Chapitre III : Spécification d'un environnement auteur idéal

1. Introduction

Au cours du chapitre précédent, nous avons pu voir les différents formalismes utilisés pour définir des documents multimédias, les types d'environnement d'édition qui existent, ainsi que les différents besoins de l'auteur au cours du processus d'édition. Nous avons vu comment ces besoins étaient plus ou moins couverts par les environnements existants. À partir de ces informations, nous allons à présent définir ce que nous considérons être l'environnement d'édition de documents multimédias idéal, c'est-à-dire celui satisfaisant au mieux les besoins des auteurs identifiés auparavant.
Pour cela nous nous inspirerons des différents systèmes auteur de documents multimédias présentés, mais ces derniers n'apportant pas une réponse satisfaisante à tous les besoins, nous élargirons notre domaine de référence pour chercher des idées de solutions dans des domaines voisins comme l'édition de texte, d'hypertexte ou de vidéo. Dans ce chapitre, nous allons essentiellement nous intéresser aux fonctionnalités que doit fournir un tel environnement. Pour le moment, nous ne nous intéresserons pas ou peu aux aspects graphiques de l'interface. Ces aspects ne sont pas indépendants, mais nous nous focaliserons sur les fonctions intrinsèques de base.
Dans un premier temps, nous présenterons les principes généraux de cet environnement (section 2).
Nous validerons ensuite cette proposition d'environnement idéal en indiquant comment il permet de répondre aux différents besoins de visualisation (section 3), ainsi qu'aux différents besoins de navigation de l'auteur (section 4). Et nous terminerons ce chapitre par l'illustration de l'utilisation de cet environnement auteur au travers d'une session d'édition (section 5).

2. Spécification de l'architecture de l'environnement idéal

Nous allons présenter dans un premier temps l'architecture générale de cet environnement d'édition idéal (section 2.1). Dans un deuxième temps, nous spécifierons le formalisme d'édition que nous utiliserons dans cet environnement d'édition (section 2.2). Dans un troisième temps nous nous intéresserons aux modes de visualisation que cet environnement doit fournir (section 2.3) et enfin dans une dernière partie nous présenterons les différents services d'aide que devra fournir cet environnement pour faciliter la tâche d'édition de l'auteur (section 2.4).

2.1 Rappel du cycle d'édition et architecture générale

Avant d'aller plus loin, nous rappelons le cycle d'édition défini dans le chapitre précédent (voir chapitre II section 2.2.1), ce processus se décompose suivant cinq étapes :

1. Ouverture du document ;
2. Visualisation du document et navigation à l'intérieur des différentes informations ;
3. Modification du document ;
4. Vérification de la cohérence et formatage ;
5. Sauvegarde du document.

De ce processus de base, on a proposé une architecture générale de l'environnement auteur qui se découpe en quatre grandes fonctions (voir chapitre II section 2.2.1 et Figure III-1) :

- Gestion de fichier ;
- Visualisation d'information ;
- Opération d'édition ;
- Aide à l'auteur.

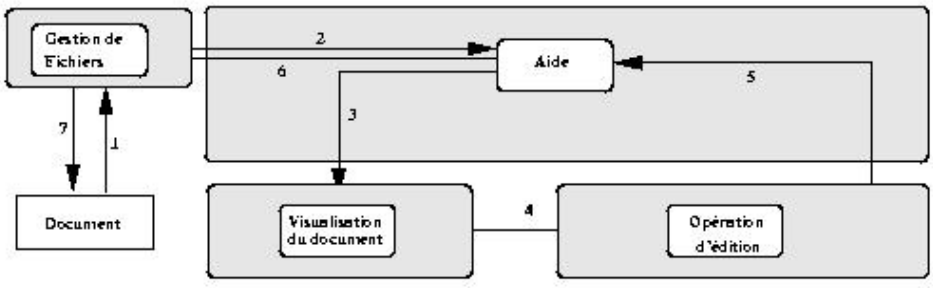


Figure III-1 : Cycle d'édition simple

Dans le chapitre précédent nous avons proposé cette architecture générale pour un environnement d'édition de documents multimédias. L'environnement auteur idéal que nous allons proposer est basé sur cette architecture. Au cours de ce chapitre, nous allons nous intéresser essentiellement à la définition du formalisme d'édition de cet environnement. Ce formalisme est le coeur de l'environnement d'édition. Nous nous intéresserons dans un deuxième temps aux services d'aide à l'édition que doit fournir un environnement auteur.

2.2 Spécification d'un formalisme d'édition

Nous allons dans un premier temps préciser quel type de formalisme d'édition nous allons choisir et justifier notre choix (section 2.2.1). Dans un deuxième temps, nous présenterons ce formalisme d'édition (section 2.2.2), et enfin, dans une troisième partie, nous identifierons les services que devra fournir l'environnement auteur idéal pour éditer ce formalisme (section 2.2.3).

2.2.1 Motivations

Comme nous avons pu le voir dans le chapitre précédent (section 3), il existe de nombreux langages pour décrire des documents multimédias.

La première solution pour réaliser un environnement auteur consiste à utiliser le formalisme du langage de sauvegarde/présentation comme formalisme d'édition. Néanmoins ces langages sont souvent adaptés à la présentation de documents multimédias, mais ils sont en général peu adaptés à l'édition.

