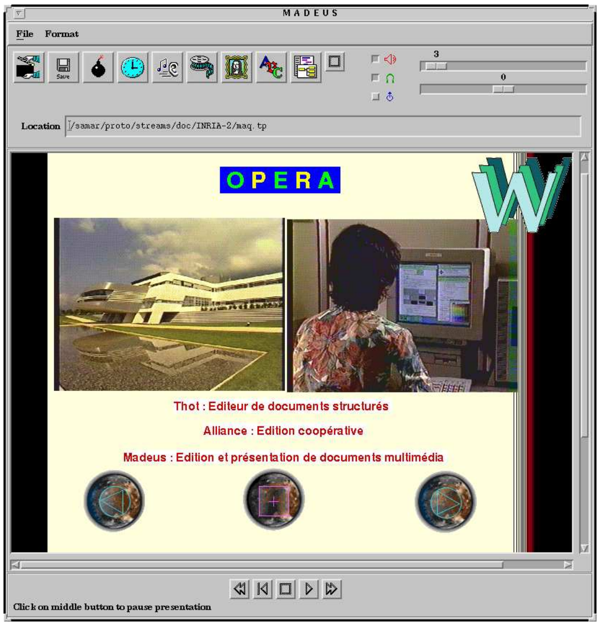


Figure 1.16 : Vue de présentation

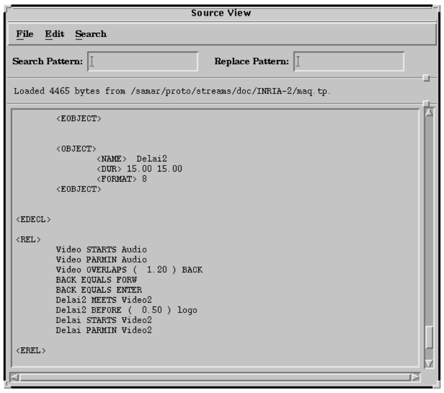


Figure 1.17 : Vue du source textuel

Les besoins liés à la visualisation du scénario seront présentés en détail dans le chapitre suivant puisqu'ils sont au cœur du sujet d'étude présenté dans ce mémoire.

I.7 Conclusion

En conclusion, nous pouvons dire que les besoins en visualisation du scénario temporel d'un document multimédia sont propres à l'approche choisie pour exprimer le scénario temporel. L'édition à base de contraintes soulève le problème spécifique de visualisation d'un ensemble de placements possibles d'objets. Aucun des environnements fondés sur cette approche (le système HPAS[51] qui n'a pas été présenté ici n'offre, lui aussi, qu'un équivalent graphique des contraintes) n'apporte de réponse satisfaisante à ce problème. Madeus en plus de cet aspect ajoute plusieurs problèmes supplémentaires: la structuration hiérarchique, la visualisation de données incertaines, la visualisation d'un grand nombre d'opérateurs différents, etc.

Nous retenons de cette analyse les trois points suivants :

1. L'utilisation des "time–lines" (utilisés à la fois dans Director et CMIFed) est un moyen simple et efficace de montrer le placement relatif de deux objets et permet d'avoir une vue précise du document à chaque instant. Cependant, elle n'est une bonne solution que pour des petits volumes de données.
2. Le problème du passage à l'échelle doit être traité avec beaucoup d'attention car c'est une critique qui peut être faite à tous les systèmes présentés ici. Nous voyons avec CMIFed que le fait de décomposer hiérarchiquement le document ne résoud pas nécessairement ce problème.
3. La satisfaction des critères d'honnêteté et d'observabilité nous apparaît comme deux points essentiels car nous avons vu qu'une difficulté commune à tous ces systèmes est de donner à l'auteur la possibilité d'observer le véritable placement temporel qui découle de sa spécification.

Chapitre II

Etat de l'art sur la visualisation de données

II.1 Introduction

Comme nous l'avons dit précédemment, les besoins de visualisation du scénario temporel d'un document multimédia sont très liés à l'approche choisie pour exprimer le scénario.

Au sein de Madeus, le scénario temporel est fondé sur :

- l'organisation hiérarchique des données ;
- l'utilisation de contraintes de placement.

Notre objectif est d'aider l'auteur d'un document multimédia en lui offrant une vue graphique de son scénario qui lui permette de visualiser l'ensemble des placements possibles des objets. Pour cela, il nous a semblé pertinent de voir comment les problèmes de visualisation sont abordés dans le domaine de la visualisation d'informations pour les cinq aspects suivants:

- la visualisation d'informations temporelles ;
- la visualisation de données hiérarchiques ;
- la visualisation de données structurées par des contraintes ;
- la visualisation de la structure de navigation ;
- la visualisation de programmes.

De plus nous présentons en fin de chapitre une application qui manipule les trois premiers types de données, d'où un certain intérêt par rapport à notre problématique.

II.2 Visualisation d'informations temporelles

II.2.1 Les Times−lines

Les Times−lines sont le plus souvent utilisés quand on doit placer des événements parallèles par rapport à un axe temporel absolu. Les Times−lines sont utilisés dans des domaines comme le traitement du son <Bibliographie>, le montage vidéo <Bibliographie>, la gestion de projet <Bibliographie>, etc. Dans la plupart des systèmes (<Bibliographie>,<Bibliographie>,<Bibliographie>) l'auteur peut placer directement les icônes sur l'axe temporel. De plus la granularité du Time−line peut être adaptée aux besoins.

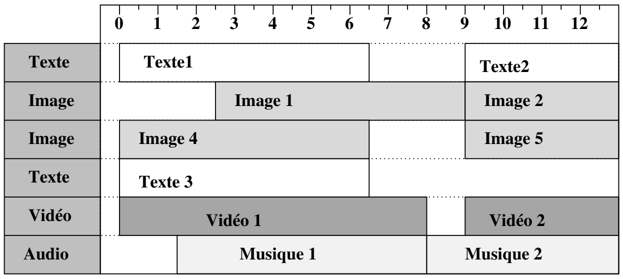


Fig. 2.1 : Un time−line

L'avantage majeur de cette approche est la facilité avec laquelle on visualise les événements qui se déroulent en parallèle. De plus, la représentation est très intuitive ce qui lui permet d'être facilement compréhensible par le lecteur.

II.2.2 Les murs en perspectives ou Perspective−Wall :

Le mur en perspective est une méthode de visualisation bien adaptée pour visualiser des grandes quantités d'informations que l'on peut ordonner selon un critère suivant l'axe des X. Cette méthode semble bien adaptée pour représenter des informations temporelles, dans ce cas le critère utilisé est le temps. L'idée à la base de ce concept est d'utiliser une vue en perspective (Fig. 2.2) pour palier les limitations physiques imposées par les tailles des fenêtres d'un système de fenêtrage classique. De cette manière, l'auteur a conscience de la totalité des informations.

Un des murs en perspective a été présenté à Xerox <Bibliographie> (Fig. 2.2) pour visualiser des informations classées chronologiquement.

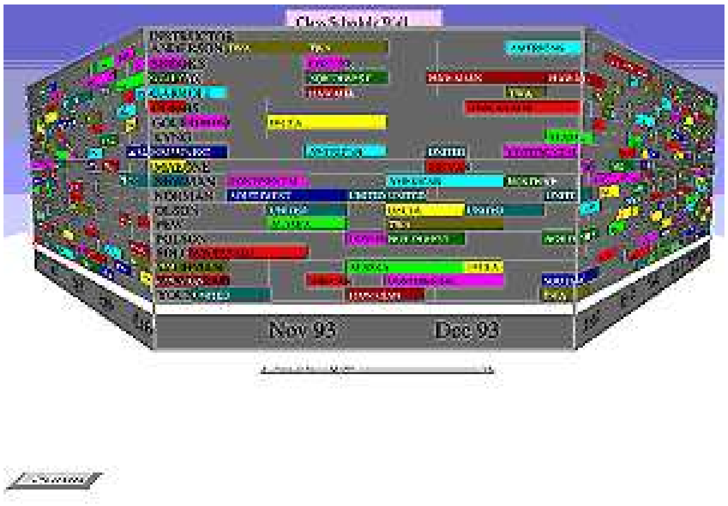


Fig. 2.2 : Un perspective wall "classique"

Le mur en perspective, dans le cas de visualisation d'informations temporelles a l'avantage de permettre la visualisation d'un plus grand volume d'informations tout en gardant au premier plan une vue proche d'un "time–line" qui permet de visualiser la période temporelle au centre de l'intérêt du lecteur.

II.3 Visualisation d'informations hiérarchiques

II.3.1 Les arbres coniques ou "cone–tree"

La première méthode de visualisation que nous allons présenter fait partie des systèmes de visualisation en 3D, les arbres coniques <Bibliographie>. Un arbre conique est une représentation tridimensionnelle d'une hiérarchie dans laquelle on associe à chaque nœud le sommet d'un cône, et on arrange ses fils autour de la base circulaire du cône. C'est ce que nous pouvons voir dans la Fig. 2.3.

Le but de cette représentation est de présenter la structure de façon telle que toute la hiérarchie ou du moins la majeure partie puisse être visualisée sans que l'utilisateur ait besoin d'utiliser les barre de défilement. Cette approche souffre d'une limitation en termes de nombre de nœuds affichables.

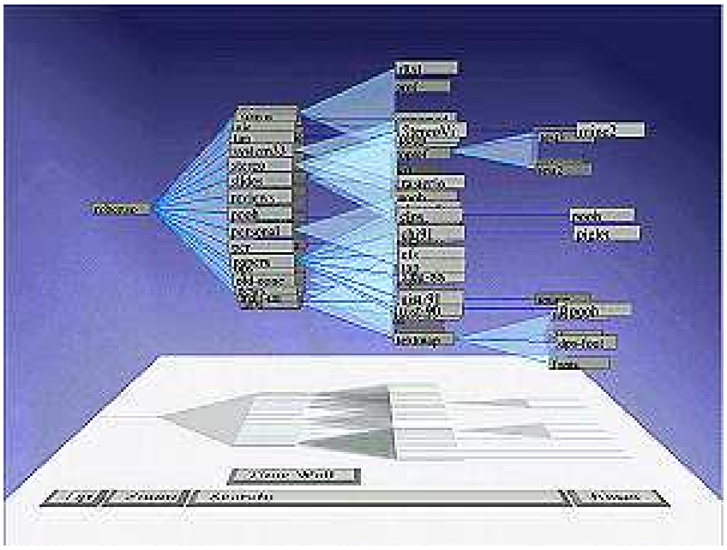


Fig. 2.3 : Un arbre conique

Dans l'exemple ci–dessus (Fig. 2.3) nous pouvons voir l'arborescence d'un système de fichiers Unix. Cette représentation nous permet d'avoir une vue globale et assez explicite de l'arborescence.

Une approche présentée par <Bibliographie> vise à étendre cette visualisation en utilisant des métaphores de couleurs et de formes pour permettre à l'utilisateur de mieux naviguer dans la structure. De plus, les concepteurs de cette approche ont proposé différentes améliorations pour faciliter le travail de compréhension de l'utilisateur et améliorer le temps de réponse lors de la manipulation de la structure. Par exemple, ils ont proposé une vue réduite de la hiérarchie qui permet de voir la forme de la hiérarchie (ou d'une partie) sans avoir les détails.

Une variante de la visualisation de données hiérarchiques par l'intermédiaire des arbres coniques (Fig. 2.4) a été proposée par Xerox <Bibliographie>. Cette approche offre des facilités de manipulation et des agréments de visualisation comme des ombres et des mouvements continus lors des rotations d'une partie de la hiérarchie.

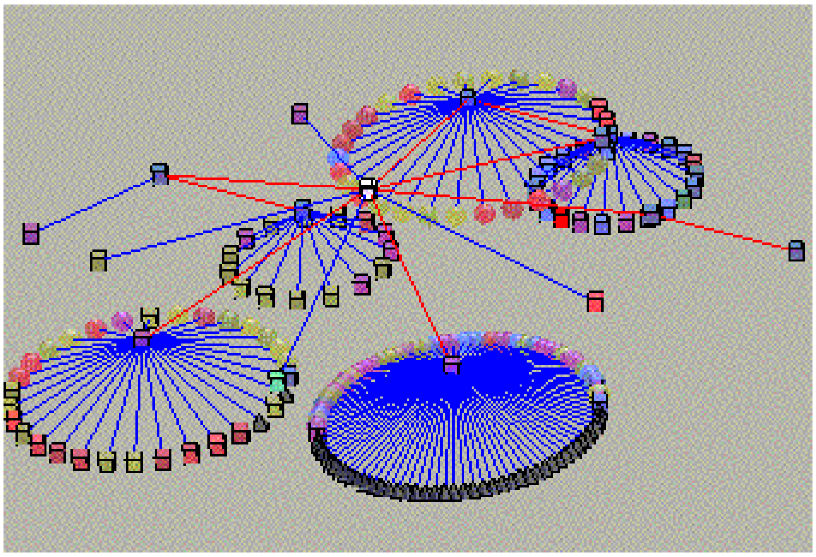


Fig. 2.4 : Un arbre conique de Xerox

L'utilisation des arbres coniques semble bien adaptée à la visualisation de grosses hiérarchies. Cependant cette représentation circulaire semble peu adaptée à notre cas où les éléments de la hiérarchie sont reliés par des informations temporelles linéaires et non pas circulaires.

II.3.2 Les arbres hyperboliques

Une approche un peu différente qui vise essentiellement à améliorer la navigation dans cette hiérarchie est celle utilisant les arbres hyperboliques (Fig. 2.5).

Les arbres hyperboliques présentés à Xerox <Bibliographie> ont pour but de faciliter la navigation à l'intérieur de la structure. Par un simple clic sur un nœud, on change le focus et on obtient une vue centralisée autour de notre centre d'intérêt. On peut rapprocher cette méthode de la vue en œil de poisson ou "fisheyes–view" <Bibliographie>.

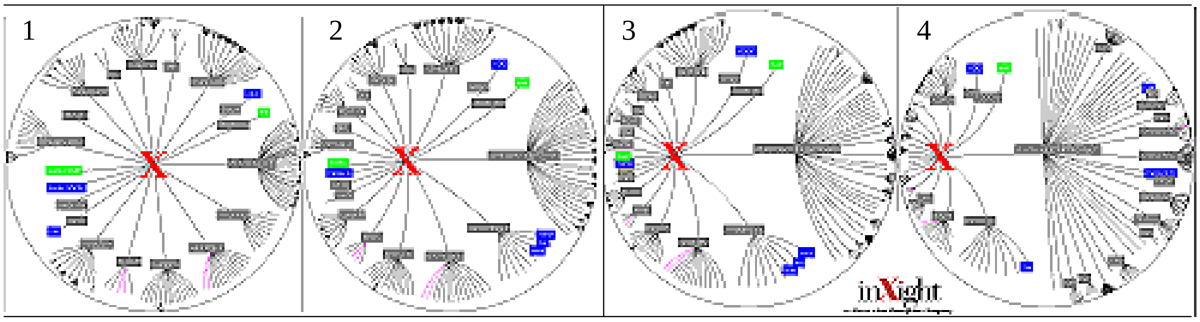


Fig. 2.5 : Un arbre hyperbolique

Dans l'exemple ci–dessus (Fig. 2.5) nous pouvons voir la visualisation de données hiérarchiques. Dans le cadre 1, nous pouvons voir la configuration initiale. L'utilisateur déplace progressivement (cadre 2, 3, 4) son centre d'intérêt de l'objet central vers le sous–arbre de droite. Plusieurs choses sont à remarquer. Par exemple, à la fin de la manipulation, l'utilisateur a complètement conscience des objets à sa droite et a des informations complémentaires par rapport à la configuration initiale. On peut aussi remarquer que les objets en bas à droite au cours du déplacement se sont rapprochés du centre de la fenêtre pour finalement s'en éloigner. La granularité de la description de ce sous–arbre a reflété ces différents changements d'états.

L'inconvénient majeur de cette approche est quelle est peu intuitive, et quelle reste relativement difficile à appréhender pour une personne non habituée.

II.3.3 Les murs en perspectives ou "Perspectives wall"

Une application des murs en perspectives, aussi présentée à Xerox <Bibliographie>, (Fig. 2.6) est utilisée pour visualiser des hiérarchies.

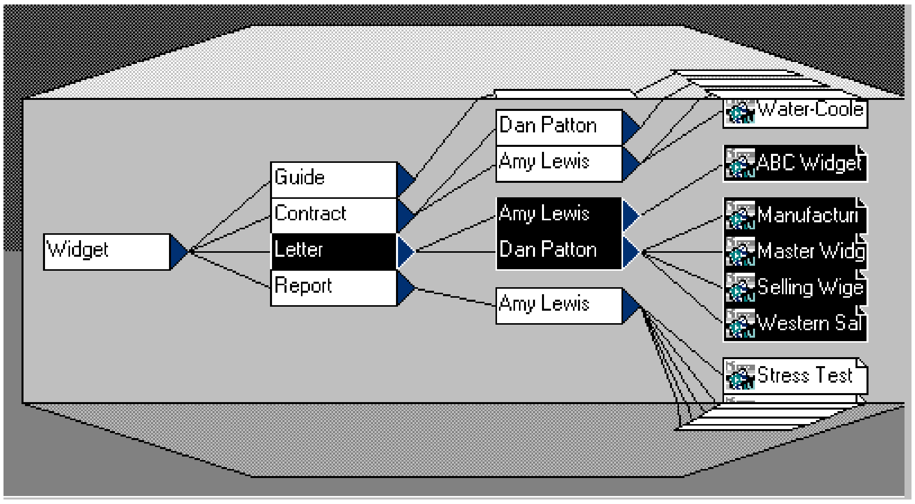


Fig. 2.6 : Un mur en perspective visualisant une hiérarchie

Ce mode de représentation permet de visualiser au centre du "perspective wall" le chemin de la hiérarchie qui centralise notre intérêt et nous permet d'avoir conscience du reste de la hiérarchie.

On peut se demander ce qu'il se passe quand on utilise des hiérarchies profondes. En effet, l'effet de perspective est utilisé pour visualiser la largeur de la hiérarchie.