Une deuxième approche consiste à réaliser un environnement avec son propre formalisme d'édition et de réaliser des fonctions d'import / export vers d'autres langages. Cependant, l'inconvénient de telles fonctions, est que l'on perd souvent les informations d'édition. En effet, le fichier de restitution essaie de conserver au maximum les informations de présentation, en oubliant généralement les informations qui ont permis d'arriver à cette présentation. Cela pose principalement deux problèmes : l'édition par un tiers du fichier de restitution, et le problème de stockage et de gestion de version des deux fichiers qu'il faut garder (sauvegarde et restitution). Cette situation arrive par exemple, quand une personne écrit ses pages Web avec Word pour les exporter finalement en HTML.

Cette deuxième solution, serait celle qui offrirait la plus grande satisfaction pour le concepteur de l'environnement auteur. Cependant, la diversité des langages actuels est tel que cela est impossible. En effet, le formalisme commun nécessaire serait une combinaison des différents formalismes, et ne permettrait pas de répondre aux besoins des auteurs vis-à-vis de l'édition. En effet, on ne peut pas éditer de la même façon tous les langages, cela ne serait pas pertinent vis-à-vis des formats existants aujourd'hui.

Une solution autre consiste à faire un compromis entre ces deux approches.

Cette solution est illustrée dans la Figure III-2. Elle consiste à faire cohabiter les deux solutions précédentes, c'est-à-dire faire cohabiter à la fois le formalisme de haut niveau proposé par l'environnement, tout en laissant la possibilité à l'auteur d'accéder au langage de présentation.

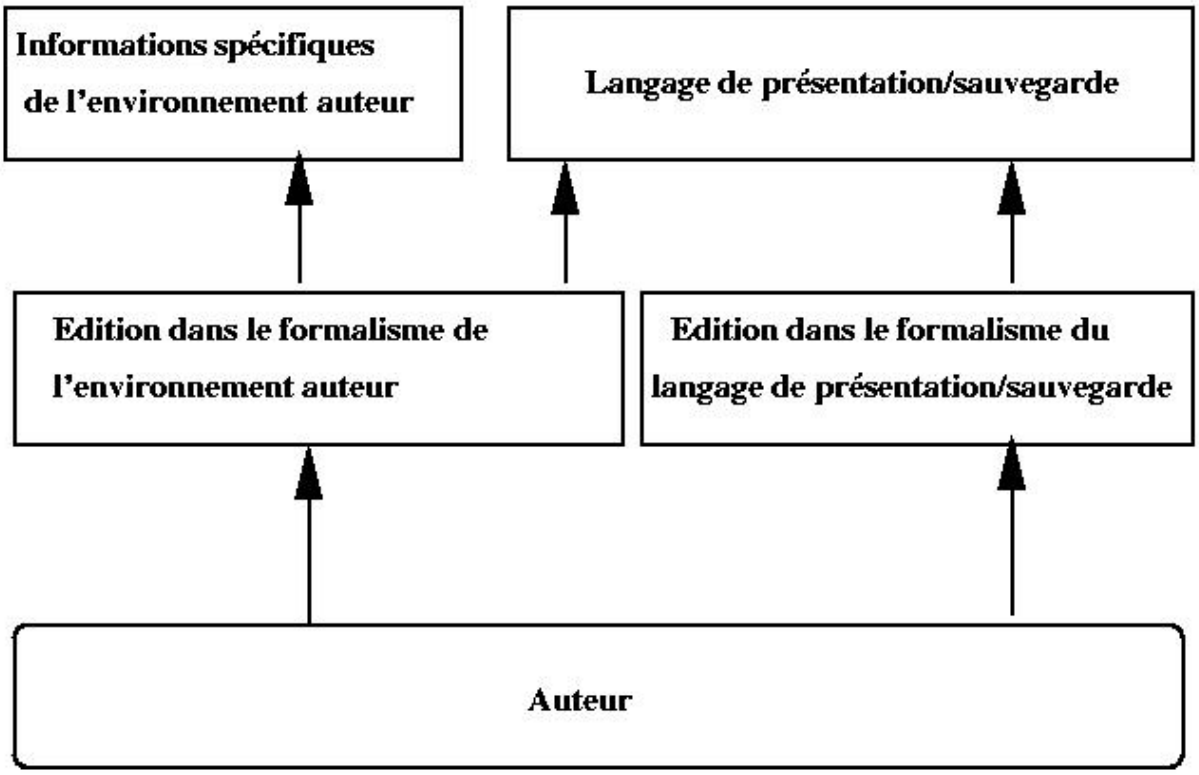


Figure III-2 : Un environnement auteur utilisant plusieurs formalismes

Dans ce cas-là, c'est le système qui s'occupe de conserver la relation entre le formalisme de l'environnement et celui du langage de présentation. Il est nécessaire pour cela de stocker certaines informations complémentaires du fait qu'il n'y ait pas de bijection entre les possibilités du langage et celui de l'environnement. Ces informations complémentaires pourront être stockées à l'intérieur même du fichier par exemple par l'intermédiaire des espaces de noms proposés dans XML. Ainsi, dans l'exemple de la Figure III-3 on peut voir un exemple de fichier SMIL où l'environnement auteur a sauvegardé des informations spécifiques (des relations spatiales).