II.4 Visualisation de données structurées par des contraintes

Un autre aspect intéressant par rapport à notre problématique dans la visualisation d'informations concerne la visualisation de données structurées par des contraintes. Malheureusement nous n'avons trouvé que peu de travaux qui s'intéressent à la visualisation des contraintes. Lorsque l'on structure des données par des contraintes (construction de figures <Bibliographie>, placement spatial <Bibliographie>, etc.) différents problèmes se posent:

- visualiser les contraintes, ce qui ne signifie pas seulement visualiser leur existence (ie : un graphe de contraintes <Bibliographie>) mais aussi leur effet ou leur sémantique ;
- visualiser l'espace des solutions possibles à partir d'un ensemble de contraintes ;
- visualiser pourquoi une solution est possible ou non.

Nous allons présenter deux approches de visualisation de ces contraintes dans des environnements graphiques de construction de figures géométriques. La visualisation de contraintes nécessite une interface graphique évoluée, plus sophistiquée que les outils habituels. En effet en plus des objets graphiques il faut rendre visible les relations entre les objets que l'utilisateur a déclarées. Par exemple, l'utilisateur doit pouvoir distinguer lorsque deux objets sont alignés s'ils le sont involontairement ou si c'est parce qu'il y a une contrainte d'alignement (directe ou indirecte) entre ces deux objets.

II.4.1 Contraintes graphiques avec CoDraw

CoDraw est un éditeur graphique par contraintes <Bibliographie>, il offre plusieurs vues à l'auteur :

- Le "work sheet" ou l'espace de travail, qui permet d'avoir une représentation graphique des figures géométriques en plus des contraintes.

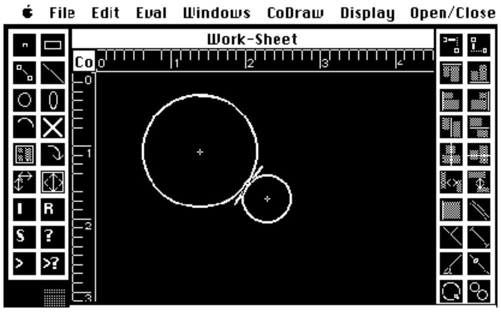


Fig. 2.7 : L'espace de travail

Sur l'exemple de la figure Fig. 2.7, le Work sheet est la partie haute de la fenêtre. On peut y voir la présence d'une contrainte de tangence entre les deux cercles.

- La structure du dessin :

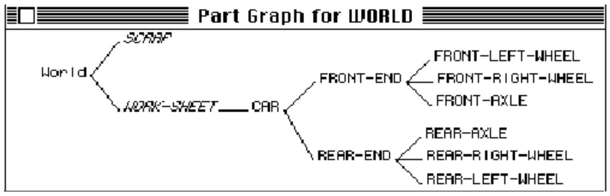


Fig. 2.8 : Structure hiérarchique du dessin

Le dessin possède une structure hiérarchique correspondant à la structure logique du dessin.

- Les contraintes héritées automatiquement de l’objet :

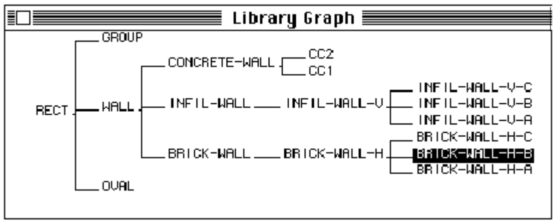


Fig. 2.9 : Le "Library graph" montre les contraires intrinsèques de l’objet

Chaque objet de part sa classe, hérite d’un certain nombre de contraintes. Un rectangle, par exemple, hérite automatiquement une contrainte: Largeur * Longueur = Aire.

- Le graphe de contraintes de chaque objet :

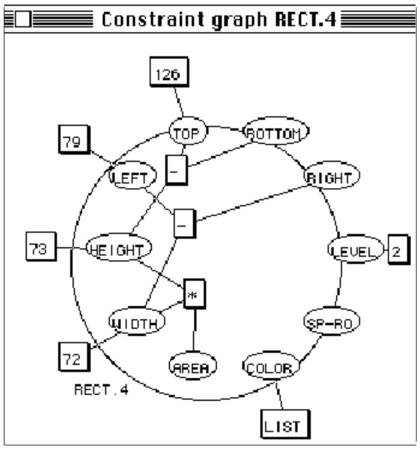


Fig. 2.10 : Graphe de contraintes pour un objet

Pour aider l’auteur à visualiser les contraintes, à chaque "clic" sur un objet une fenêtre indique l’ensemble des contraintes de cet objet.

- Le graphe de dépendances :

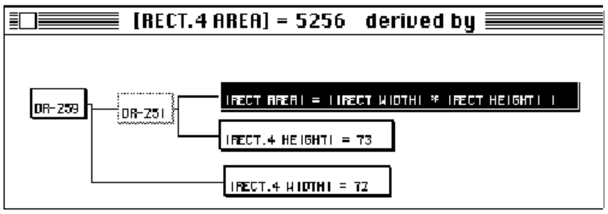


Fig. 2.11 : Le graphe de justification montre les dépendances

A la demande de l'auteur une fenêtre visualisant le graphe de justification lui permet de comprendre l'état d'une variable du système.

Nous pouvons faire plusieurs critiques sur cette interface :

- La représentation explicite des contraintes par l'opération arithmétique associée rend difficile la lecture de l'ensemble des contraintes.
- L'absence d'une vue globale des contraintes limite aussi la possibilité pour l'auteur d'appréhender le comportement global de son dessin.

II.4.2 Drawing with constraints

Le but de ce système (<Bibliographie>) est d'offrir une interface qui permet à l'auteur de visualiser et de manipuler des contraintes entre les objets tout en ayant une interface proche des interfaces graphiques habituelles.

Plusieurs efforts ont été faits pour faciliter la tâche de l'auteur:

- Lors de la phase de la sélection d'objets.

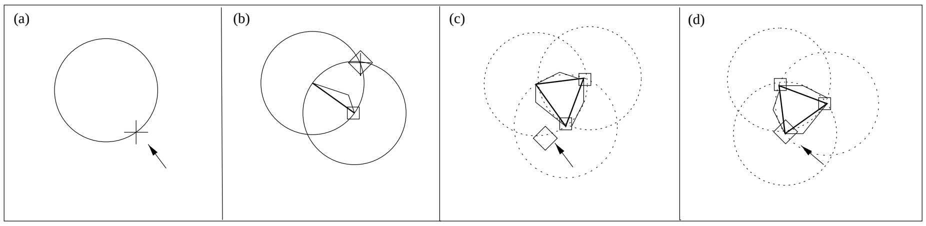


Fig. 2.12 : La désignation des points

Quand un auteur dessine un objet un mécanisme de "snap–dragging" l'aide à sélectionner les points d'un objet existant (idée du champ de gravité qui attire la souris). Il est important pour lui de pouvoir distinguer si les points qu'il sélectionne sont des points appartenant à des objets ou des points choisis au hasard. Le système permet de faire cette distinction.

Lorsque l'auteur sélectionne un point sur une figure déjà existante (l'intersection des deux cercles sur notre exemple (Fig. 2.12 (b)). Pour construire un autre élément (un nouveau cercle sur la figure) il peut soit vouloir attacher ce nouvel élément à ceux déjà construits à l'aide d'une contrainte, soit vouloir le placer ponctuellement à cet endroit sans le lier explicitement aux autres. Dans ce dernier cas, un déplacement des autres cercles n'entraînera pas de déplacement du cercle rajouté.

- Lors de la visualisation des contraintes.

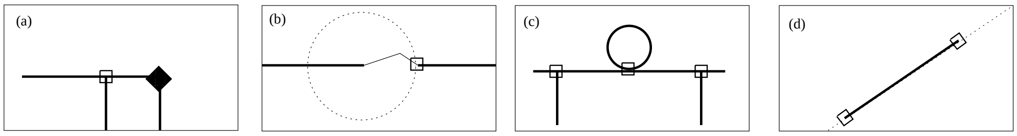


Fig. 2.13 : la visualisation des contraintes

Le système permet aussi de visualiser les contraintes. Par exemple les contraintes de distance sont représentées par un cercle en pointillés qui permet de visualiser les positions possibles, les contraintes d'orientation sont représentées par un ligne en pointillés qui permet d'imaginer les déplacements possibles de l'objet.

- Lors du déplacement des objets :

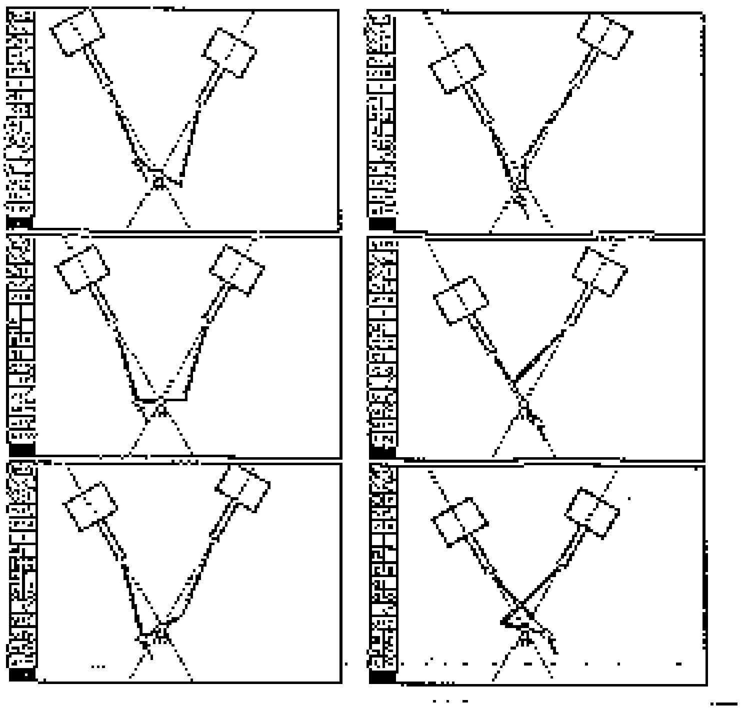


Fig. 2.14 : La manipulation de la figure et le maintien des contraintes

Lors du déplacement des objets l'affichage des contraintes est maintenu, cela permet d'anticiper en partie les déplacements. Par exemple dans la Fig. 2.14, on visualise le fait que les objets peuvent se déplacer selon des axes, mais on ne visualise pas le fait qu'il ne peuvent pas se superposer.

La visualisation choisie pose quelques problèmes. Comme nous venons de le dire les contraintes permettent de visualiser l'axe de déplacement mais ne permettent pas de visualiser l'intervalle de déplacement. Sur des figures simples cette représentation est bonne, mais on peut cependant se demander si sur des dessins complexes l'utilisateur n'aura pas besoin "d'explications" sur le comportement de sa figure.

II.5 La visualisation de la structure de navigation

La visualisation graphique de la structure de navigation d'un document, offerte au lecteur par l'intermédiaire des liens, permet une exploration rapide du document dans sa globalité et donne des points de repère indispensables pour l'exploitation des vastes espaces d'informations.

Nous présentons à présent les techniques utilisées pour visualiser ces structures de navigation. L'intérêt de cette présentation par rapport à notre problématique est double :

- Madeus intègre la notion de liens hypermédia, il offre donc une structure de navigation au lecteur que l'auteur a besoin de visualiser en phase d'édition.
- La visualisation de tels liens a développé des méthodes de visualisation de gros volumes de données qui pourront être utiles au sein de Madeus pour la visualisation du scénario dans sa globalité.

Avant de commencer cette présentation précisons que nous avons choisi de définir un document grâce à la notion de fichier : un document est une entité contenu dans un même fichier. D'autres solutions sont possibles selon que l'on inclut tout ou partie des entités pointées par les liens d'un document. Cette définition d'un document nous est utile pour distinguer les techniques utilisées pour visualiser les liens internes à un document de celles utilisées dans le cas de liens externes.

Un exemple de liens internes dans le cas de l'édition d'un mémoire est donne par les liens qui font par exemple, référence à des sections ou des figures de ce document. Un exemple de liens externes toujours dans le cadre de l'édition d'un mémoire est donnés par les liens associés aux références bibliographiques et qui permettent d'accéder aux articles cités.

En fait, nous n'avons pas trouvé de méthodes qui permettent de visualiser explicitement cette structure de liens. En général, les systèmes auteurs de tels fichiers permettent à l'auteur de se déplacer à l'intérieur de son document comme le ferait le lecteur, c'est à dire en activant les liens.

Pour ce qui est de la visualisation des liens externes il nous faut différencier le cas des structures de petite taille (cas par exemple du mémoire découpé en sous documents (chapitre)) des structures de grandes tailles (documents référençant d'autres documents, eux mêmes liés entre eux).

II.5.1 Visualisation de réseaux de liens de petite taille

Dans le cas de petites structures de navigation, fournir des moyens visuels est un moyen efficace pour visualiser et naviguer au travers de ces structures.

Bien que ce besoin existe, ces moyens visuels sont rarement utilisés dans des systèmes de conception de document multimédia. Ils sont issus d'une autre communauté : l'édition de document hypermédia. Un document hypermédia est un document hypertexte usuel dans lequel des objets peuvent être de nature "multimédia" (video, son, ...). Dans ces documents l'organisation temporelle des objets est entièrement portée par la structure de navigation offerte au lecteur. La différence entre un document multimédia et un document hypermédia et l'existence du scénario temporel.

MacWeb <Bibliographie> et AOL Press <Bibliographie> sont des systèmes hypermédia où une vue graphique est utilisée pour transmettre la connaissance sur la structure des hyper–documents.

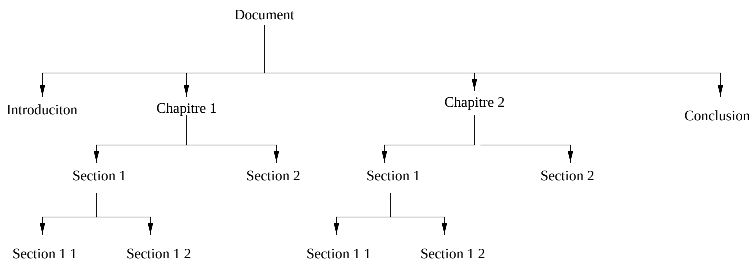


Fig. 2.15 : Hiérarchie de fichiers

AOL Press autorise un découpage hiérarchique des fichiers, à chaque niveau de la hiérarchie on peut lier les fichiers entre eux (par exemple entre Introduction et Conclusion dans la Fig. 2.16). Néanmoins ces liens ne sont pas limités à un seul niveau de la hiérarchie, ils peuvent être mis entre des fichiers de niveaux différents (par exemple en Introduction et Section 1.1 dans la Fig. 2.16). Le système n'offre pas de vue globale de ces liens, il offre seulement un vue par niveaux. Pour résoudre le problème de la visualisation des liens entre des fichiers de niveaux différents, une image grisée du fichier du niveau différent est ramenée au niveau courant de la hiérarchie (la Section 1.1 de la Fig. 2.16).

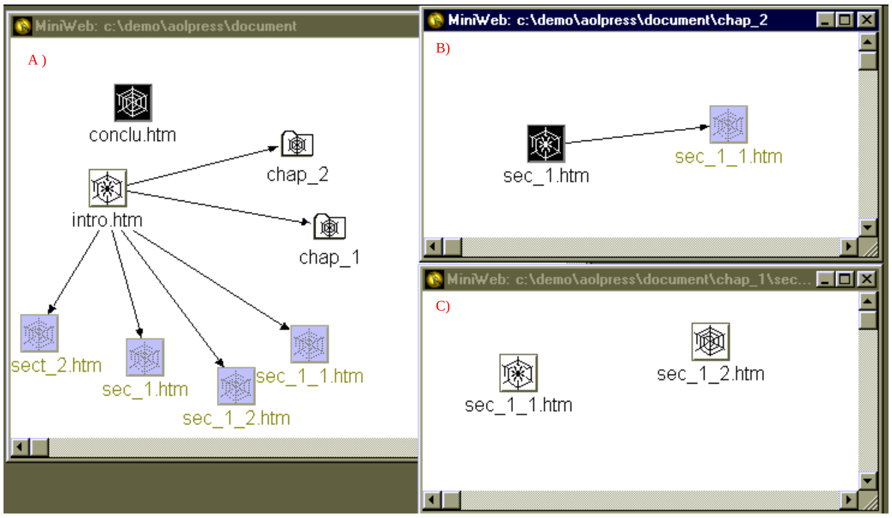


Fig. 2.16 : Un niveau de hiérarchie avec des liens

Néanmoins, on peut regretter que sur cette représentation, la seule information sur la provenance du fichier vers lequel on pointe soit la présence d'un lien vers son ancêtre présent au niveau courant (c'est par exemple le cas du lien entre introduction et Chapitre_1 qui indique l'existence d'un lien entre introduction et un fichier présent dans le chapitre 1). Cette représentation est un début dans la visualisation de la structure de navigation. Elle a plusieurs inconvénients :

- L'absence d'information sur la provenance d'un fichier lors de liens latéraux. Par exemple, lorsque l'on met un lien de la section_1 du chapitre 2, vers la section1.1 du chapitre 1, on n'a pas d'ancêtre à visualiser au niveau courant).
- L'information sur les liens devient inexploitable dès que l'on a deux liens vers des objets qui appartiennent à des branches différentes. En effet dans le cas de la Fig. 2.16, on voit que l'introduction pointe vers des fichiers du chapitre 1 et 2, et on peut se demander à quel chapitre appartient la section 1.
- On ne visualise pas les fichiers qui pointent sur un fichier donné. Par exemple, on ne visualise pas les fichiers qui font référence à la section 1.1.
- On peut mettre en doute la capacité de cette solution de visualisation à passer à l'échelle. La Fig. 2.17 permet de mettre en évidence qu'une représentation adaptée à de petit volume d'information ne s'adapte pas directement à de gros volume.