La manipulation de ces deux formalismes peut rendre plus complexe la tâche de l'auteur. L'environnement doit aider l'auteur dans cette compréhension en offrant par exemple des services de visualisation adaptés.

Nous verrons en conclusion de ce chapitre en quoi ce double formalisme permet de répondre aux différents besoins auteur.

```
<?xml version="1.0"?>

<smil>
<head>
<layout xmlns:EnvAuteur="http//www.inrialpes.fr/opera/EnvA.dtd">
      <region id="région1" top="180" left="80" width="100" height="40" />
      <region id="région2" top="240" left="100" width="100" height="40" />
      <region id="région3" top="300" left="120" width="100" height="40" />
      <region id="région4" top="360" left="140" width="100" height="40" />
      <region id="région5" top="420" left="160" width="100" height="40" />
      <EnvAuteur:Relation>
      <EnvAuteur:LeftAlign EnvAuteur:el1="région1" EnvAuteur:el2="région2">
      </EnvAuteur:Relation>
</layout>
</head>
<body>

<par dur="50" id="Document" >
      <text src="/home/Texte1.html" id="T1" region="region1"/>
      <text src="/home/Texte2.html" id="T2" begin="10.0" region="region2"/>
      <text src="/home/Texte3.html" id="T3" begin="20.0" region="region3"/>
      <text src="/home/Texte4.html" id="T4" begin="30.0" region="region4"/>
      <text src="/home/Texte5.html" id="T5" begin="40.0" region="region5"/>
</par>
</body>
</smil>
```

Figure III-3 : Exemple de fichier utilisant un espace de noms

De plus, l'importance réciproque de ces besoins dépend de l'objectif de l'auteur. On peut définir trois classes d'objectifs lors de la conception d'un document :

- *simplicité d'édition et diffusion*: tout auteur souhaite pouvoir éditer simplement ses documents, pour répondre à cette attente l'environnement auteur idéal fournit un formalisme de haut niveau qui permet de simplifier la phase d'édition. De plus, la sauvegarde des informations d'édition dans le fichier de présentation permet à l'auteur de ne pas avoir le problème de gestion de version entre les différentes formes de son document, problème rencontré dans les approches utilisant des fonctions d'export pour diffuser le document.
- *partage entre plusieurs auteurs du document* : l'environnement auteur basé au dessus d'un standard de spécification garantit à l'auteur une portabilité de son document tout en permettant à d'autres personnes d'éditer son document. Ce besoin arrive essentiellement dans un contexte d'édition du document à plusieurs où l'utilisation d'un format commun impose des contraintes entre les différents utilisateurs.
- *simplicité d'édition, diffusion et partage du document produit*: ce besoin qui est la combinaison des deux premiers est satisfait par le double formalisme offert. En effet, celui-ci permet à l'auteur d'utiliser un standard de spécification, tout en utilisant partiellement les facilités de spécification de l'environnement pour simplifier sa tâche.

En conclusion, le choix que nous allons faire est de proposer un formalisme d'édition propre à l'environnement tout en laissant la possibilité à l'auteur d'accéder au formalisme du langage.

2.2.2 Le formalisme d'édition de l'environnement auteur idéal

Le choix du formalisme que nous allons employer est motivé par l'analyse du chapitre précédent. En effet, des formalismes comme le formalisme absolu ou événementiel sont trop complexes et rendent le document difficilement gérable de manière globale. De plus, les travaux menés avec l'équipe Airelle [Bétrancourt99] ont permis de montrer que l'approche basée sur les groupes et les relations était plus facilement compréhensible et offrait une édition plus souple.

De ces faits nous allons proposer un formalisme d'édition relationnel caractérisé par :

- La spécification des informations temporelles et/ou spatiales par relations, même pour des langages non relationnels.
- La spécification des médias sous forme d'intervalle de valeurs comme dans les langages (ISIS, Madeus).

Cependant, comme nous venons de le voir dans la section précédente le formalisme d'édition offert par l'environnement d'édition est là pour faciliter l'édition du langage de présentation. Pour chaque environnement d'édition réalisé, le formalisme d'édition est donc dépendant du langage de présentation sur lequel sera basé l'édition. Néanmoins, quel que soit le langage de présentation et de sauvegarde utilisé les environnements d'édition auront un certain nombre de caractéristiques communes. Nous allons donc présenter ses caractéristiques communes en trois étapes : définition des médias, définition des groupes et définition des relations, tout en spécialisant nos propos dans le cadre de langages de présentation précis lorsque cela sera nécessaire. Les langages qui nous servirons à illustrer ce chapitre seront SMIL, MHML et Madeus. Ces langages sont aujourd'hui représentatifs des approches événementielles et relatives de spécifications.