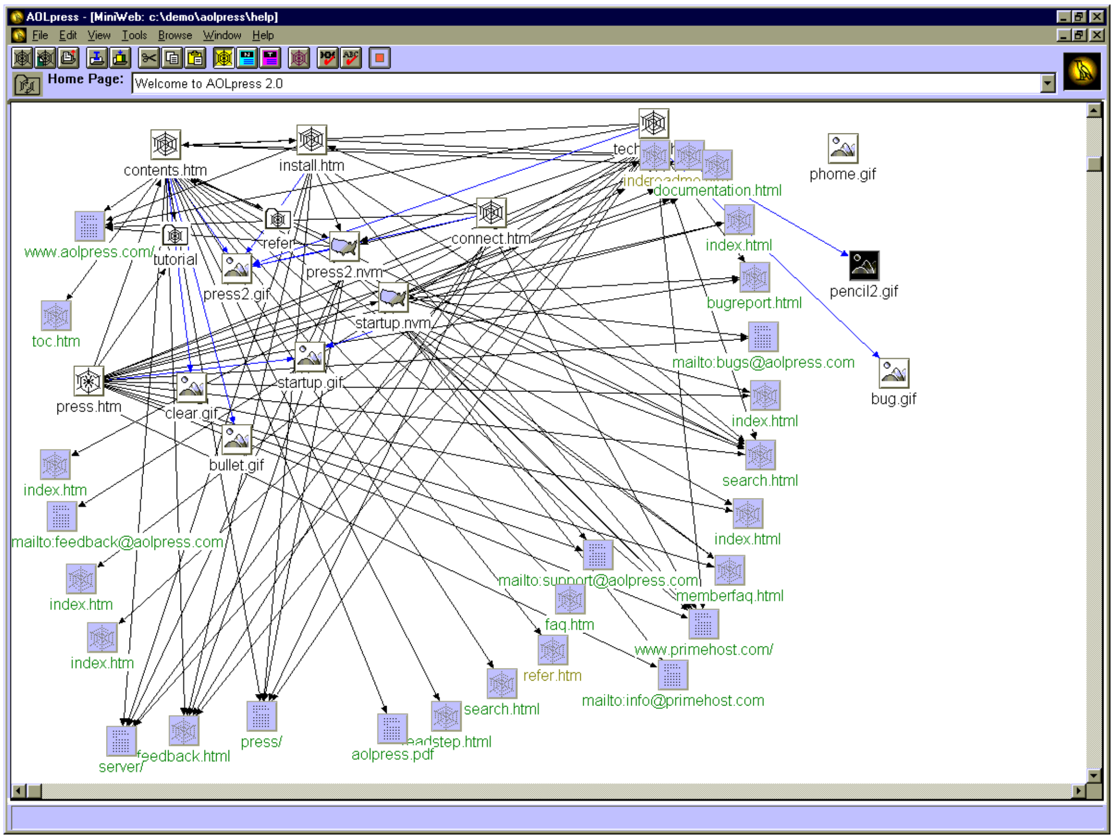


Fig. 2.17 : Représentation d'une scène complexe

II.5.2 Visualisation de réseaux de liens de grande taille

La visualisation de nombreux liens hypertextes ou hypermédia nécessite l'utilisation de méthodes de visualisation adaptées pour permettre une visualisation claire et éviter des vues trop complexes et incompréhensibles (voir un contre exemple sur la Fig. 2.18).

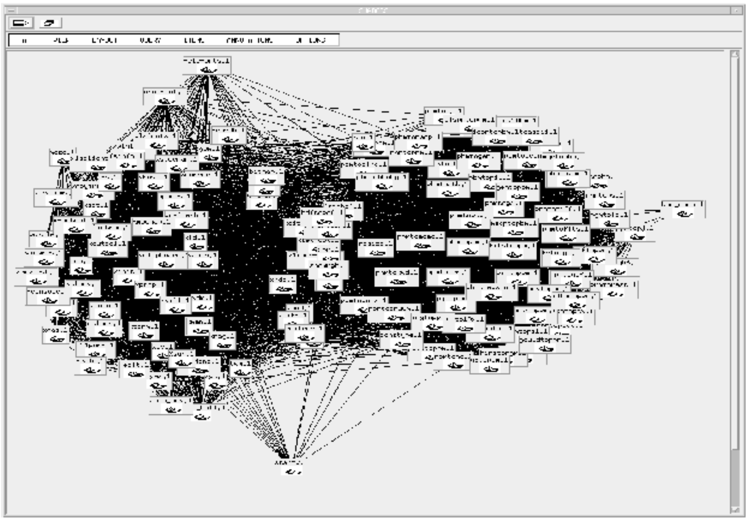


Fig. 2.18 : Nécessité d’avoir des méthodes de visualisation adaptées

La visualisation des liens entre les documents revient du fait de la grande taille du graphe, à visualiser une véritable toile d’araignée. Différentes techniques pour améliorer la visibilité de ces graphe ont été proposées. On peut par exemple, citer les méthodes de vue en œil de poisson (<Bibliographie>, <Bibliographie>).

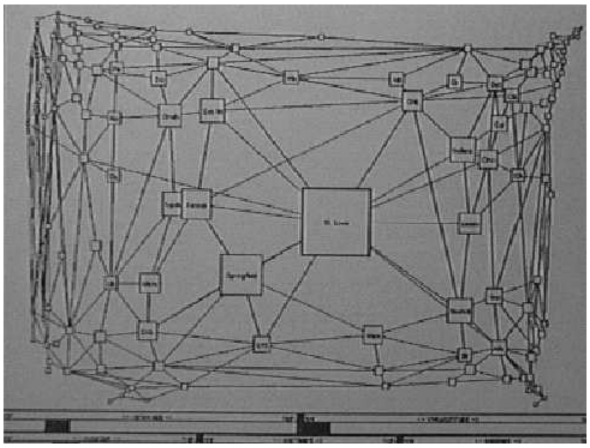


Fig. 2.19 : Vue en œil de poisson

La technique de la vue en œil de poisson (Fig. 2.19), a pour objectif d’améliorer la lisibilité du graphe en centralisant la vue sur un élément particulier.

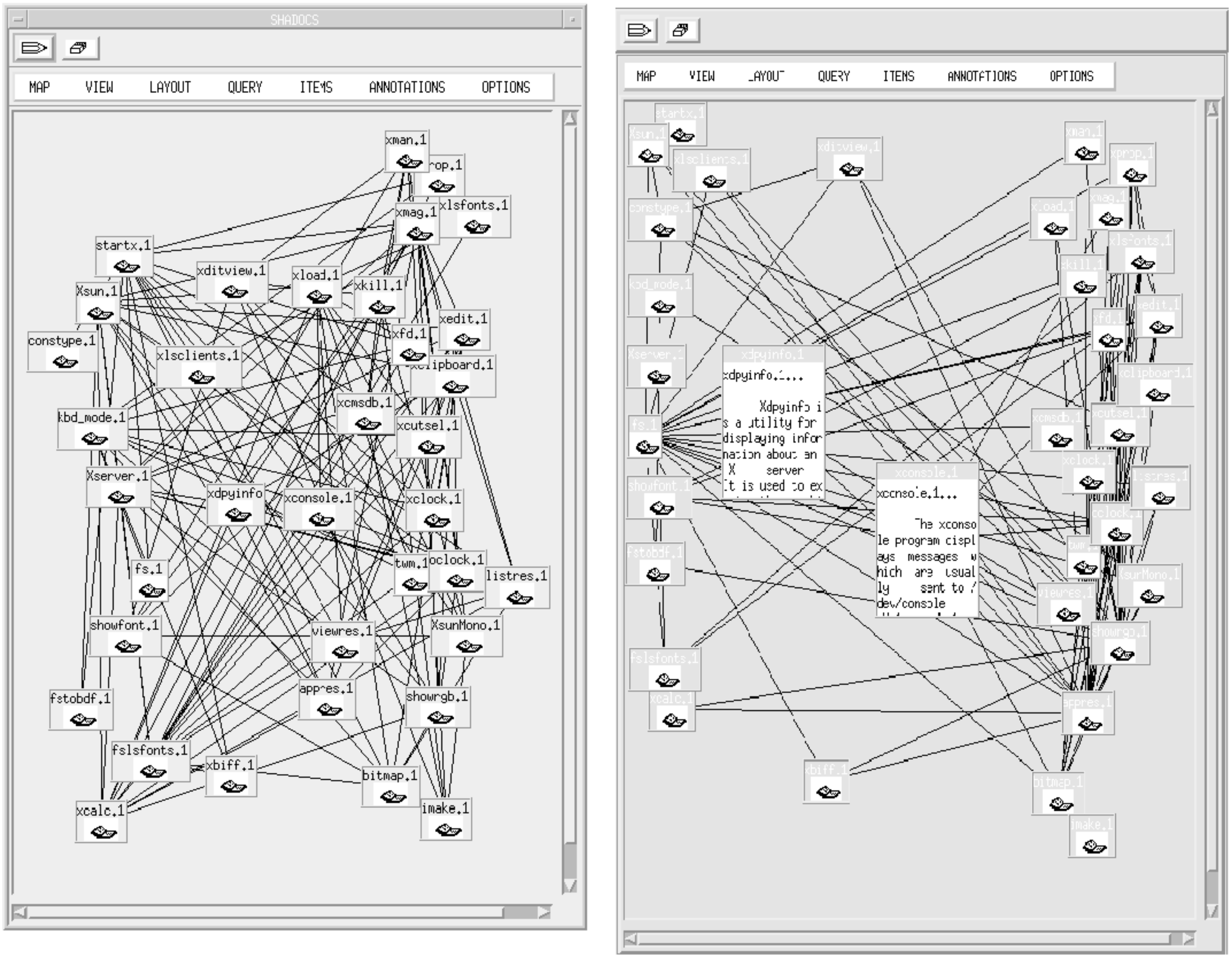


Fig. 2.20 : Réseau avec et sans vue en œil de poisson

Une généralisation de cette méthode a par exemple été faite dans le cadre de la visualisation des liens entre les fichiers résultants d'une requête de recherche sur le WEB <Bibliographie>. La méthode utilisée permet de visualiser les liens et les documents en trois dimensions (Fig. 2.21).

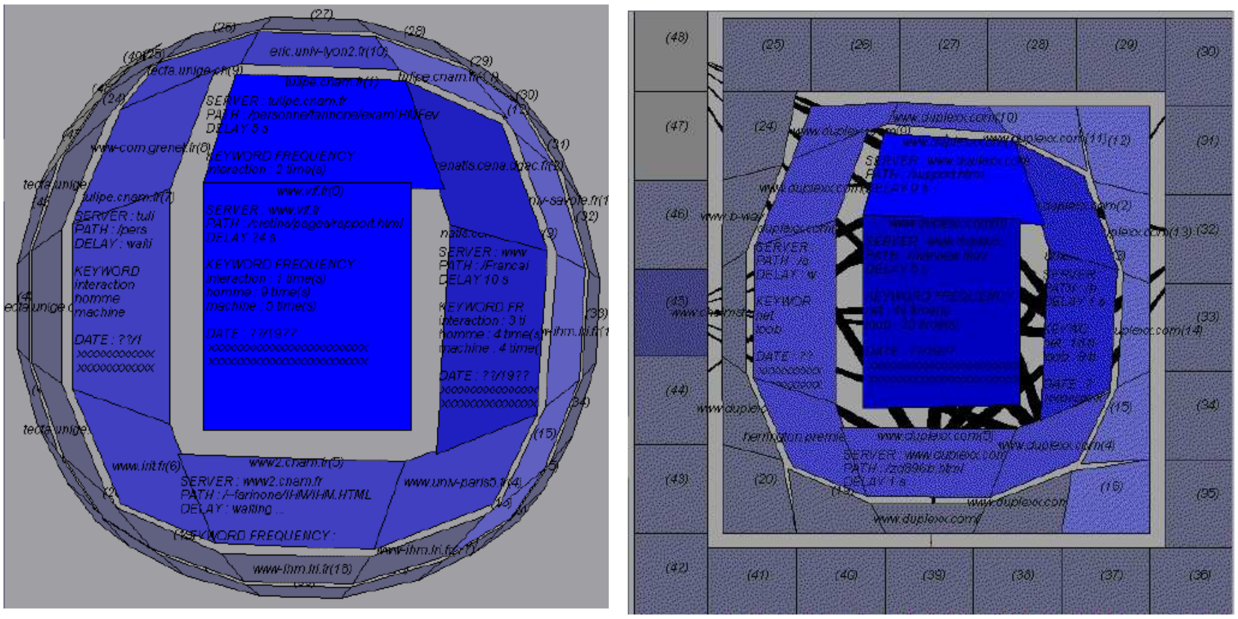


Fig. 2.21 : Technique appliquée au logiciel Vitesse <Bibliographie>

II.6 Visualisation de programmes

Un aspect du domaine de la visualisation d’information qui est aussi intéressant par rapport à notre problématique est la visualisation de programmes. En effet notre problématique de visualisation graphique du source textuel du scénario temporel s’inscrit bien dans cette thématique.

La visualisation de programmes a pour but d’aider le programmeur tout au long du cycle de vie du programme. Dans ce mémoire, je ne considère que l’aspect visualisation du code d’un programme et de ses données.

Nous pouvons distinguer quatre catégories de techniques de visualisation <Bibliographie> :

- Visualisation statique du code :

  ces techniques visent à améliorer la lisibilité des programmes par des subterfuges de présentation, offrir des outils graphiques pour documenter les programmes ou produire des représentations graphiques de ce que pourra être l’exécution du programme. Parmi les techniques utilisées, on peut citer l’utilisation d’organigrammes et de diagrammes de flots.

- Visualisation dynamique du code :

  ces techniques s’appuient sur une représentation graphique dynamique de ce qu’est l’exécution du programme. Parmi les techniques utilisées ont peut citer l'utilisation de la video inverse, qui permet dans le cas de sources sous forme textuelle de mettre en évidence les parties de code exécutées. Dans le cas de programmation à objets, l'effort de visualisation porte sur la communication entre les objets, généralement des méthodes graphiques sont utilisées.

- Visualisation statique des données :

  les structures de données y sont représentées de façon statique à l'aide d'une représentation graphique qui dépend du domaine d'application du programme (courbes, diagrammes).

- Visualisation dynamique des données :

  ces techniques permettent de tracer automatiquement l'évolution de données jugées digne d'intérêt par l'utilisateur. On peut par exemple citer l'utilisation de méthodes telles que les arbres coniques.

Il est intéressant de noter le parallèle avec notre problématique :

- La visualisation statique du code se traduit chez nous par la volonté de visualiser l'ensemble des solutions d'un système de contraintes.
- La visualisation dynamique du code et l'effort pour faire le lien entre l'exécution du programme et son fichier source, s'apparente dans notre contexte à faire le lien entre une exécution du scénario temporel et sa définition.
- La visualisation statique des données s'apparente à la visualisation des durées (minimales, maximales, optimales) des objets, que l'on assimile aux variables d'un programme (ici, le scénario temporel).
- La visualisation dynamique des données, revient à visualiser les durées effectives des objets.

Comme nous venons de le voir, le lien entre la visualisation de programme et notre problème de visualisation du scénario temporel d'un document multimédia peuvent être rapprocher. Néanmoins les techniques de visualisation statiques sont surtout développées pour des approches impératives et ne s'adapte pas à notre technique déclarative. Par contre, la visualisation dynamique nous semble intéressante à approfondir lors de l'intégration de la phase d'édition dans la vue graphique du scénario temporel.

II.7 Microsoft project

Nous allons maintenant présenter une application qui manipule à la fois des données hiérarchiques, temporelles et des contraintes: elle se nomme Microsoft Project <Bibliographie>.

Microsoft Project est un planificateur de projet industriel. Le but d’un planificateur est d’aider une personne à ordonner dans le temps des tâches qui sont liées temporellement (séquence, parallèle,...). Nous allons plus particulièrement présenter MSP, un planificateur de tâches à base de Time–Line et de contraintes sur des instants. MSP offre plusieurs vues à l’auteur. Les deux principales sont le time–line (Fig. 2.23) et le graphe de contraintes (Fig. 2.22).

L’auteur, pour construire le graphe de contraintes, crée des objets (tâches) et met des liens entre deux objets, ce lien est une contrainte qui oblige la seconde tâche à ne commencer qu’après la fin de la première.

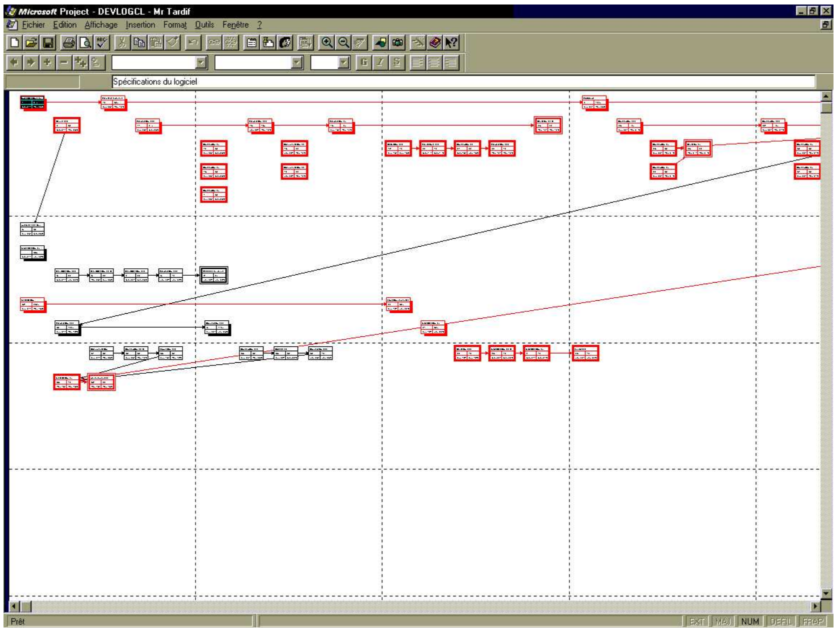


Fig. 2.22 : Graphe de contraintes

La sémantique de ces contraintes est visualisée sur une vue "Time–line" présentée dans la Fig. 2.23. Dans cette vue la relation de séquentialité est montrée par des flèches verticales. Des informations personnalisables peuvent être associées aux contraintes.

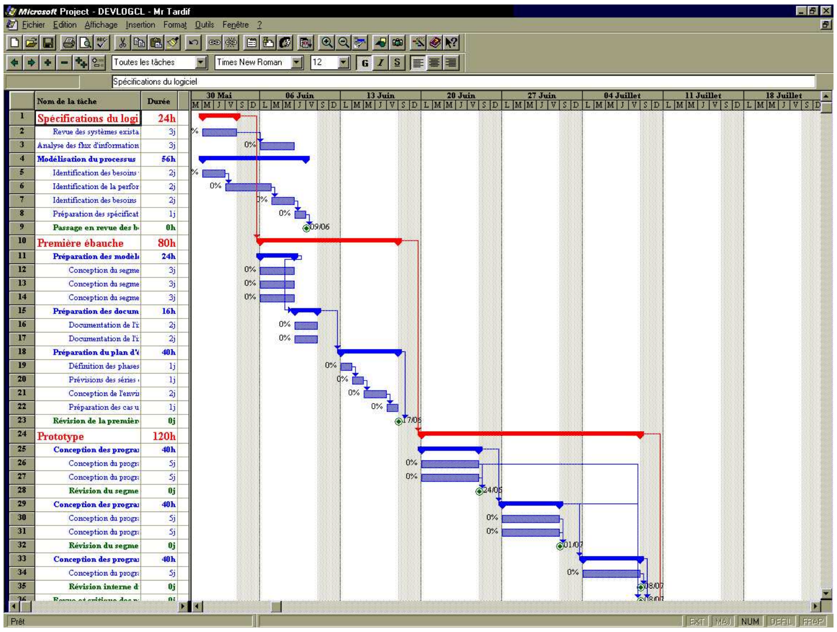


Fig. 2.23 : Time–line

Dans le "Time–Line" on peut remarquer la notion de hiérarchie, qui est représentée par des effets de style dans la partie gauche. Il est intéressant de noter que les tâches composées ainsi que leurs liens, sont aussi représentés sur le "Time–line". La possibilité est donnée à l'auteur de déplacer les objets (les tâches) en fonction des contraintes qui les régissent. Le rendu visuel est relativement pauvre, la vue n'est mise à jour que lorsque l'objet a été placé à la position désirée. De plus au moment du déplacement, l'auteur n'a aucune information sur l'intervalle dans lequel il peut se déplacer. Au moment où il fait un déplacement non–autorisé une fenêtre est ouverte lui indiquant la raison pour laquelle ce déplacement est interdit. Cette information, aussi intéressante soit–elle, n'est qu'un début dans l'information pour l'auteur sur les raisons de cette interdiction, de plus dans des planifications complexes elle est très insuffisante (Fig. 2.24 , Fig. 2.25). Lorsque l'on veut déplacer l'objet 1, le système nous signale seulement que notre tâche est liée à une autre qui ne peut être déplacée. En aucun cas le système nous signale laquelle, de plus le type de lien (la tâche Objet1 doit se terminer avant le début de la tâche Objet 2) mis sur les objets, permet un petit déplacement de l'objet en haut à gauche.

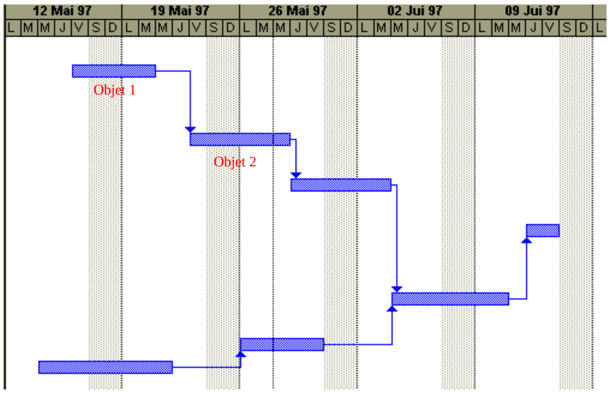


Fig. 2.24 : Interdiction de déplacement

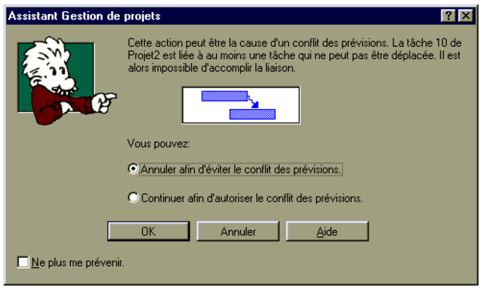


Fig. 2.25 : Explication de l'interdiction

On peut noter que le graphe de contraintes est relativement complexe à comprendre et demande un effort cognitif important de la part de l'auteur. De plus il ne permet pas d'anticiper les répercussions que peut avoir la modification de ce graphe sur la vue "time–line".

II.8 Conclusion

Dans ce chapitre nous avons abordé plusieurs aspects de la visualisation de données : l'aspect temporel, l'aspect hiérarchique, l'aspect visualisation de contraintes, l'aspect visualisation de structures de navigation et l'aspect visualisation de programmes. La synthèse que nous faisons sur ces différents travaux est la suivante :

- Les travaux sur la visualisation d'informations temporelles sont peu nombreux et ne proposent pas d'autres solutions que l'utilisation de "Time–line" améliorée par une vue en perspective.
- Les méthodes proposées pour la visualisation d'informations hiérarchiques s'avèrent difficilement utilisables dans notre contexte du fait de

l'interdépendance entre les informations temporelles et les informations hiérarchiques.

- Les travaux sur la visualisation de contraintes sont peu développés et sont basés sur la manipulation directe du résultat avec respect des contraintes.
- Les travaux sur la visualisation de structures de navigation ont mis en évidence la nécessité d'avoir un mode de représentation adapté au volume d'information manipulé. Ce qui peut nous être utile pour améliorer la lisibilité de la vue graphique du scénario temporel d'un document multimédia.
- Les travaux sur la visualisation de programmes portent essentiellement sur des approches impératives. Il nous faut surtout retenir les quatre types de besoins mis en évidence et l'application possible des techniques de visualisation dynamique lors de l'intégration de la phase d'édition dans la vue graphique du scénario temporel.

Nous avons aussi présenté une application qui intègre les trois premiers aspects et qui permet de manipuler directement les objets. Nous retiendrons de l'étude de ce système la nécessité et la difficulté d'expliquer les causes d'un blocage, ainsi que la visualisation explicite de la hiérarchie qui nous semble réutilisable dans notre contexte.

Nous allons voir à présent comment ces différentes techniques de visualisation peuvent être utilisées pour répondre au problème de visualisation posé dans Madeus.

Chapitre II

Etat de l'art sur la visualisation de données

Chapitre III

Visualisation du scénario temporel au sein de Madeus

III.1 Introduction

Le scénario temporel de Madeus est défini par un ensemble de contraintes entre les objets du document. Nous allons nous intéresser dans ce chapitre à l'étude d'une visualisation graphique de ces contraintes, visualisation qui n'est pas seulement un équivalent graphique du source textuel mais qui a pour but de permettre à l'auteur d'appréhender le comportement global de son scénario.

Dans une première partie (section III.2) nous présentons à la fois les besoins et les difficultés que soulève la réalisation de cette vue graphique.

Par la suite, nous présenterons une solution qui répond en partie à ces besoins. La restriction majeure que nous faisons au problème général est de considérer que les objets ont une durée unique. Par conséquent la distinction entre les objets contrôlables et les objets non–contrôlables (présentée dans la section I.6.3) n'est pas prise en compte.

Cette restriction s'explique par la difficulté de manipuler dynamiquement des objets de taille variable et un placement temporel variable. L'intégration de ces deux types de contraintes nécessite l'utilisation d'un résolveur incrémental de contraintes qui permet des manipulations directes de la part de l'auteur. En effet, l'introduction de contraintes sur la durée des objets ajoutée aux contraintes de placement peuvent être à l'origine de cycles. Par exemple, lorsque l'on définit une contrainte `A equal B` entre deux objets de taille flexible, plusieurs contraintes sont définis dans le résolveur :

- Début(A) = Début(B)
- Fin(A) = Fin(B)
- Durée(A) = Fin(A) – Début(A)
- Durée(b) = Fin(B) – Debut(B)

Si on trace le graphe de dépendance des variables on s'aperçoit de l'existence d'un cycle. De plus ce ce résolveur doit pouvoir gérer des contraintes d'inégalité qui traduisent des placements temporels du type `A Before B`.

Les travaux de L. Carcone [8] effectués au sein du projet Opéra en parallèle à mon stage, ont permis de mettre en évidence qu'actuellement aucun résolveur ne le permettait de manière satisfaisante. Malgré cette restriction le besoin en visualisation graphique du scénario subsiste.

Nous avons porter nos efforts sur les autres aspects du problème afin de dégager et de valider des principes de base pour répondre à notre problématique initiale. Afin de diviser les problèmes nous avons choisi de traiter d'abord le problème sans structuration hiérarchique (section III.3) puis d'intégrer celle–ci à la solution de base (section III.4). Notre proposition fait l'objet de la réalisation d'un prototype, externe au système Madeus, qui permet à partir d'un scénario donné de visualiser l'ensemble des placements temporels possible des objets. Cette implantation a été réalisée en Tcl/Tk [44], un langage interprété qui nous a permis d'élaborer de manière relativement simple et rapide des prototypes d'interface.

Bien que nous n'ayons pas pu, à l'occasion de ce stage répondre à tous les besoins identifiés dans la section III.2, il nous a semblé pertinent de donner dans la section III.5 une première ébauche les idées que nous avons pour en traiter certains.

La section III.6 est dédiée à la présentation de certains algorithmes clés utilisés pour mettre en œuvre la solution.

La dernière partie de cette section est dédiée à la présentation des premiers tests réalisés avec des utilisateurs potentiels.

Avant de présenter les besoins et difficultés soulevés par notre la visualisation de scénario temporel, nous présentons un exemple qui servira de fil conducteur tout au long de ce chapitre. La Fig. 3.1 donne les contraintes (la sémantique intuitive des contraintes a été présentée dans la section I.6) qui relient les objets du document ainsi que la durée des objets. Tous les objets considérés ici sont des objets de base (pas de structure hiérarchique).

Video 1 *Start* Texte 1

Video 1 *Equal* Son 1

Video 1 *Before* Image 1

Image 1 *Start* Texte 2

Image 1 *Before(d3)* Image 2

Texte 2 *During* Texte 1

Texte 2 *Finish* Image 2

OBJETS	DUREES
Texte 1	25
Video 1	5
Son 1	5
Image 1	5
Image 2	5
Texte 2	13
d3	3

Fig. 3.1 : Le scénario de référence

Dans la Fig. 3.2 nous montrons un placement temporel possible des objets.

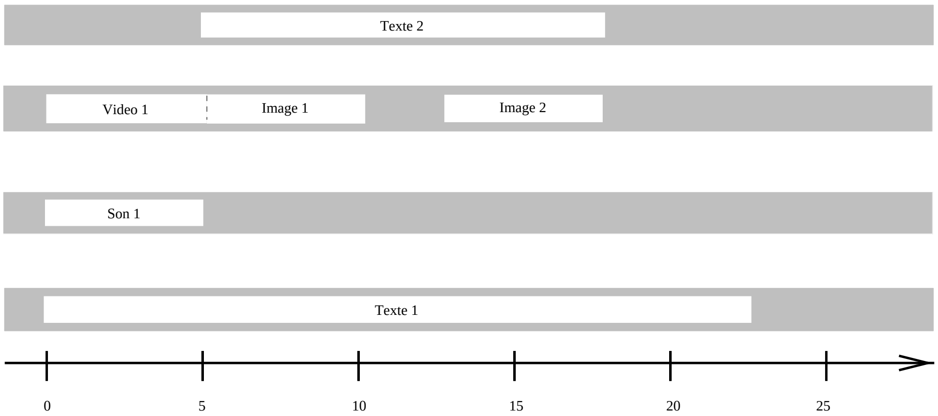


Fig. 3.2 : Un exemple de placement temporel

III.2 Besoins et difficultés

Nous allons maintenant présenter les différents besoins liés à la visualisation graphique du scénario temporel (le problème est traité ici sans restriction).

III.2.1 Visualisation d'un ensemble de solutions

- Nature des objets:

  Une des idées fortes de l'approche adoptée dans Madeus est que l'auteur n'a pas à fixer la durée précise de chaque objet. C'est au système de calculer l'ensemble des durées de manière à satisfaire la spécification (rôle du planificateur ou formateur temporel). Pour cela, deux types d'objets sont différenciés dans Madeus : les objets contrôlables et les objets non–contrôlables (qui ont été introduits dans la section I.6).

Par conséquent, non seulement la durée des objets que l'on manipule n'est pas unique, elle varie dans un intervalle délimité par des bornes, mais en plus ces intervalles de durées possibles ont deux sémantiques différentes pour l'auteur. Dans le cas des objets contrôlables, c'est un intervalle de choix et l'auteur peut, s'il le souhaite, changer sa valeur en modifiant sa spécification. Dans le cas des objets incontrôlables, c'est un intervalle de valeurs possibles à l'exécution sur lequel l'auteur ne peut rien faire. Il est donc nécessaire de faire apparaître cette différence lors de la visualisation.

- Nature des relations:

  Certaines relations de placement relatif introduisent la présence de délai "flexible". C'est le cas par exemple des relations `Before` ou `During`. Le délai entre les deux objets n'est pas fixé, c'est un délai contrôlable au sens défini ci–dessus.

La nature des objets ainsi que celle des relations conduisent à la visualisation d'un ensemble de solutions et non pas d'une solution unique, ce qui rend plus complexe le problème auquel nous tentons d'apporter une solution.

III.2.2 Visualisation des causes d'une configuration

Etant donnée une spécification, certaines configurations de placement des objets dans le temps sont possibles alors que d'autres ne le sont pas. Comprendre les raisons de ce partage est une tâche complexe, car la cause peut provenir d'une contrainte induite par transitivité à partir des contraintes placées par l'auteur. Nous donnons ici un exemple de contrainte induite : si l'auteur spécifie "A Before B' et "B Before C", on obtient une contrainte induite qui est "A Before C".

On rejoint avec ce souci de visualiser les causes d'une configuration la volonté de satisfaire le critère d'observabilité tel qu'il est défini dans la section I.2. Nous avons vu dans la section II.7 avec Microsoft Project que ce besoin est ressenti dès que l'on utilise des contraintes.

Si l'on considère l'exemple donné en introduction, il n'est pas possible de rapprocher `Image 2` de `Video 1` en deçà d'une certaine limite à cause des contraintes placées entre `Image 2` et `Image 1` `(Image 1 before(d3)`

`Image 2)` d'une part, et entre `Image 1` et `Video 1 (Video 1 before Image 1)` d'autre part.

Il faut donc trouver un moyen efficace pour aider l'auteur dans cette tâche de compréhension. Cela pose le problème de visualisation des contraintes. Doit–on les représenter explicitement au risque de surcharger l'affichage, comme dans le système FireFly présenté dans la section I.6.1, et dans ce cas comment traitons–nous les contraintes induites par transitivité ?

Cette aide est d'autant plus difficile à mettre en place qu'une même configuration d'objets peut être obtenue par des spécifications différentes et qu'il est important pour l'auteur de visualiser ces différences. En effet, deux placements identiques mais s'interprétant de manière différente, peuvent mener à des ensembles de configurations différentes. Par exemple, la configuration présentée dans la Fig. 3.2 peut aussi s'obtenir en remplaçant la contrainte `Vidéo 1 equal Son 1` par `Son 1 Before Image 1.` Dans les deux cas, nous obtenons un ensemble de solutions différent, même s'ils ont des solutions communes.

III.2.3 Visualisation d'une hiérarchie d'informations

Pour faciliter la compréhension de cette section, nous allons étendre l'exemple introductif (section III.1) pour intégrer la notion de hiérarchie. Nous utilisons une hiérarchie simple définie avec trois objets composites :

- Document
- Scène 1
- Scène 2

Les éléments des scènes 1 et 2 sont représentés dans la Fig. 3.3. Les contraintes définies entre les objets de ces deux scènes sont celles introduites dans l'exemple introductif (Fig. 3.1).

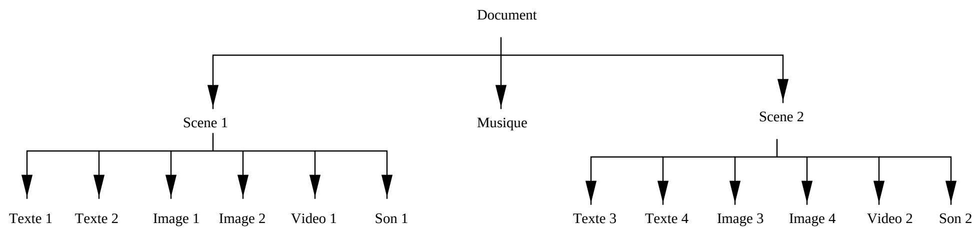


Fig. 3.3 : Un exemple de hiérarchie

Les relations entre les objets Scène 1, Scène 2 et Musique sont définies dans la Fig. 3.4.

Scène 1 *Equal* Scène 2
Scène 1 *Equal* Musique

Fig. 3.4 : Extension de l'exemple de la Fig. 3.1

Dans Madeus, il est possible de décomposer hiérarchiquement son document afin de maîtriser sa complexité. Cette hiérarchie se retrouve au niveau de la spécification temporelle du document. Plus précisément, à chaque objet composite de la hiérarchie est associé un ensemble de contraintes temporelles reliant potentiellement tous les fils de cet objet. Autrement dit, tous les fils d'un même père forment un groupe et sont reliés par des contraintes temporelles. Si l'on considère l'exemple de la Fig. 3.3, des contraintes peuvent relier les scènes 1, 2 et l'objet Musique. Un autre ensemble de contraintes relie les objets composant la `Scène 1` et ainsi de suite. Par conséquent, la structure hiérarchique n'est pas complètement déconnectée de la visualisation des placements temporels et il paraît important de pouvoir visualiser ces deux informations au sein d'un même affichage. Dans l'exemple de la Fig. 3.3, il est nécessaire que l'auteur puisse visualiser dans une même vue les composants de la `Scène 2` et ceux de la `Scène 1`, même s'ils n'ont pas de liens directs entre eux, car la contrainte mise sur les objets `Scène 1` et `Scène 2` impose un recouvrement temporel entre leurs composants. L'auteur doit aussi pouvoir visualiser s'il le souhaite, le contenu de deux scènes mises en séquence pour voir comment s'effectue leur transition.