Le formalisme fourni par l'environnement auteur est le suivant :

Définition de médias :

Les médias sont caractérisés par un ensemble d'attributs :

- Le type du média.
- Les attributs spatiaux du média: X, Y, Hauteur, Largeur.
- Les attributs de présentation du média : couleur, fonte, police.
- Les attributs temporels : début, fin, durée.
- Le lien de navigation associé au média, ce lien permet au lecteur de naviguer de document en document.
- Le lien temporel associé au média, un lien temporel permet d'aller à un instant donné de la présentation : dans l'exemple de la Figure III-4, on visualise une exécution d'un document multimédia dans lequel on a spécifié le lien temporel : "Si l'auteur clique sur l'objet A, alors on saute à la 40ème seconde du document". Le comportement de ce lien est le suivant : si le lecteur clique sur l'objet A à la 20ème seconde de la présentation, cela aura pour effet d'interrompre tous les objets en cours (A, D et E) et de déclencher la présentation de tous les objets

présents à la 40ème seconde (B et C). Dans le cas de l'objet B, celui-ci sera déclenché à sa 5ème seconde du fait qu'il aurait normalement dû commencer à la 35ème seconde du document.

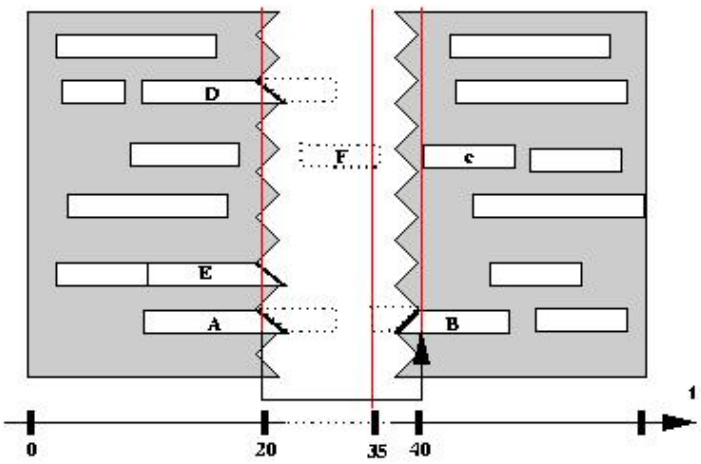


Figure III-4 : Lien temporel

On peut noter que la spécification de la valeur d'un attribut n'est pas obligatoire. De plus, chaque attribut, dont l'ensemble de valeurs est énumérable est défini par intervalle de valeurs possibles, par exemple :

- Début = [13, 15, 18], qui signifie que le début de l'objet est compris entre 13 et 18 secondes et l'auteur indique qu'il aimerait qu'il commence à la 15e seconde si cela est possible ;
- X = [100, ? ?], qui signifie que l'auteur indique que l'objet doit au moins être à 100 unités du bord.

Dans l'exemple de la Figure III-5 nous donnons un exemple de syntaxe (décrite en XML) pouvant servir pour décrire un média.

```
<Média Type="Image" >

<AttributSpatiaux
       X="8 10 12"
       Y="10 10 14"
       Hauteur="12"
       Largeur="10">
<AttributTemporels
       Begin="12 14 15"
       End="16 16 16"
       Durée="1 2 4">
<LienNavigation =http://www.inrialpes.fr/opera/>
<LienTemporel ="#14s">
</Media>
```

Figure III-5 : Exemple de syntaxe décrivant un media

Définition de la présentation :

Nous venons de voir comment définir un élément média, cet élément est la base de tout document multimédia. Nous allons maintenant nous intéresser à la définition de la présentation du document. C'est-à-dire à l'agencement des différents médias temporellement et spatialement. Cet agencement se fera dans le cadre d'une structure à base de groupes. Nous allons dans un premier temps écrire les attributs associés à un groupe avant de décrire plus précisément les groupes temporels et spatiaux.

Un groupe est caractérisé par :

- Un ensemble d'éléments qui composent ce groupe.
- Un opérateur qui permet de définir un comportement pour tous les éléments.
- Liste de relations entre les objets. La liste des relations permises est dépendante du langage de présentation/sauvegarde.
- Définition d'attributs de présentation :
  - attributs : X, Y, Hauteur, Largeur pour les groupes spatiaux.
  - attributs : début, fin, durée pour les groupes temporels.

Nous allons maintenant donner un exemple de syntaxe pouvant permettre de décrire cette spécification de la présentation (Figure III-6).

```
<GroupeTemporel
      ListeElement="A B C D E"
      Opérateur ="Inside" >
      <AttributsTemporel
            Début = "14 18 19"
            Durée = " 5 6 8"
            fin = "19 24 27" />
      <ListeRelations >
            <Relation Nom="Before" Objet1="A" Objet2="B" Param="4s">
      </ListeRelations>
</GroupeTemporel>
```

Figure III-6 : Exemple de syntaxe permettant de décrire la présentation

Le formalisme que nous venons de définir permet d'associer à chaque groupe un opérateur, un ensemble de relations et des attributs. Cet ensemble d'informations permet de définir le placement relatif de chaque objet du groupe par rapport au début du groupe. Le placement relatif d'un objet est donc le résultat d'un calcul (le formatage) entre ces différentes informations.