L'auteur a donc besoin de :

- visualiser la hiérarchie pour savoir entre quels objets il peut définir une contrainte ;
- visualiser les placements temporels possibles entre les objets d'un même groupe ;
- visualiser les placements temporels possibles entre des objets appartenant à des groupes de niveaux quelconques.

III.2.4 Visualisation de données de grandeur hétérogène

Nous voulons mettre ici en évidence le problème de la représentation à l'échelle de la durée des objets et de la visualisation du document dans sa globalité. Les difficultés qui se posent ici sont classiques et sont liées aux limitations physiques de la taille d'un écran et des capacités de perception humaines.

Imaginons par exemple, que l'objet `Texte 1` (qui par exemple peut être le titre de la présentation) soit présent tout au long du document et dure 25 minutes, et qu'à un moment de sa présentation nous affichions une image qui reste 5 secondes à l'écran. Si nous affichons dans la vue graphique du scénario des objets dont la longueur est proportionnelle à leur durée, il faut que la taille de `Texte 1` soit environ deux cent fois plus grande que la taille de l'image. Il n'est donc pas possible de visualiser complètement les deux objets à l'échelle tout en voulant que l'objet `Image 1` reste preceptible.

III.2.5 Non visualisation de fausses informations

Parmi l'ensemble des configurations possibles que peuvent avoir deux objets, certaines sont plus "chargées d'informations" que d'autres. Par exemple, placer deux objets A et B de manière à ce qu'ils démarrent en même temps alors que cette configuration particulière n'est pas requise par les contraintes est un choix malencontreux qu'il faut éviter. On rejoint ici la volonté de satisfaire le critère d'honnêteté tel que défini dans [10].

Dans l'exemple introductif (section III.1), le placement temporel que nous avons choisi de montrer pour illustrer l'exemple est tel que les objets Video 1 et Image 1 se touchent, pourtant aucune contrainte ne force ce placement, mais aucune ne l'interdit. Nous souhaitons proposer une visualisation qui évite autant que possible ce genre de configuration et donc l'interprétation erronée du scénario temporel.

Remarquons ici, comme nous l'avions observé dans la section I.6.2 que la solution de visualisation proposée dans l'environnement ISIS ne satisfait pas ce besoin.

III.2.6 Liens entre les fenêtres

L'introduction d'une vue graphique du scénario introduit une troisième vue au sein de Madeus (vue d'édition textuelle, vue graphique du scénario, vue d'exécution du document (<Introduction>)).

Il est donc nécessaire de permettre à l'auteur de faire aisément le lien entre ces trois vues.

Dans l'exemple introductif, nous voyons que la donnée d'un exemple de placement (Fig. 3.2) temporel ne permet pas à l'auteur de visualiser les contraintes spécifiées. On retrouve ici le besoin de visualiser les contraintes exprimées par l'auteur déjà évoqué dans la section III.2.2.

Le premier besoin ressenti concerne la liaison entre la vue graphique du scénario temporel et la fenêtre comportant le source textuel. Ce lien doit d'être à double sens. En effet, quand l'auteur sélectionne une contrainte dans le source textuel, il doit pouvoir visualiser l'impact qu'elle a dans la vue graphique du scénario. De plus, quand il sélectionne un objet il doit pouvoir visualiser l'ensemble des contraintes directes qui sont sur cet objet.

De manière symétrique il est nécessaire de faire le lien entre la vue graphique du scénario temporel et la fenêtre d'exécution. Comme dans le cas précédent ce lien doit être à double sens. En effet, lorsque l'auteur joue le scénario il doit pouvoir visualiser à quel endroit il se trouve dans la vue graphique du scénario temporel. De même, quand il manipule la vue graphique et qu'il se place à un instant précis à l'intérieur, il doit pouvoir visualiser "un instantané" de la fenêtre d'exécution, par exemple pour avoir des informations sur le placement spatial des objets.

III.2.7 Visualisation de la structure de navigation

Un autre besoin que nous avons identifié est la visualisation de la structure de navigation. Les objets de Madeus peuvent être associés à des liens potentiellement activables (les différents types de liens ont été présentés dans la section I.6). Le problème consiste donc à visualiser statiquement ces possibilités offertes au lecteur.

Dans l'exemple introductif, si nous ajoutions un lien de type inclusion à l'objet Texte 1, de telle façon que lorsqu'on clique dessus un objet composite soit joué en même temps, il faut permettre à l'auteur de le visualiser dans la vue graphique du scénario temporel.

III.2.8 Visualisation de gros volumes d'informations

Une autre difficulté est le problème du passage à l'échelle. En effet, une méthode de visualisation qui s'avère bonne pour de petits documents peut s'avérer difficilement compréhensible pour de gros documents (qui peuvent comporter des centaines d'objets). Le problème sera donc de trouver une méthode de visualisation qui soit adaptée pour tous les documents. On a vu que les systèmes actuels répondaient mal à ce critère.

III.2.9 Visualisation des opérateurs d'interruption

Il est nécessaire de bien distinguer l'opérateur `parmin` de l'opérateur `equal`. En effet, les deux opérateurs se traduisent par un placement temporel identique : les opérandes démarrent en même temps et se terminent en même temps mais leur sémantique est différente: dans le cas du `parmin`, il se peut que l'un des deux opérandes ait été interrompu. Cette différence doit apparaître dans la vue graphique du scénario. De plus il est important pour l'auteur de savoir précisément quels sont les objets qui risquent d'être interrompus, car ceux–ci peuvent potentiellement ne pas délivrer la totalité de leur contenu.

Cette information n'est pas simple à connaître dès que les objets manipulés peuvent avoir différentes durées et que de plus il existe des objets incontrôlables. Considérons par exemple une relation `A parmin B` où A est un objet contrôlable dans l'intervalle [10, 15] et B est un objet incontrôlable dans l'intervalle [8, 12]. Parmi l'ensemble des exécutions possibles, certaines interrompent A (par exemple si A=10 et B=8), d'autres interrompent B (par exemple si A=10 et B=12). Cependant, l'auteur, qui peut agir sur la durée de A, peut modifier son intervalle de valeurs possibles pour qu'il ne soit jamais interrompu (par exemple en le fixant à [12, 15]). Il ne peut pas faire de même avec B car celui ci est incontrôlable. Ce sont toutes ces informations qu'il faut pouvoir fournir à l'auteur. Le problème est tout aussi difficile avec l'autre opérateur d'interruption (le `parmaster)` qui se traduit selon les cas soit par une interruption soit par le déroulement complet des deux objets.

Nous concluons cette partie analyse des besoins en précisant les points que nous avons abordés au cours du stage. Les autres sont évoqués dans la partie perspective qui termine ce chapitre. Sous l'hypothèse de durée unique des objets, nous avons apporté une réponse aux besoins suivants :

- visualisation d'un ensemble de solutions, cet ensemble est du à la nature des relations ;
- liaisons entre la vue graphique et la vue textuelle du scénario ;
- visualisation des causes ;
- visualisation de la hiérarchie.

Les trois premiers points sont présentés dans la section III.3, et la visualisation de la hiérarchie sera traitée dans la section III.4.

III.3 Une vue graphique sans hiérarchie

L'idée générale sur laquelle repose notre proposition est la suivante : nous proposons à l'auteur un placement possible des objets sur un ensemble d'axes temporels et nous matérialisons sur cette vue toutes les contraintes présentes dans le scénario. L'auteur peut manipuler (en déplaçant les objets sur leur axe temporel) cette vue afin de parcourir l'ensemble des placements temporels possibles à partir du scénario.

Nous commençons par présenter comment nous visualisons les objets, puis comment sont visualisées les contraintes exprimées dans le scénario (section III.3.1). Dans une deuxième partie (section III.3.2) nous présentons comment l'auteur visualise l'ensemble des solutions possibles à partir du scénario. Dans une troisième partie nous expliquons le lien entre les différentes fenêtres.

III.3.1 La vue graphique

Les objets de base sont représentés par des rectangles dont la longueur est proportionnelle à leur durée effective. Chaque type d'objet (vidéo, son, texte, image) est représenté par un motif discriminant (voir Fig. 3.5). Les objets sont placés sur des lignes horizontales qui représentent des axes temporels, le temps s'écoulant de gauche à droite. Les lignes donnent aussi l'intuition du déplacement temporel par un déplacement graphique horizontal des objets.

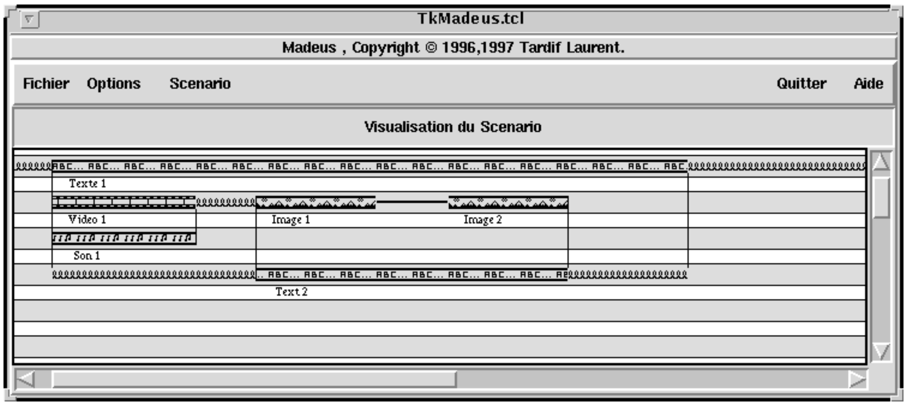


Fig. 3.5 : Vue graphique du scénario donné en exemple

Hormis les objets, les contraintes introduisent deux types d'éléments dans la vue graphique :

- les délais, qui sont de deux natures :
  - fixes, c'est par exemple le cas lorsque l'auteur spécifie `Image 1 before (15) Image 2.` Dans ce cas le délai est modélisé par un trait rigide.
  - flexibles, c'est par exemple le cas lorsque l'auteur spécifie `Video 1 Before Image 1`. Dans ce cas le délai est modélisé par un ressort.
- les traits verticaux, qui sont utilisés pour représenter certaines contraintes. Ils indiquent que deux instants sont simultanés. C'est le cas pour les opérateurs `start, equal, end,`... Par exemple lorsque l'auteur spécifie `Image 1 Start Texte 2`, un trait vertical relie les instants de début de ces deux objets.

La représentation explicite des objets, des contraintes et des délais introduits par ces contraintes permet une interprétation directe du scénario sans nécessairement avoir recours à la vue textuelle. Cela permet d'envisager la disparition de la vue textuelle telle qu'elle est actuellement dans Madeus, le jour où l'édition du scénario temporel se fera dans la vue graphique du scénario.

III.3.2 Visualisation de l'ensemble des solutions

Lorsqu'on sélectionne et déplace un objet, tous les objets dont le placement temporel est solidaire à cet objet (ils lui sont reliés par des contraintes qui n'introduisent pas de délais flexibles) suivent les mouvements imposés par l'objet sélectionné. Lors de cette sélection, l'objet sélectionné apparaît en rouge et les objets qui lui sont solidaires apparaissent en bleu. En vert apparaissent tous les délais flexibles dont la valeur est susceptible d'être modifiée par des déplacements de l'objet sélectionné.

Afin d'aider l'auteur dans le parcours de l'ensemble des solutions nous avons jugé opportun de lui fournir la visualisation de l'intervalle de déplacement de manière à ce qu'il puisse **anticiper le comportement** de l'objet qu'il manipule (critère de prévisibilité). On retrouve ici l'idée des axes de déplacement possibles visualisés dans Drawing with Constraint (section II.4.2).

Dans l'exemple de la Fig. 3.6, l'auteur a sélectionné l'objet `Texte 2.` Un trait rouge apparaît sous la ligne où est placé l'objet sélectionné, indiquant ainsi la latitude de déplacement de l'objet dans les deux sens. Sur notre exemple, Texte 2 est limité à sa gauche par la fin de l'objet Vidéo 1 et est limité à sa droite par la fin de l'objet Texte 1.

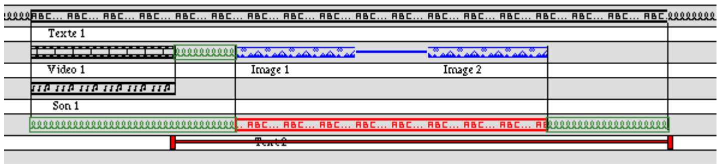


Fig. 3.6 : Visualisation d'informations

Ce choix de ne visualiser des informations pertinentes que lorsque l'auteur sélectionne un objet nous semble primordial pour éviter de surcharger la fenêtre et pour limiter l'effort cognitif nécessaire à la compréhension de cette dernière.

Dans cette solution, nous avons fait le choix de ne pas bouger les objets qui ne sont pas liés par des contraintes rigides (qui n'introduisent pas de délais flexibles) avec l'objet sélectionné. Un autre choix aurait été de bouger tous les objets liés avec l'objet sélectionné. L'idée de cette autre solution est la suivante : l'objet sélectionné après avoir absorbé un délai flexible ne se bloque plus sur l'objet atteint mais l'entraîne avec lui dans son déplacement.

Pratiquement cela revenait à tout bouger. Nous avons pensé que cela demanderait un effort cognitif trop important à l'auteur pour appréhender le comportement global d'un déplacement. Néanmoins tous les placements possibles restent accessibles à l'auteur. Pour cela, il lui suffit simplement de sélectionner un autre objet.

Sur l'exemple de la Fig. 3.6, si nous voulons déplacer Image 1 jusqu'à la fin de Video 1, il aurait fallu déplacer Video 1, mais aussi Son 1 et Texte 1, et donc tous les objets de la scène. Cela aurait été gênant pour l'auteur, car trop complexe. Si l'auteur veut atteindre une telle position il lui faut sélectionner Image 1 et le déplacer jusqu'à la position voulue, puis redéplacer Video 1.

Bien évidemment, lorsque l'auteur déplace un objet sur le time–line, la solution courante du scénario ainsi que la représentation graphique de cette dernière évolue immédiatement de manière à s'adapter aux modifications que l'on a faites (ie: la position des objets et des intervalles s'adapte et est visualisée mmédiatement à l'écran). Par exemple, la Fig. 3.7 montre la vue obtenue lorsqu'on déplace l'objet Texte 2 le plus à droite possible.

Fig. 3.7 : Texte 2 le plus à droite possible

III.3.3 Liaisons entre la vue graphique et la vue textuelle du scénario

Comme nous l'avons dit dans la section III.2.6 il est nécessaire de faire explicitement le lien entre les différentes fenêtres. Dans cette solutio, nous nous sommes intéressés à deux types de lien:

- le lien entre la spécification textuelle des contraintes et la vue graphique du scénario temporel ;
- le lien entre la vue graphique du scénario temporel et la spécification textuelle du scénario.

Dans le premier cas (Fig. 3.8), lorsque l'auteur sélectionne une contrainte dans la vue textuelle du scénario temporel, on visualise dans la vue graphique les objets et les délais introduits par cette contrainte. Dans l'exemple de la Fig. 3.8, l'auteur sélectionne une contrainte, tous les objets avec lesquels elle est directement liée sont mis en évidence.

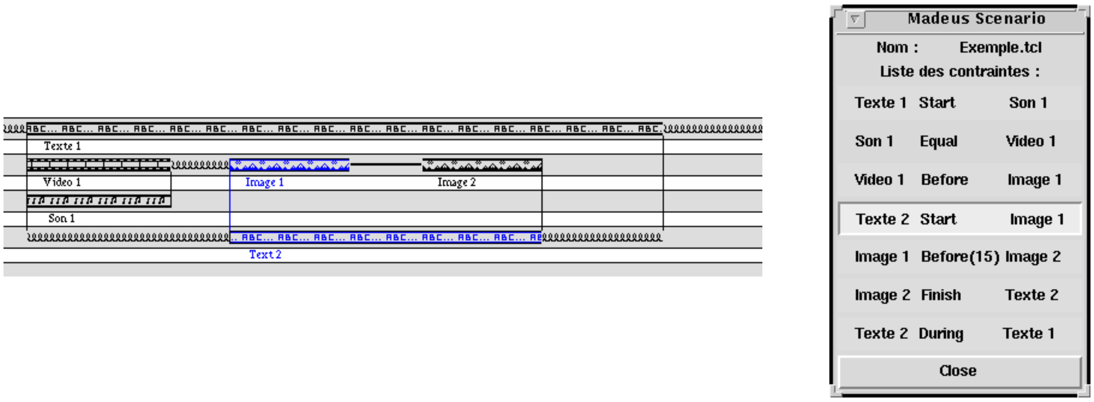


Fig. 3.8 : Liens entre les vues textuelle et graphique du scénario

Dans le second cas, lorsque l'auteur sélectionne un délai (fixe ou flexible) dans la vue graphique (Fig. 3.10), il visualise dans la vue textuelle la contrainte à l'origine de cet intervalle. Lorsqu'il sélectionne un objet dans la vue graphique (Fig. 3.9), il peut visualiser l'ensemble des contraintes liées à cet objet.

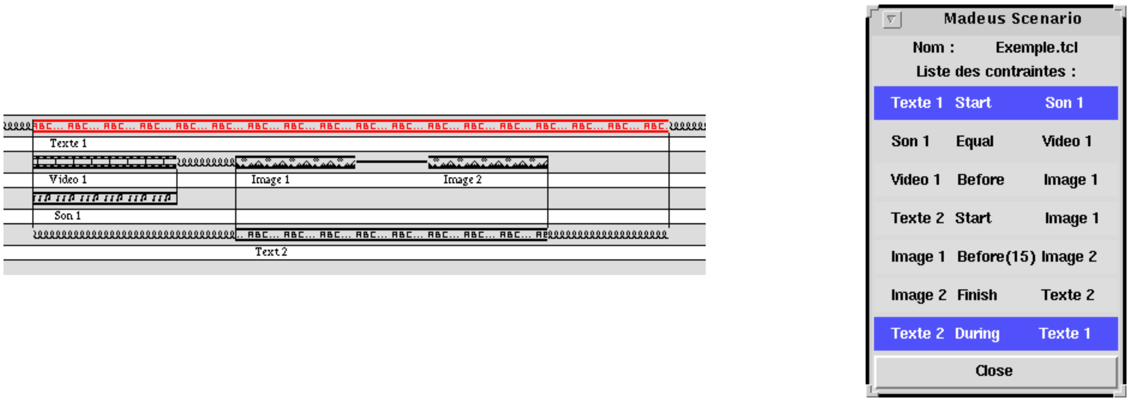


Fig. 3.9 : Liens entre les vues graphique et textuelle du scénario (a)

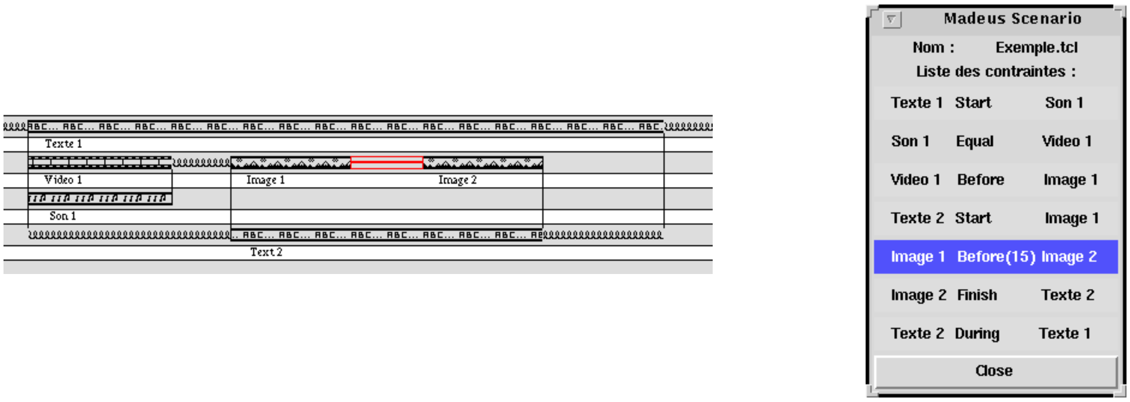


Fig. 3.10 : Liens entre les vues graphique et textuelle du scénario (b)

III.3.4 Visualisation des causes

La visualisation des causes d'une configuration se ramène dans notre cas à visualiser les causes qui limitent le déplacement d'un objet. En effet, la visualisation du "qui" bloque le déplacement de l'objet et du "où" il se bloque ne suffise pas, il faut aussi le "pourquoi" il se bloque. Un objet se bloque que si un des délais flexibles liés à son déplacement (ressort vert) devient nul. Cette information est donnée dans notre vue et de plus, grâce au mécanisme de liaison entre la fenêtre source et la vue textuelle, il est facile pour l'auteur de comprendre d'où vient ce délai flexible.

Sur l'exemple de la figure Fig. 3.11 il faut visualiser le fait que l'objet Image 1 est bloqué dans son déplacement à droite par l'objet Texte 1 à cause des relations `Image 1 Start Texte 2` et `Texte 2 During Texte 1`.

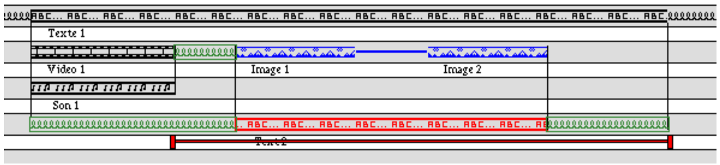


Fig. 3.11 : Visualisation d'informations

III.4 Une vue graphique avec hiérarchie

La solution que nous venons de présenter visualise un niveau de contraintes entre les objets. Nous présentons à présent la manière d'intégrer différents niveaux de hiérarchie à cette vue. Nous rappelons que des contraintes peuvent être placées entre deux objets si et seulement si ils partagent le même père.

Rappelons aussi que l'auteur a besoin non seulement de visualiser l'ensemble des placements possibles des objets reliés par le même père (ils font partie d'un même groupe), mais aussi de voir simultanément le placement d'objets qui ne font pas partie du même groupe pour, par exemple, affiner la transition entre deux scènes en séquence. C'est ce dernier cas qui rend complexe le problème d'intégration de la hiérarchie, puisque dans le premier cas toute méthode "classique" (arbre, structure indentée) ou plus novatrice (arbre conique) de représentation de la hiérarchie est satisfaisante. Il aurait en effet suffit d'associer à chacun des pères la vue graphique du scénario définie précédemment et de donner la possibilité à l'auteur d'ouvrir et fermer ces vues en cliquant sur les nœuds pères.

Dans l'article [32] une première solution de visualisation pour représenter dans une même vue la structure hiérarchique et les informations temporelles au sein de Madeus a été présentée (Fig. 3.12). Cependant, les informations temporelles sont uniquement constituées par le nom des contraintes qui relient les objets fils d'un même père. La sémantique de ces contraintes n'est donc pas visualisée.

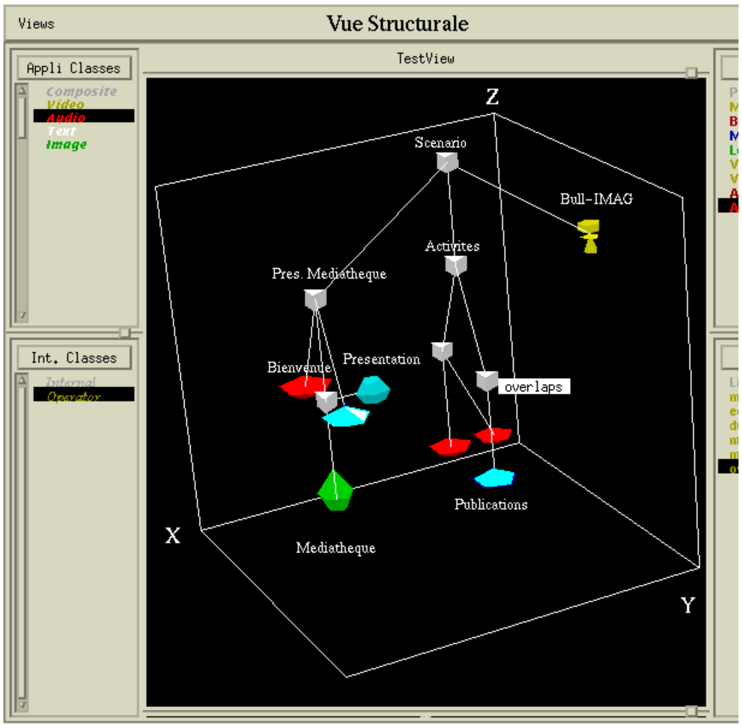
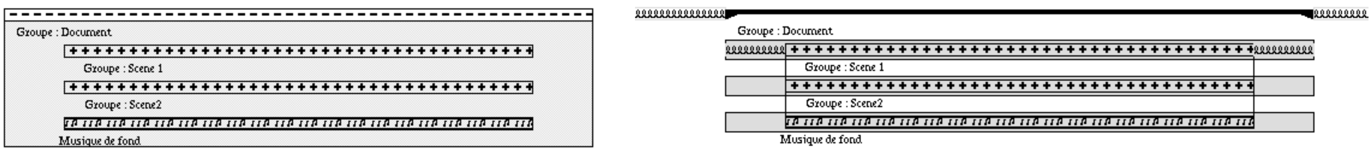


Fig. 3.12 : Visualisation de la hiérarhie

Afin de satisfaire les trois types de besoins énoncés dans la section III.2.3, nous avons opté pour un affichage de la hiérarchie à l'intérieur du scénario temporel (solution inspirée de Microsoft Project (section II.7)) associé à une vue de la hiérarchie sans informations temporelles.

Sur la Fig. 3.13 nous voyons la fenêtre de visualisation associée à notre exemple avec à gauche le premier niveau de hiérarchie et à droite la vue graphique associée à l'ensemble des contraintes qui relient ces trois objets. L'auteur peut ouvrir ou fermer un niveau de hiérarchie en cliquant dessus (Fig. 3.14).



VUE HIERARCHIQUE

VUE GRAPHIQUE DU SCENARIO

Fig. 3.13 : Hiérarchie fermée

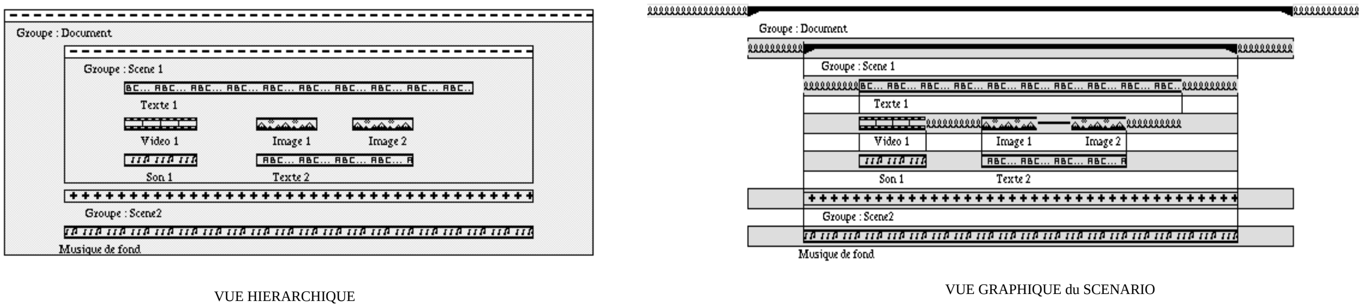


Fig. 3.14 : Hiérarchie ouverte

Une fois ouvert, l'objet composite reste visible dans la vue graphique du scénario et peut être soumis à des déplacements. Dans ce cas, nous appliquons le principe suivant : tous les objets du même niveau qui ont une relation forte avec cet objet sont mis en bleu, ainsi que tous leurs descendants. Tous les objets suivent les déplacements de l'objet sélectionné. Les rails de déplacement que nous visualisons à chaque niveau limitent les déplacements de l'objet à son niveau de hiérarchie. Par exemple lorsque l'élément Musique de fond est sélectionné il ne peut se déplacer qu'entre le début et la fin de son père (le groupe Document (Fig. 3.15)).

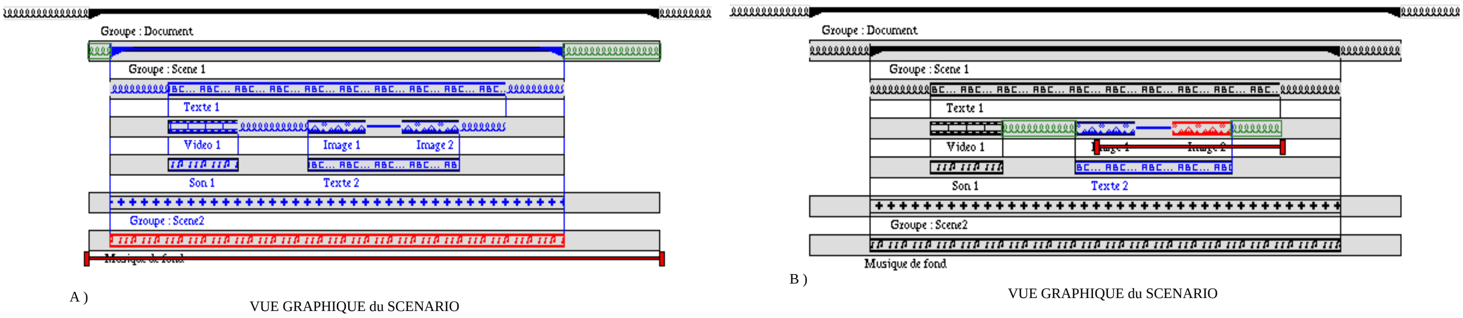


Fig. 3.15 : Sélection d'éléments

Nous avons choisi de représenter explicitement la hiérarchie de manière à offrir à l'utilisateur une vue directe et aisément compréhensible de cette dernière, avec la possibilité de faire des manipulations directes (ouvrir, fermer des objets composites) dessus sans avoir toutes les informations temporelles associées aux objets. Une de nos premières idées avait été de représenter cette hiérarchie sous la forme d'un arbre n–aire. Cependant la difficulté était de faire directement le lien entre la vue hiérarchique et la vue graphique du scénario temporel. Par conséquent, afin de faciliter ce problème de liaison entre les vues, nous avons imaginé une vue hiérarchique qui de part le placement spatial des objets reste très proche de la vue du scénario.

III.5 Des idées à développer

Nous présentons ici les idées que nous avons pour aborder certains des points qui ont été soulevés dans la partie besoin et pour lesquelles nous n'avons pas eu le temps d'apporter une réponse complète. Nous nous plaçons encore ici sous l'hypothèse de durée unique des objets.

Gros volume d'informations

Parmi les points qui nous restent à aborder, l'aspect visualisation de données de dimension hétérogène nous semble pouvoir se régler par l'utilisation de méthodes comme le perspective−wall.

Les opérateurs d'interruption

Un début de solution consiste à rajouter une flèche sur les traits verticaux visualisant un opérateur d'interruption et à trouver un marquage graphique pour l'objet interrompu.

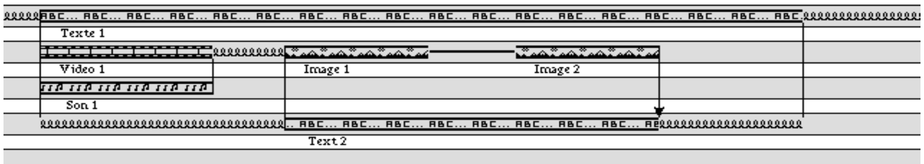


Fig. 3.16 : Visualisation des opérateurs d'interruptions

Il faut cependant trouver un moyen de distinguer les opérateurs d'interruptions tels que le `parmin` et le `parmaster` qui peuvent avoir les mêmes effets. De plus, tant que l'on reste sous l'hypothèse de durée unique des objets cette proposition ne pose pas de difficultés, mais dès que nous allons intégrer des objets à durée variables (contrôlables ou non) établir le sens de la flèche sera une tâche plus complexe car celui−ci pourra changer selon les déplacements des objets.

Les liens de type inclusion

Pour les hyper−liens de type inclusion nous pensons à une méthode relativement proche de ce qui se passe lors de l'exécution du scénario. C'est à dire que nous donnons à l'auteur la possibilité d'activer ou de réactiver les liens. Dans la Fig. 3.17 , nous montrons le cas d'un lien de type inclusion.

Fig. 3.17 : Exemple de liens

Dans le cadre A de la Fig. 3.17 nous voyons un objet de type lien d'inclusion, dans le cadre B nous voyons ce lien activé. L'activation de se lien par l'auteur entraîne l'insertion dans le scénario d'une partie du document. Remarquons que pour les liens références il nous faudra nécessairement réfléchir à un moyen de visualiser la structure de navigation.

Liens entre les fenêtres scénario et exécution

Nos premières expérimentations ont aussi montré qu'il s'avérait utile pour l'auteur de faire le lien entre la vue du scénario la fenêtre dans laquelle le document est joué par Madeus. Comme nous l'avons déjà mentionné, le scénario joué n'est pas nécessairement celui affiché à cause des objets incontrôlables.

Nous avons vu qu'il y avait deux aspects :

- Etant donnée une vue du scénario temporel, il faut pouvoir choisir un instant dans cette vue et visualiser dans la fenêtre d'exécution le placement spatial du document à cet instant.
- Etant données une visualisation graphique du scénario temporel et une exécution, l'auteur doit pouvoir faire le lien entre les deux fenêtres.

Dans le premier cas (Fig. 3.18), ce lien est réalisé au moyen d'une ligne temporelle, qui permet de définir "un instant" dans le scénario temporel, et de placer cet instant dans la fenêtre où nous visualisons l'exécution du scénario. Cela permet d'avoir un "instantané" du scénario.

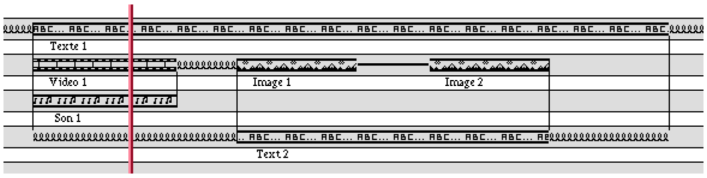


Fig. 3.18 : Liens entre la vue graphique et l'exécution du scénario

Dans le deuxième cas, nous voyons deux types de solutions pour répondre à ce problème : utiliser une autre vue où s'affiche petit à petit la configuration présentée par Madeus, ou bien utiliser une seule vue qui serve à la fois à la visualisation de l'ensemble des scénarios possibles et à faire le lien entre la configuration affichée et celle effectivement présentée. La première solution est plus simple à mettre en œuvre mais l'auteur va devoir faire l'effort de maîtriser deux vues graphiques. En plus il faut maîtriser les problèmes de cohérence entre ces deux vues. Dans la deuxième solution, il va se produire des effets de "sauts" (qui peuvent surprendre l'auteur) lorsqu'un objet va apparaître plus tôt que dans la configuration présentée à l'écran. Dans ce cas un objet de la vue initiale va être brusquement ramené vers la gauche. Prenons par exemple la Fig. 3.19 qui montre une configuration possible satisfaisant les contraintes de notre exemple introductif. La Fig. 3.20 montre la même vue en cours de présentation. La ligne verticale présente l'instant courant et reste fixe sur le graphique, tandis que le scénario se déplace vers la gauche. A cet instant l'objet D vient de terminer sa présentation. A partir de cet instant les objets B et C peuvent à tout moment être présentés, d'où si c'est le cas, l'effet de saut mentionné ci–dessus. De plus, si on adopte cette solution, l'auteur a perdu sa configuration initiale et les possibilités de visualisation qu'il avait dessus.