Ces informations peuvent être :

- *Cohérentes :*dans ce cas la solution trouvée par le formateur satisfera toutes les informations données par l'auteur. Dans l'exemple de la Figure III-7, les différentes informations spécifiées sont complémentaires et permettront au formateur de calculer une solution pour le placement temporel des objets.

```
<GroupeTemporel Id="C"
      ListeElement="A B"
      Opérateur="Inside"/>
      <AttributTemporel dur="10 10 10" />
      <ListeRelations >
            <Relation Nom="Before" Objet1="A" Objet2="B">
      </ListeRelations>
</GroupeTemporel>
```

Figure III-7 : Exemple d'informations complémentaires

- *Incohérentes*: dans ce cas le module d'aide du système doit vérifier la cohérence de la spécification en fonction de la sémantique du langage de présentation. Dans l'exemple de la Figure III-8, les durées spécifiées sur l'objet Composite C et l'élément A sont incompatibles avec l'opérateur *Inside*. Dans ce cas, le système essayera de satisfaire la relation, et indiquera à l'auteur qu'il y a une incohérence avec les valeurs des attributs. On peut noter, que si l'auteur avait spécifié un intervalle de valeurs plus large pour l'objet composite, le système aurait de lui-même ajusté cet intervalle de manière à satisfaire l'ensemble de la spécification. Les différentes incohérences possibles seront présentées dans la section 2.4.

```
<GroupeTemporel Id="C" ListeElement="A " Opérateur="Inside"/>
      <AttributTemporel dur="9 9 9" />
      <Element Id="A" />
            <AttributTemporel dur="10 10 10"/>
      </Element>
</GroupeTemporel>
```

Figure III-8 : Exemple d'informations contradictoires

En introduction de cette section nous avons dit que le jeu de relations et d'opérateurs offert par l'environnement idéal est lié au langage de présentation utilisé. Nous allons donner ici le jeu de relations et d'opérateurs défini pour les trois langage SMIL, Madeus et MHML.

Ces relations seront :

- soit une simplification ou une spécialisation des relations proposées par le langage de sauvegarde, et dans ce cas elles seront sauvegardées directement dans le langage de présentation
- soit un raccourci pour manipuler un ensemble de relations temporelles de plus bas niveau. Le mécanisme d'espace de noms offert par XML sera utilisé pour sauvegarder ces informations qui sont propres à l'environnement d'édition.

Ces relations seront définies par le concepteur de l'environnement auteur, mais elles doivent être aussi extensibles par l'auteur s'il en éprouve le besoin. Nous présenterons ce mécanisme dans le chapitre IV section 7.

Nous allons présenter maintenant, le formalisme pour les trois langages qui nous servirons à illustrer ce principe : Madeus, SMIL et MHML.

Dans le cas de langages événementiels, l'environnement fournit à l'auteur la possibilité d'éditer directement les événements du langage par l'intermédiaire d'une palette par exemple.

Le formalisme d'édition pour le langage Madeus :

Le jeu de relations de Madeus étant relativement complet spatialement, nous l'étendons seulement du coté temporel.

Pour cela nous intégrons quatre nouvelles relations aux relations définies dans Madeus-97 (chapitre II section 3.4.2) :

- Le *begin* et le *end*, qui permettent de sortir du cadre des relations d'Allen de base tout en restant dans les relations pointisables (elles peuvent s'exprimer en relation d'instants). Ces relations lèvent la contrainte sur les relations *starts* et *finishes* qui disait que l'objet A devait être plus petit que l'objet B.
- Le *center* et le *sameDuration* : qui nous font sortir des relations pointisables. Ce type de relation impliquera des contraintes sur les mécanismes de formatage que nous utiliserons (chapitre V). La relation *sameDuration* indique que deux objets ont la même durée, sans imposer que ces deux objets commencent en même temps, c'est une généralisation de la relation *equal*. La relation *center*, elle, est une

spécialisation de la relation *during*.

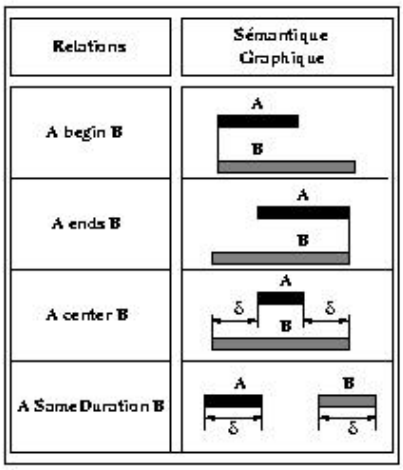


Figure III-9 : L'extension des relations temporelles de Madeus

Le formalisme d'édition pour le langage SMIL :

Le langage SMIL contient essentiellement des opérateurs temporels. Le langage doit donc être essentiellement étendu spatialement. En effet, le langage SMIL permet uniquement de définir un positionnement spatial absolu, il est donc nécessaire pour faciliter l'édition de l'auteur d'offrir une édition de plus haut niveau. Pour cela nous définissons dans le formalisme de l'environnement auteur associé à SMIL les relations spatiales suivantes *: LeftAlign, RightAlign, Center, TopAlign, BottomAlign*. Chacune de ces relations pouvant prendre un paramètre permettant de définir un décalage.

Nous étendons aussi le formalisme d'un point de vue temporel, en insérant l'opérateur *ParEquals* permettant de définir un élément composite de type parallèle avec comme contrainte supplémentaire que tous les fils ont la même durée (voir Figure III-10).

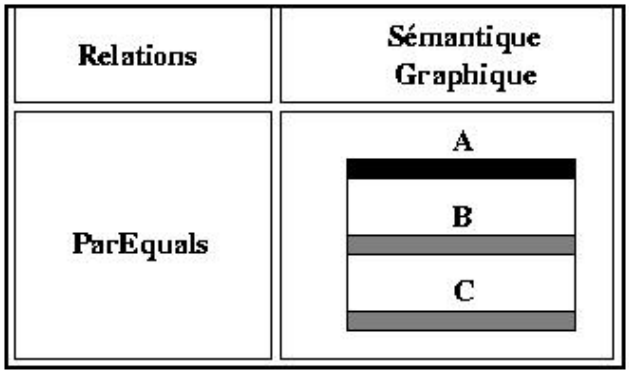


Figure III-10 : L'extension de relation temporelle de SMIL

Le *ParEquals* permet de préciser une sémantique particulière à l'opérateur *Par*, celle-ci sera que tous les objets contenus dans le *Par* auront la même durée.

De plus, dans le cas d'un opérateur *Par*, nous permettons à l'auteur de définir des relations entre les éléments du *Par*. Ces relations sont les relations de Madeus-97 étendues des quatre relations décrites dans la Figure III-9. Cette spécification relative facilitera la tâche lors des phases d'édition en permettant à l'auteur de spécifier le comportement de son document de manière relative.

Le formalisme d'édition pour le langage MHML :

Le langage MHML est de plus bas niveau que les deux langages présentés ci-dessus. Il est donc nécessaire pour faciliter la tâche de l'auteur de définir un jeu de relations de haut niveau plus important que pour les deux langages précédents. Pour cela nous définissons pour les relations temporelles l'ensemble des relations de Madeus-97 (Allen + relations causales) que nous étendons comme pour Madeus des quatre relations décrites dans la Figure III-9.

Nous définissons comme jeu de relations spatiales, les relations spatiales suivantes *: LeftAlign, RightAlign, Center, TopAlign, BottomAlign*. Chacune de ces relations pouvant prendre un paramètre permettant de définir un décalage.

Ce formalisme permet de répondre aux besoins spécifiés dans le chapitre précédent. Les différents besoins en expressivité sont satisfaits (dans le cadre de documents prédictifs avec de la navigation). De plus les besoins de structuration et de support pour la modification du document sont satisfaits grâce d'une part à la structure de groupe et d'autre part à l'utilisation de relations flexibles et la définition des attributs par intervalle.

2.2.3 Les services d'édition de l'environnement idéal

Un système auteur doit fournir un ensemble de fonctions d'édition et de services pour permettre à l'auteur d'éditer son document au travers du formalisme d'édition proposé. Ce formalisme est une extension du formalisme du fichier de sauvegarde.

Dans le chapitre précédent, nous avons listé les différents besoins de l'auteur pendant ce processus, nous allons ici lister les services que le système doit offrir pour répondre en partie à ces besoins. Ces fonctions sont :

- Les opérations de base :
  - l'insertion de nouveaux médias dans le document ;
  - la modification des attributs des médias ;
  - l'ajout de relations spatiales et/ou temporelles.
- Les fonctions plus évoluées : comme le copier/coller, qui peut être un simple copier/coller de médias, mais aussi un copier/coller plus évolué qui permet de dupliquer des groupes d'objets ou les relations entre plusieurs objets.

On peut remarquer que ce sont des fonctionnalités que l'on retrouve dans l'édition de documents structurés [Quint99]. On peut noter, que la réalisation d'une fonction telle que le copier/coller évolué pose de nombreux problèmes. En effet, le copier/coller ne se résume pas seulement à une duplication de la structure et des informations de présentation, mais elle nécessite souvent une transformation de ces informations en fonction du contexte dans lequel on l'insert. Cette transformation ne peut être complètement automatique et doit être assistée de l'auteur ([Bonhomme98]). Dans notre contexte cette difficulté est liée à la notion de groupe ainsi qu'aux relations.

2.2.4 Edition par manipulation directe

Une fonctionnalité offerte par l'environnement auteur idéal doit être l'édition pat manipulation directe. C'est-à-dire que l'auteur doit pouvoir éditer les attributs de ces objets dans les vues spatiale ou temporelle.

De plus, le système doit maintenir les relations existantes entre les objets durant tout le long du cycle d'édition, y compris lorsque l'auteur déplace graphiquement les objets :

- dans la vue de présentation pour les relations spatiales ;
- dans la vue temporelle pour les relations temporelles.

Dans la Figure III-11 l'auteur déplace la bicyclette, et ce dernier reste sur la droite qui modélise une pente. Au cours de ce déplacement le logiciel Cabri [Laborde95] maintiendra les différentes relations que l'auteur a spécifiées.

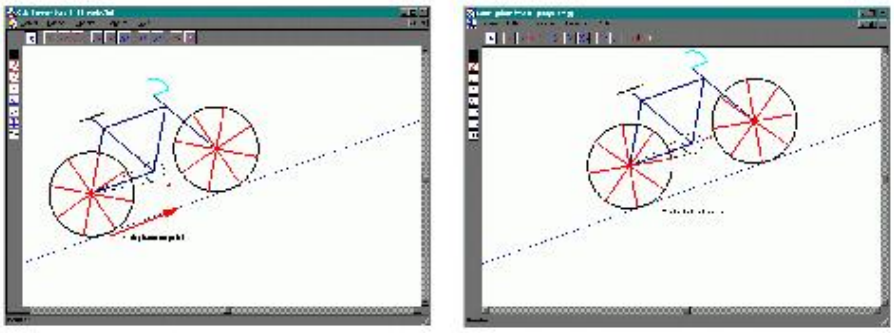


Figure III-11 : Manipulation directe dans Cabri Géomètre II

On peut noter cependant que, dans cette représentation, l'auteur ne visualise pas les relations entre les différents éléments, il en a seulement une perception par les déplacements autorisés. De plus, lors du déplacement, il n'a aucune explication sur le comportement de l'objet, notamment sur les relations qui sont maintenues.