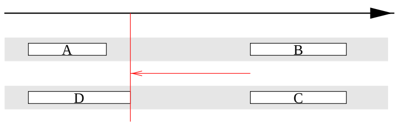


Fig. 3.19 : Problème du lien

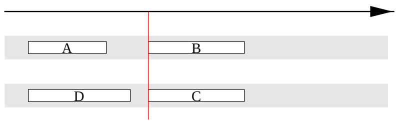


Fig. 3.20 : Problème du lien

III.6 Présentation des algorithmes

Pour mettre en œuvre notre solution il nous a fallu résoudre les problèmes suivants :

- Etant donnés un ensemble de relations entre les objets et un objet sélectionné :
  - calculer les objets solidaires de l'objet sélectionné ;
  - assurer la validité des déplacements de l'objet sélectionné ;
  - mettre à jour l'affichage en répercutant toutes les modifications (sur les objets solidaires et les délais flexibles) le plus rapidement possible.
- Etant donné un ensemble de relations temporelles hiérarchisées, il faut calculer les positions des objets sur les axes temporels en tenant compte du fait que :
  - l'auteur peut ouvrir et fermer des niveaux de hiérarchie ;
  - on peut avoir plusieurs objets (composites ou non) sur le même axe;
  - il faut assurer la validité et la répercussion sur l'affichage des déplacements en tenant compte des déplacements de l'objet sélectionné.

Pour cela nous nous sommes appuyés sur la structure de graphe d'instants qui est construite par le système Madeus à partir du source textuel. Nous présentons brièvement cette structure, sa construction et la structure de données qui lui est associée dans nos algorithmes (section III.6.1). Nous présentons ensuite séparément les algorithmes utilisés pour répondre à chacun des deux points précédemment évoqués (section III.6.2 et section III.6.3). Nous rappelons que ces algorithmes ont fait l'objet d'une implémentation en Tcl/Tk. L'outil de développement d'interface Tcl/Tk est constitué d'un langage de programmation, Tcl (Tool Command Langage), qui est associé à une boite à outils X windows (Tk).

Le problème du choix des objets à placer sur le même axe temporel en étant certain que ceux–ci ne se recouvreront jamais quelques soient les déplacements effectués par l'auteur se règle facilement en s'aidant du graphe temporel.

Une difficulté que nous n'évoquons pas ici car nous ne l'avons pas traitée est le calcul du bon placement initial des objets sur les lignes (axes temporels) de manière à éviter les coupures des objets ou délais par des traits verticaux.

III.6.1 Le graphe temporel

Le graphe temporel est un graphe acyclique orienté, dont les nœuds sont les instants de début et de fin des intervalles et dont les arcs représentent la durée.

Pour expliquer la construction du graphe temporel (pour plus de détails voir [31]) nous allons donner l'idée de la construction incrémentale de ce graphe.

Nous distinguons deux types d'actions :

- Insertion et création d'un objet (cadre 1 de la Fig. 3.21) :

  L'objet est représenté pas un intervalle. Nous créerons donc un arc entre deux nœuds.

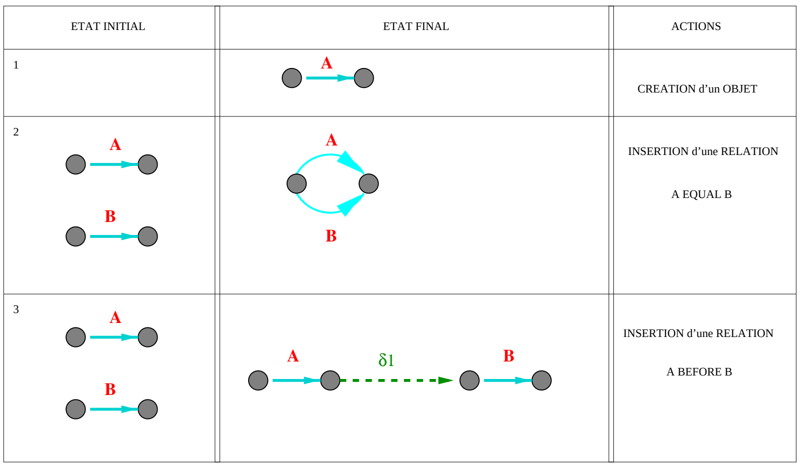


Fig. 3.21 : Construction du graphe temporel

- Insertion d'une relation:

  L'insertion d'une relation entre deux objets a pour effet de modifier le graphe temporel. Ces modifications peuvent être de deux types:

  – Fusion de deux instants (cadre 2 de la Fig. 3.21), c'est par exemple le cas pour les relations start, end, equal...

– Insertion d'intervalle (cadre 3 de la Fig. 3.21), c'est par exemple le cas pour les relations before, after, during ...

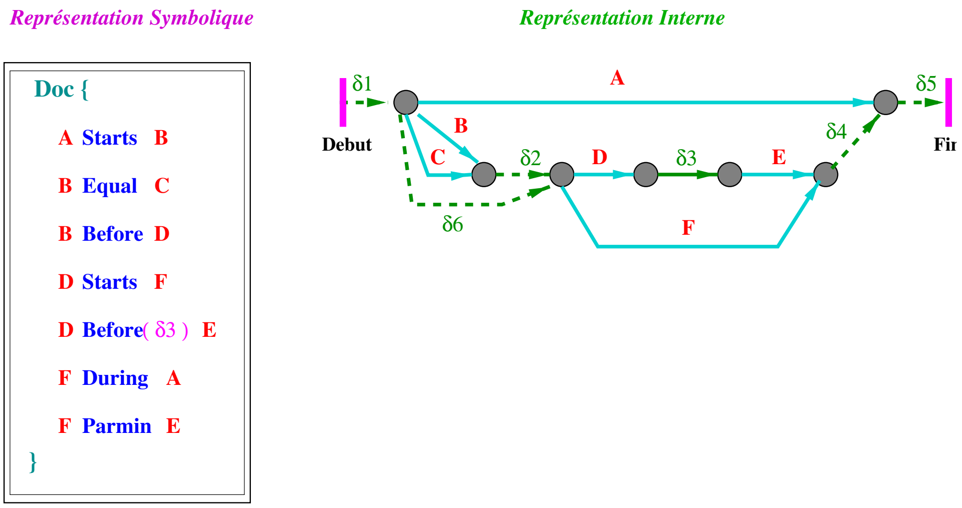


Fig. 3.22 : Le graphe temporel interne

Dans la Fig. 3.22 nous présentons le graphe temporel associé à notre exemple introductif (III.1). Dans le cas général (objets contrôlables et non contrôlables) les arcs sont paramétrés par un triplé de valeurs qui représentent les durées minimale, maximale et optimale.

Nous allons décrire maintenant rapidement la structure de données utilisée;

 # Définition des types propres à la manipulation des graphes temporels

```
Type T_Arc
# un arc relie deux nœuds
   Nœud1, Nœud2 : T_Nœud
   Objet        : T_Objet
```

Type *T_Noeud*
 # un nœud est le point de départ d'une liste d'arcs,
 # Les arcs sont orientés dans la SDD (structure de données) en fonction du placement
 # temporel des objets

```
    L_arc : Liste( T_Arc)
```

Type *T_Objet*
 # Les objets ont des informations qui leurs sont
 # propres:
 # – Durées (min,max, idéale, réelle)

 # − Position
 # et un lien vers l'arc auquel ils sont associés

```
Durees       : T_Info
Position     : T_Info
Arc          : T_Arc
```

Type *T_Hiérarchie*
 # La structure T_Hiérarchie permet de manipuler les informations
 # sur les éléments de la hiérarchie. Plusieurs informations sont sauvées :
 # − les fils du niveau
 # − les positionsabsolues (qui seront mises à jour par lesystème) et relatives du niveau
#

```
Xrel, YRel  : T_Info  # Les positions relatives
Xabs, Yabs  : T_info  # les positions absolues
```

III.6.2 Algorithmes liés au déplacement d'un objet

 # Nous allons utilisé par la suite différentes fonctions que nous ne détaillerons pas :

Liste_arc : qui à partir d'un nœud renvoie la liste d'arc associés à ce nœud.

Type : permet de savoir le type de l'objet, c'est−à−dire si c'est un intervalle flexible ou pas.

Marque_Existe : qui nous dit si un nœud est marqué.

Marque : Marque un nœud.

Objet_Associer : renvoie l'objet associé à un intervalle.

Arc_Associer : renvoie l'arc associer à deux nœuds.

Orientation : nous permet de savoir si nous parcourons l'arc dans le sens du temps.

Initialisation_varialbes_locales : met à jour toutes les variables locales nécessaires.

DeplacementPermis : permet de savoir si l'auteur veut déplacer l'objet sélectionné dans une zone autorisée.

CorrectionNecessaire : permet de savoir si une correction est nécessaire.

 # nous expliquons par la suite dans quel cas nous avons besoin d'effectuer une correction.

 # Pour toutes les fonctions de mise à jour de variables, nous nous sonmes permis
 # quelques libertés, comme l'utilisation d'une fonction unique Mise_A_jours
 # qui permet d'effectuer ces mises à jours.

 # La procédure Calcul_structure_déplacement est le cœur du noyau
 # fonctionnel du programme. Elle permet, à partir d'un objet selectionné de
 # calculer tous les objets qui vont se déplacer, ainsi que les intervalles
 # qui vont évoluer au cours des déplacements.

 # La liste des objets qui vont se déplacer est sauvegardée dans la variable : Liste_Qui_Bouge

 # Les délais qui amortissent les déplacements sont sauvés dans deux listes.
 # Ce choix s'explique par le fait que certains objets vont s'allonger et d'autre
 # rétrécir lorsque l'objet sélectionner va se déplacer. Le comportement des
 # objets est lié au sens de déplacement de l'objet sélectionné. En effet, lorsque
 # ce sens s'inverse le comportement des délais s'inverse aussi.
 # Les délais qui amortissent sont donc sauvegarder dans deux listes :
 # – Liste_Amortie _Droite , pour les délais qui vont rétrécir
 # quand l'objet sélectionner va se déplacer à droite, et s'allonger quand il se
 # déplacer à gauche
 # – Liste_Amortie _Gauche, pour les objets qui délais qui vont rétrécir
 # quand l'objet sélectionner va se déplacer à gauche.
 # Les objet qui bloque le déplacement sont sauvés dans deux listes:
 # – ListeBloqueD (droite)
 # – ListeBloqueG (gauche)

Procedure Calcul_structure_deplacement
 # Définition des paramètres

```
in  =>  Ob : T_Objet
        G  : T_Graphe temporel
out =>

# Définition des variables locales
Soient :
      Nœud1 := Premier_Noeud (Inter)
      Nœud2 := Second_Nœud(Inter)

DébutProc

      Pourchaque Elmt de Liste_Arc(Nœud1) faire

      # On parcourt tous les arcs qui partent du
      # nœud1

            # On insert des marques pour éviter de
            # boucler dans le graphe

            Si MarqueExiste(Elmt)faire
                  Sortir_de_cette_iteration
                  # On passe à l'élément suivant

            Sinon faire
                  Marquer(Elmt)
```

```
FinSi

Selon Type(Elmt)

     cas DelaiVariable faire:
          Soit
               tmp := Objet_Associer(Nœud,Elmt)
               arc := Arc_Associer(Elmt,tmp)

          Selon Orientation(arc)

               cas Elmt -> tmp faire :

               # Le delai va évoluer au cours
               # du déplacement, on l'insert
               # donc dans la liste AmortieG

                    Met_a_Jour(L_AmortieG,tmp)

               # Le delai, s'il est le
               # plus petit des delais,
               # sera le delai bloquant,on
               # met cette information a jour.

                    Met_a_jour(DeltaXg,tmp)

               # On rajoute le délai dans la
               # liste des délais qui évoluent.
               Soit:
               tmp:=Objet_Associer(Nœud,Elmt)

                    Met_a_jour(L_BloqueG,tmp)

               cas tmp -> Elmt faire :

               # Le délai va évoluer au cours
               # du déplacement, on l'insert
               # donc dans la liste AmortieD

                    Met_a_Jour(L_AmortieD,tmp)

               # Le délai, s'il est le
               # plus petit des délais,
               # sera le délai bloquant,on
               # met cette information à jour.

                    Met_a_jour(DeltaXd,tmp)

               # On rajoute le délai dans la
               # liste des délais qui évoluent.
```

```
                    Soit:
                    tmp:=Objet_Associer(Nœud,Elmt)

                         Met_a_jour(L_Bloqued,tmp)

               FinSelon

          cas Video, Son, Image, Texte, DelaiFixe  faire :
               Soit :
                    tmp := Objet_Associer(Nœud,Elmt)

               # Dans le cas ou on a un objet qqc,
               # cet objet suivra le déplacement
               # on le rajoute donc dans la liste

               Met_A_Jour(Liste_Qui_Bouge,tmp)
               Calcul_structure_deplacement(tmp,G)
     FinSelon

FinPourChaque

# on traite de façon relativement similaire le
# deuxième nœud

FinProc
```

 # Quand on sélectionne un objet, on calcule la structure de déplacement

 # cette structure ne sera plus calculée au cours du déplacement

 # elle sera mise à jour à la fin du déplacement

Procedure Debut_deplacement

```
in  =>  Ob : T_Objet
        G  : T_Graphe temporel
out =>
DebutProc
     Calcul_structure_deplacement(Ob,G)
FinProc
```

 # La fonction déplacement sera appelée à chaque traitement du

 # mouvement de la souris. L'information récupérée sur le déplacement de

 # la souris n'est pas livrée (donc traitée) en continu.

 # La fonction fera les corrections nécessaires et mettra à jour

 # l'affichage graphique. Des corrections sont dues à la discontinuité dans

 # le traitement des mouvements de la souris.

 # Prenons par exemple le cas de deux objets A et B, reliés par
 # un " A before B", placés sur la même ligne et supposons que nous déplaçons
 # B vers A. Il se peut qu'entre deux traitements de la souris, la souris ait
 # atteint une position invalide. Dans ce cas, il ne faut pas que l'objet
 # suive la souris, ni qu'il reste à la dernière position valide connue. Il faut
 # qu'il se "replace" depuis la dernière position connue jusqu'à la
 # dernière position valide parcourue par la souris.
 # Lorsque l'objet ne peut plus se déplacer, il est "bloqué".

P**rocedure** Deplacement

```
in  =>  Ob     : T_Objet
        Delta  : T_Intervalle
out =>


DebutProc

     Si DeplacementPermis(Delta)faire

          Si CorrectionNecessaire(Delta)
               FaireCorrection ;
          FinSi
          # on met à jours les objets dont la taille ou la position a
          # évoluée.

          Met_a_Jour(Affichage)

     FinSi

FinProc
```

 # A la fin du deplacement nous mettons à jours la SDD

Procedure Fin_deplacement

```
in  =>  Ob : T_Objet
        G  : T_Graphe temporel
out =>


DebutProc
     Mettre_A_Jour(SDD,Xoriginal,DeltaX)
FinProc
```

III.6.3 Algorithmes liés à la hiérarchie

 # La procédure Calcul_Position_relative permet de mettre à jour la SDD. Elle calcule la

 # position absolue des objets affichés en fonction des niveaux de hiérarchie ouverts.

Procedure Calcul_Position_Relative

```
in  =>  Niveau : T_Hierarchie
out =>

DebutProc
      Initialisation_Variables_Locales

      Si Type(Niveau) = Elementaire

      # Nous sommes dans le cas où nous pointons sur un objet
      # élémentaire,toutes les variables globales ont été
      # mises à jour au niveau supérieur, il suffit donc de
      # mettre à jour la position en fonction de la
      # position relative de l'objet (en X) et de son décalage
      # (en X et en Y).

            Mettre_A_Jour(SDD,Niveau.Xr,Xabs)
            Mettre_A_Jour(SDD,Niveau.Yr,Yabs)

      Sinon # Type(Niveau) = Complexe

            Si Etat(Niveau) = Fermé

            # Nous sommes dans le cas ou nous sommes sur un
            # un niveau de hiérarchie ferme.
            # Toutes les variables globales ont été mises à jour

            # au niveau supérieur, il suffit donc de mettre a
            # jours la position en fonction de la position
            # relative de l'objet (en X) et de son décalage
            # (en X et en Y).

                  Mettre_A_Jour(SDD,Niveau.Xr,Xabs)
                  Mettre_A_Jour(SDD,Niveau.Yr,Yabs)

            Sinon /* Etat(Niveau) = Ouvert */

            # Nous sommes dans le cas ou nous sommes sur un
            # un niveau de hiérarchie ouvert.
            # Il faut calculer les informations dans tous les
            # niveaux inférieurs et répercuter les informations
            # nécessaires à la suite du traitement
            # Il faut aussi mettre à jours les informations sur
            # le nœud courant.

                  # Mise à jour des informations locales
```

```
Mettre_A_Jour(SDD,Niveau.Xr,Xabs)
Mettre_A_Jour(SDD,Niveau.Yr,Yabs)

# On effectue un tri des fils du nœud courant
# selon un ordre spécifique.
# Dans un premier temps on les classe par ligne
# et dans un second temps, à l'intérieur de ces
#  lignes, on les classe dans l'ordre croissant

# de leur apparition (dans le temps).
# Le fait de manipuler une solution et
# un découpage en lignes fait que cet ordre
# est total à l'intérieur d'un niveau de
# la hiérarchie.

Tri_des_Fils_Ligne_Placement_Temporel (Niveau)

# On effectue une sauvegarde de la valeur du
# Y globale de manière à pouvoir la passer
# au frère du niveau courant pour la suite
# du traitement.

Sauvegarde(Xabs,Yabs)

# Met a jours le Xglobal pour les fils en
# fonction de la position du père.

Mettre_A_Jour(SDD,Xabs,Niveau.Xr)

# on effectue le traitement pour chaque fils,
# en  fonction de l'odre calculé précédemment.

Pourchaque fils de Niveau faire

    Sauvegarde(Xabs)
    Mettre_A_Jour(SDD,Xabs,Niveau.Xr)

    # Si l'objet est le premier d'une ligne on
    # incremente le Y global

    Si PremierLigne( fils)
        Mettre_A_Jour(SDD,Yabs)
    Fin Si

    # sauvegarde le Y global pour les freres
    # du fils

    Sauvegarde(Yabs)

    # Calcul recursif du placement
```

```
                    ResLoc := Calculer_Position_Relative (fils)

                    # On sauvegarde la valeur de la ligne
                    # la plus basse du niveau courant,
                    # pour pouvoir dessiner une
                    # boite englobante

                    ResAbs := Maximum (ResAbs, ResLoc)

                    # on restaure les valeurs de X et Y pour
                    # la suite du niveau.

                    Restaurer(Xabs)
                    Restaurer(Yabs)
                Fin PourChaque

                # On restaure les valeurs pour le traitement
                # des frères du niveau courant

                Restaurer(Xabs)
                Mettre_A_Jours(SDD, DerniereLigne, ResAbs)
                Mettre_A_Jours(SDD,Yabs,ResAbs)

            Fin Si
        Fin Si
    FinProc
```

III.7 Premiers retours utilisateurs

La réalisation de ce prototype a permis d'effectuer un début d'évaluation. Dans la mesure où l'évaluation n'a pas été effectuée avec toutes les règles requises il faut considérer les résultats qui suivent comme des premières indications. Le but était d'évaluer notre interface de visualisation du scénario temporel dont l'objectif est d'aider l'auteur à appréhender le comportement du scénario temporel. Pour cela nous avons réalisé des sessions de tests. Nous avons demandé à différentes personnes (une dizaine), de profil différent, d'utiliser notre prototype. Après une brève description de ce qu'était un scénario temporel, nous leur avons fourni une visualisation d'un scénario temporel test (scénario relativement complexe) sans leur montrer la vue du source textuel. Nous leur avons alors demandé de nous décrire le scénario temporel correspondant et de nous expliquer son comportement. Les résultats ainsi obtenus ont permis de tirer plusieurs conclusions :

- Les personnes habituées à manipuler des scénarios temporels n'ont eu aucune difficulté à expliquer et justifier le comportement du scénario.
- Les personnes non habituées à manipuler des scénarios temporels sous forme de contraintes ont eu du mal, au départ, à appréhender la sémantique des contraintes mais après quelques manipulations elles sont arrivées à décrire le scénario et à le justifier.
- Les ergonomes (du projet Airelle) ont apprécié la différenciation graphique faite entre les délais flexibles et les délais fixes.