2.3 Spécification d'un système multivues

Nous venons de voir que nous avons choisi d'utiliser un formalisme essentiellement relationnel pour favoriser la tâche d'édition de l'auteur. Cependant, dans le chapitre II section 3.4, nous avons vu que les contreparties à cette flexibilité sont une plus grande complexité de la tâche de visualisation et la nécessité d'automatiser les services de cohérence et de formatage. Nous présenterons les différentes techniques de cohérence et de formatage dans le chapitre V.
Nous allons nous intéresser maintenant aux services de visualisation que doit fournir l'environnement auteur idéal. Ces services de visualisation ne doivent pas être indépendants de l'édition et doivent être liés au processus et aux fonctions d'édition. Ces services de visualisation se feront au travers de *vues*.
Une *vue* est non seulement une entité graphique qui permet de visualiser à l'écran une information, mais c'est une entité qui permet à l'auteur de l'éditer.

2.3.1 Spécification des différentes vues

Étant donnée la complexité des documents multimédias, offrir une seule vue à l'auteur serait insuffisant. Le système auteur doit donc offrir plusieurs vues adaptées à la visualisation des informations attendues par l'auteur. Cependant le système auteur doit aussi fournir des vues pour aider l'auteur, notamment lors de la navigation dans son document. Les vues nécessaires sont :

- La vue de présentation qui permet de présenter le document. Cette vue doit permettre aussi de visualiser les informations spatiales du document. C'est-à-dire que le système offre un mécanisme de visualisation des différentes informations liées aux relations ou aux groupes définis par l'auteur. D'un point de vue édition cette vue doit permettre à l'auteur d'éditer les attributs spatiaux par des manipulations directes.
- La vue temporelle qui présente la dimension temporelle du document et qui permet à l'auteur de naviguer dans l'espace de solutions défini par son document. Cette vue doit aussi offrir des mécanismes permettant de visualiser les relations ainsi que les informations liées à la hiérarchie temporelle. D'un point de vue édition, cette vue doit permettre la manipulation des différents attributs temporels.

- La vue hiérarchique qui visualise les structures temporelle et spatiale du document. Cette vue doit aussi servir de support pour l'édition en facilitant les opérations de restructuration que désire faire l'auteur. Cette vue doit aussi permettre à l'auteur de visualiser les opérateurs associés à chacun des groupes d'objets.
- La vue hypertexte qui permet à l'auteur de visualiser la structure de navigation de son document. Cette vue doit permettre non seulement de voir l'ensemble des liens contenus dans le document, mais elle doit aussi permettre à l'auteur d'avoir une vue plus globale de l'interdépendance entre plusieurs documents.
- La vue objet, qui permet à l'auteur de visualiser l'ensemble des objets présents dans son document.
- La vue attributs, qui permet de visualiser les différents attributs d'un objet du document.
- La vue résumé, qui visualise les instants clés du document. Ces instants clés peuvent être définis par l'auteur ou calculés automatiquement par l'environnement auteur selon plusieurs paramètres (par exemple en ne prenant en compte que les instants de début ou de fin des objets).
- La vue rapport d'exécution, qui permet de visualiser le résultat d'une exécution.
- La vue source, qui permet de visualiser la source du fichier de sauvegarde. Cette vue textuelle peut être une vue qui offre des services de synchronisation évolués (par exemple sur la sélection) entre les éléments de cette vue et les autres vues. L'outil Smartool développé dans l'équipe Oasis de l'INRIA-Sophia Antipolis, offre de telles vues dans le cadre de l'édition de langages dédiés à la programmation.

Nous verrons dans les sections 3 et 4 comment ces vues permettent de répondre aux différents besoins de visualisation et de navigation.

2.3.2 Coopération entre les différentes vues du document

Une des difficultés pour l'auteur vient du nombre de vues nécessaires à la visualisation des différents éléments contenus dans le document. Pour faciliter le travail de l'auteur dans les différentes vues, on peut imaginer une synchronisation sur l'objet sélectionné qui impose au système, lorsque l'auteur change de vue, de focaliser la vue sur cet objet. Ainsi, si l'auteur passe de la vue temporelle à la vue de présentation, cette dernière se positionne à l'instant de début de l'objet sélectionné.

Un autre moyen de faciliter la tâche de l'auteur est de visualiser plusieurs informations dans une vue. Par exemple, la vue temporelle peut non seulement visualiser l'enchaînement temporel des objets mais aussi la structure hiérarchique. La visualisation de cette structure peut permettre à l'auteur de faire plus facilement le lien entre la vue hiérarchique et la vue temporelle. De même, au cours de l'exécution du document dans la vue de présentation, le système peut afficher la progression de la présentation du document. Ces synchronisations sont essentielles pour que l'auteur puisse se repérer facilement entre les différentes vues.