On peut dire que ces premiers résultats sont très encourageants mais restent à valider de manière plus rigoureuse.

III.8 Conclusion

Au cours de ce chapitre nous avons présenté les différents besoins liés à la visualisation graphique du scénario temporel en insistant sur les difficultés que soulèvent ces besoins (section III.2). Nous proposons ensuite une solution qui, bien qu'elle ne réponde pas à tous les besoins évoqués, permet de poser un principe de base : visualiser à tout instant un placement temporel possible des objets et donner à l'auteur tous les moyens d'accéder simplement à tous les autres. Cette proposition a fait l'objet d'une mise en œuvre en Tcl/Tk. Nous avons présenté (section III.6) les algorithmes à la base de cette implémentation. Les résultats d'une première évaluation informelle de notre proposition terminent ce chapitre.

Les points importants qui se dégagent de cette solution sont :

- L'observabilité qui est mise en évidence par :
  - La facilité de compréhension de la vue graphique du scénario grâce à sa correspondance directe avec la vue textuelle du scénario. Cette correspondance est de plus accessible par l'auteur grâce aux mécanismes de liaison entre les deux vues (section III.2.6).
  - La bonne lisibilité de la vue, par le choix de ne donner certaines informations que lorsque l'auteur sélectionne un objet.
  - La qualité de perception de l'ensemble des solutions associées à un scénario donné qui est obtenue par plusieurs facteurs :
    + la visualisation des contraintes qui permet d'expliquer pourquoi une configuration est ou n'est pas possible ;

    + le rendu en temps réel des conséquences des déplacements des objets.

- L'honnêteté du système, en effet le positionnement spatial des objets est le reflet exact de leurs contraintes temporelles. De plus, leur taille est proportionnelle à leur durée.
- La prévisibilité du système, en effet lorsque l'auteur sélectionne un objet, il peut anticiper le comportement de son scénario, car il visualise l'intervalle de déplacement de l'objet sélectionné ainsi que les objets qui vont se déplacer avec lui.
- Le lien entre les vues, qui permet à l'auteur de comprendre facilement l'influence d'une contrainte.

Concernant la capacité de notre proposition à s'adapter à de gros volumes de données la solution que nous avons choisie pour intégrer la hiérarchie à la visualisation d'informations temporelles nous permet d'espérer des résultats satisfaisants. En effet l'auteur peut sans restriction fermer et ouvrir des niveaux de hiérarchie.

Chapitre III

Visualisation du scénario temporel au sein de Madeus

Conclusion

1 Synthèse

La spécification du scénario temporel d'un document multimédia occupe une place majeure lors de l'édition d'un tel document, de part son caractère novateur (cette notion de scénario n'existe pas dans les documents usuels) et sa complexité.

Différentes approches ont été proposées pour mener à bien cette tâche qui vont de l'utilisation d'un langage de script à l'utilisation d'un langage déclaratif à base de contraintes, en passant par l'utilisation de structures graphiques comme les arbres ou les réseaux de Pétri. Au cours de ce stage, notre intérêt s'est porté sur les approches fondées sur l'utilisation de contraintes et plus précisément sur celle proposée par Madeus l'environnement d'édition/présentation de documents multimédia développé au sein du projet Opéra.

Dans ce contexte, nous nous sommes intéressés à la définition d'une vue graphique du scénario qui soit une véritable aide pour l'auteur. Par conséquent, nous avons souhaité aller plus loin que la définition d'un équivalent graphique à chaque contrainte, comme cela est proposé dans les autres environnements d'édition de documents multimédia à base de contraintes. Cette représentation graphique se veut être utile et efficace en offrant une réponse visuelle aux questions que se pose l'auteur sur le plan de la compréhension et de l'analyse du scénario temporel du document multimédia.

Notre proposition s'appuie sur une étude détaillée des solutions de visualisation graphique du scénario temporel d'un document multimédia offertes dans d'autres environnements d'édition (chapitre I), ainsi que sur une étude d'un ensemble de techniques de visualisation pertinentes par rapport à notre problématique (chapitre II). De ces études nous avons retenu les éléments suivants :

- La pertinence d'une représentation de type "time−line" pour les enchaînements temporels.
- L'impossibilité d'utiliser des techniques classiques de visualisation (hiérarchie arborescente) et l'enjeu que représente la solution apportée à

ce point particulier par rapport au problème général de passage à l'échelle de la solution.

- Les nombreuses difficultés soulevées par la visualisation de contraintes (présence de contraintes induites, explication des causes,...) et le peu de travaux effectués dans ce domaine.

Afin d'avoir une vue globale du problème, nous avons tenu à établir une liste aussi complète que possible de l'ensemble des besoins soulevés par ce sujet (section III.2.1), même si tous ces besoins ne sont pas couverts pas notre proposition. Une restriction faite au problème général est de considérer une définition unique, connue en phase d'édition, de la durée des objets.

Le principe sur lequel repose notre solution est le suivant : visualiser un placement des objets qui respecte les contraintes du scénario, en expliquant pourquoi celui–ci est possible, et donner la possibilité à l'auteur de déplacer les objets dans la limite autorisée par le scénario, afin de parcourir l'ensemble des placements possibles. Communiquer des idées complexes comme les concepts de flexibilité avec clarté, précision et efficacité est une entreprise difficile. Afin d'isoler le problème de la visualisation de la hiérarchie identifié après notre étude bibliographique comme étant un des points clés du problèm, nous avons travaillé en deux temps :

- proposition d'une solution sans hiérarchie ;
- intégration de la hiérarchie à la solution précédente.

Cette solution fait l'objet de tests grâce à la réalisation d'un prototype indépendant de Madeus, en utilisant le langage de prototypage d'interface Tcl/Tk. Comme il est montré dans la section III.6, un soin tout particulier a été accordé aux performances temps réel de notre réalisation. Le résultat est une maquette (évolutive) qui produit une visualisation satisfaisante (les premiers retours de la part des utilisateurs sont très encourageants) expérimentée sur de nombreux scénarios et permettant la visualisation de toutes les relations d'Allen utilisées dans Madeus. Les structures de données utilisées sont compatibles avec celles du prototype Madeus. Il sera donc aisé d'intégrer ces travaux au sein de Madeus (travail prévu comme stage d'été).

Pour conclure, remarquons que le travail effectué au cours de ce stage bien qu'il ait été fait dans le cadre de l'environnement Madeus, s'adapterait aisément aux autres environnements d'édition fondés sur les contraintes d'intervalles : Isis [43] et HPAS [51].

2 Perspectives

Le travail effectué au cours de ce stage doit être vu comme la première phase d'une réflexion dont l'objectif à long terme est l'édition, par manipulation directe, du scénario temporel d'un document multimédia. Ceci explique pourquoi l'étude bibliographique, qui représente une part non négligeable du travail, couvre des aspects qui ne sont pas abordés dans notre proposition. De ce fait, les perspectives ouvertes par ce travail sont nombreuses et plus précisément liées aux besoins mis en évidence mais à ce jour non traités: la flexibilité des objets, les opérateurs d'interruptions, la structure de navigation,...

Nous allons devoir réfléchir sur la façon d'intégrer ces différents aspects tout en essayant de conserver le principe de la proposition présentée ici: visualisation à tout moment d'une solution possible et atteignabilité de toutes les autres par manipulation.

Parallèlement il s'avère indispensable de réaliser une véritable évaluation ergonomique de la solution de visualisation proposée, notamment en collaboration avec l'équipe Airelle d'André Bisseret.

L'étape suivante consistera à rendre cette vue utilisable directement comme vue d'édition.

Bibliographie

[1] ADOBE System Inc, http://www.adobe.com.

[2] AOL Press, http://www.aolpress.com.

[3] ALLEN J.F., ''Maintening Knowledge about Temporal intervals'', *Communications of the ACM*, Vol. 26 (No 11), pp. 832–843, november 1983.

[4] BIER E., STONE M,, ''Snap Dragging '', *Computer Graphics*, vol. 20(Num. 4), pp. 233–240, Proceeding SIGGRAPH'86 1986.

[5] BUCHANAN C. Zellweger P.T., ''Specifying Temporal Behavior in Hypermedia Documents '', *Proceeding of the ACM, Conference on Hypertext*, pp. 262–271, December 1992.

[6] BULTERMAN D. and HARDMAN L., ''Multimedia Authoring Tools: State of Art and Research Challenges'', Springer LNCS(1000), pp. 575–591, 1995.

[7] CABRI GEOMETRE, http://www.cabri–imag.fr.

[8] CARCONE L., *Gestion de contraintes spatiales dans Madeus*, Mémoire d'ingenieur CNAM, (à paraître), Grenoble 1997.

[9] CARRIERE J. KAZMAN R., ''Interacting with Huge Hierarchies: Beyond cone Trees'', *Proceedings of the IEEE symposium on Information visualization*, pp. 74–81, Atlanta, Octobre 1995.

[1 0] COUTAZ J, ''Interface homme–ordinateur: conception et réalisation'', *Ed Dunod informatique*, 455 pages ,1990.

[1 1] CONKLIN J., ''Hypertext: An introduction and survey'', *IEEE Computer*, 20 (9), pp. 17–41, September 1987.

[1 2] DEL BIMBO A., VICARIO E., ''Visual Programming of Virtual Worlds Animation'', *IEEE Multimedia*, Vol 3(n 1), pp. 40–49, Spring 1996.

[1 3] DUDA A. et KERAMANE C., ''Structured Temporal composition of Multimédia Data'', *Multimédia Database*, pp. 136–142, August 1995.

[1 4] Director, *Manuel d'apprentissage de DIRECTOR*, MACROMEDIA 1996.

[1 5] ELLIOT G.J., Jones E., ‘‘Authoring semantic hypermedia: a concept mapping approach ’’, *ED–Media et Ed–Telecom96*, Boston, june 96.

[1 6] FURNAS G., ‘‘Generalised Fisheye views’’, *In proceedings oh CHI’86 Human factor in Computing Systems*, pp. 16–23, 1986.

[1 7] FURUTA R., QUINT V. et ANDRE J., ‘‘Interactively Editing Structured Documents’’, *Electronic publishing – Origination*, 1 (1), pp. 19–44, April 1998.

[1 8] GLEICHER M, WITKIN A., ‘‘Drawing With Constraints’’, *The visual computer*, 11 (1), pp. 39–51, 1994.

[1 9] GRAM C.and COKTON G., *Design Principles for interactive software*, Christian Gram and Gilbert Cockton, Chapitre 5 pp 133–187.

[2 0] GROSS M., ‘‘Graphical Constraints with CoDraw’’, *Visual Langage 92*,

[2 1] HARDMAN L., VAN ROSSUM G., BULTERMAN D.C.A., ‘‘Structured Multimédia Authoring’’, *Proceedings of the first ACM International Conference on Multimédia*, pp. 283–289, Anaheim , California, August 1993.

[2 2] HODGES M.E., HEINES R.M. and ACKERMAN M.S., ‘‘A Construction Set for Multimedia Applications ’’, *IEEE Software*, 6 (1), pp. 37–43, Janvier 1989.

[2 3] HyTime Information Technologie, ‘‘Hypermedis/ Time–based Structuring Langage (HyTime)’’, *ISO/IEC 10743*, November 1992.

[2 4] IBM Corp, *"AVC: Audio Visual Connection"*, User’s Guide 1990.

[2 5] IconAuthor 6.0, *User’s Guide*, Aim Tech.

[2 6] JOURDAN M., LAYAIDA N., SABRY–ISMAIL L., ‘‘Time Representation and Management in MADEUS: an authoring environment for multimedia documents’’, *Multimedia Computing and Networking 1997*, pp. 68–79, February 1997.

[2 7] KERAMAN C, DUDA Andrzej, ‘‘Interval Expressions–a functional Model for interactive Dynamic Multimedia Presentation’’, *IEEE ICMCS’96*, 1996.

[2 8] KIM M. and SONG J., ‘‘Multimedia Documents with Elastic Time’’, *Proceedings of the third ACM International Conference on Multimedia, edite par ACM Press*, pp. 143–154, Polle Zellweger, San Fransisco, Californie 5–9 novembre 1995.

[2 9] KIMBER D.G., WILCOX L.D., CHEN F.R. and MORAN T.P., ‘‘Speaker Segmentation for Browsing Recorder Audio’’, *CHI’95*, pp. 212–213, May 1995.

[3 0] LAYAIDA N, SABRY–ISMAIL L., ‘‘Maintaining Temporal Consitency of Multimédia Documents using Constaint Networks’’, *Multimedia Computing and Networking 1996*, M. Freeman, P. Jardetzky H. M. Vin ed.pp. 124–135, January 1996.

[3 1] LAYAIDA N., *Madeus: un système d’édition et de présentation de documents structurés multimédia*, Thèse de doctorat, Université Joseph Fourier, Juin 1997 (à paraître).

[3 2] LAYAIDA N., VION–DURY 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.

[3 3] MACKINLAY J.D., ROBERTSON G.G. , CARD S.D., ‘‘The Perspective Wall: Detail and context smoothly integrated’’, *Proceedings of CHI’91 Human facrtors in Computing Systems*, pp. 173–179, Addison Wesley 1991.

[3 4] Microsoft Project, Microsoft, http://www.microsoft.com.

[3 5] Multimedia Toolbook 3.0, *User’s Guide*, Asymetrix.

[3 6] MYERS B.A., ‘‘Taxonomies of visual programming and programm visualization’’, *Journal of Visual Langages and Computing*, vol 1 pp. 97–123, mars 1990.

[3 7] NANARD J. and NANARD M., ‘‘Using Structured Types to incorporate Knoledge in Hytpertext’’, *in Proceedings of Hypertexte’91*, ACM PRESS 1991.

[3 8] OUSTERHOUT J.K., *Tcl and the Tk Toolkit*, Addison–Wesley, 1993.

[3 9] QUINT V., Vatton I., ‘‘Combinig Hypertext and Structured Documents in Grif’’, *Proceeding of the ACM, Conference on Hypertext*, pp. 23–32, December 1992.

[4 0] ROBERTSON G, CARD S., MACKINLAY J., ‘‘Cones Trees: Animated 3D Visualizations of Hierarchical Information’’, *Proceedings of CHI’91*, pp. 189–194, 1991.

[4 1] SABRY L., *Langage de specification de la structure temporelle des documents multimedia*, DEA Informatique, Université Joseph Fourier, Grenoble , Juin 1995.

[4 2] SARKAR M., BROWN M., ‘‘Graphical Fisheye Views’’, *Communications of the ACM*, Vol. 37 (Num. 12), pp. 73–84, 1994.

[4 3] SONG J., KIM M., RAMALIGAN G. et MILLER R., ‘‘Interactive Authoring of Multimedia Documents’’, *Visual Language 96* , 1996.

[4 4] Tcl/Tk , *http://sunscript.sun.com/.*

[4 5] TUFTE R., *The Visual Display of Quantitative Information*, Graphics Press, Cheshire, CT, 1983.

[4 6] VAN ROSSUM G. and al, ‘‘CMIFed: A presentation environment for portable hypermedia documents ’’, *Proceedings of the first ACM International Conference on Multimédia*, pp. 183–188, august 1993.

[4 7] VERNIER F., *Etude et mise en œuvre de techniques d’interaction pour les moteurs de recherche du World Wide Web*, Rapport de Magistère d’informatique 2ème année, Université Joseph Fourier, Grenoble 1996.

[4 8] XEROX, ‘‘Cones tree’’, http://http://www.inxight.com/products/visual/overview.shtl.

[4 9] XEROX, ‘‘Hyperbolic Tree’’, http://www.inxight.com/products/visual/overview.shtl.

[5 0] XEROX, ‘‘Perspective Wall’’, http://www.xerox.com/products/visual–recall/ treeeview.hrml.

[5 1] YU J. and XIANG Y., ‘‘Hypermedia Presentation and authoring System’’, *Proceedings of the sixth international World Wide Web*, 7–11 Avril, Santa Clara, USA.

[5 2] ZIZI M., BEAUDOIN–LAFON M., ‘‘Accessing Hyperdocument through Interactive Dynamic Maps’’, *ECHT ’94 Proceedings*, pp. 126–134, September 1994.