Le fait d'avoir différentes vues permet aussi d'aider l'auteur dans les tâches d'édition complexes, en découpant ces tâches en sous-tâches, chacune s'accomplissant dans la vue la plus adaptée. Nous verrons cela plus précisément dans l'exemple de la section 5.

Aujourd'hui de nombreux systèmes proposent plusieurs vues pour faciliter les tâches de l'auteur. Amaya [Amaya00], par exemple, permet d'utiliser la vue *structure* pour parcourir les pages HTML, ce qui permet à l'auteur d'avoir une vue globale de son document et de naviguer à l'intérieur.

La spécification proposée ici vise à généraliser et à adapter ces principes au contexte multimédia.

Par exemple, pour permettre à l'auteur de faire facilement le lien entre une présentation de son document et la spécification (spécification qui permet de définir un ensemble de présentations), nous offrons en cours d'exécution une ligne temporelle qui se déplace dans la vue temporelle. Cette ligne permet de visualiser l'instant courant de la présentation. Elle permet à l'auteur de faire le lien entre la présentation et le document. Ce mécanisme se retrouve dans Grins et Director.

Notons que l'utilisation d'un système multi-vues nécessite une synchronisation des informations visualisées dans chacune des vues de manière à garantir une certaine cohérence à l'auteur.

2.4 Les services d'aide

Comme nous avons pu le voir précédemment, l'édition de l'auteur peut se faire à deux niveaux : au niveau du formalisme de l'environnement auteur ou alors au niveau du formalisme du langage de sauvegarde utilisé. De ce fait, les services d'aide sont, eux aussi, à deux niveaux.

Le système auteur doit assurer des services complets de vérification de la cohérence, d'aide lors de la spécification (par exemple en l'aidant dans le calcul des durées des objets), mais aussi dans les cas d'erreur ou d'incohérence de documents, en offrant une visualisation et un diagnostique des erreurs sur les deux niveaux possibles de la spécification.

Nous allons maintenant nous intéresser aux différents services d'aide fournis par l'environnement auteur.

La vérification de cohérence : à chaque modification du document, le système d'édition doit vérifier la cohérence du document et calculer une nouvelle solution. En effet, la complexité des documents multimédias est telle que l'auteur ne peut pas appréhender toutes les répercussions possibles d'une modification qu'il a effectuée. Le système doit donc être capable de vérifier en temps réel la cohérence d'un document. Quand une incohérence est détectée le système d'édition doit informer l'auteur de l'erreur, et lui donner un diagnostique aussi complet que possible sur l'erreur de manière à ce que celui-ci puisse rendre le document de nouveau cohérent [Song99]. Il existe différents types de cohérence possibles [Layaïda97, Sabry99] :

- *Cohérence qualitative :* une incohérence qualitative est détectée dans les cas suivant :
  - Si l'auteur met des relations incompatibles entre elles, quelle que soit la valeur des différents attributs sur les objets. Dans la Figure III-12a, on peut voir un exemple où l'auteur a mis 3 relations : A meets B, B meets C, et il essaye de mettre une relation C meets A. Cette dernière relation est incohérente avec les deux autres. Ce type d'incohérence est aussi valable pour les relations spatiales.
  - Si l'auteur définit une relation avec un objet qui n'existe pas.

- Si l'auteur définit des liens temporels incohérents, comme, par exemple, un lien qui pointerait vers la 30ème seconde d'un document qui ne durerait que 25 secondes, ainsi que la détection des objets qui ne seront jamais joués. Par exemple, un objet commençant à la 30ème seconde d'un objet composite qui ne durerait que 20 secondes.

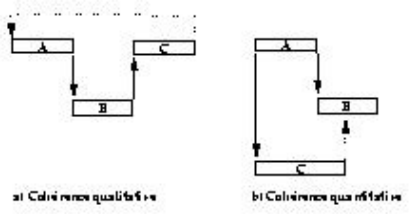


Figure III-12 : Cohérence qualitative et quantitative

- *Cohérence quantitative* **:**une incohérence quantitative arrive lorsque le document est qualitativement cohérent, mais qu'une valeur sur un attribut fait que le document est incohérent. Par exemple, dans Figure III-12b, l'auteur a spécifié que A meets B, A starts C et C finishes A, ce qui est qualitativement cohérent. Mais si comme dans l'exemple A a une durée de 5 unités de temps, B a une durée de 4 et C une durée de 3, les objets A et C ne peuvent terminer en même temps comme l'indiquent les relations. Le document est donc quantitativement incohérent.
- *Cohérence causale*: une incohérence causale est détectée quand un objet doit interrompre un objet qui est déjà fini. Elle est liée à l'insertion de relations telles que : *parmin, parmax, parmaster*. Cette cohérence se ramène dans le cas de documents déterministes à la vérification de la cohérence quantitative [Layaïda97].
- *Cohérence événementielle*: il s'agit de vérifier que les actions associées à l'événement sont réalisables. C'est-à-dire que les objets impliqués dans les actions seront *actifs* (ils seront en cours de présentation au moment de l'activation du lien). Dans le cas particulier des documents prédictifs, on se ramène à la vérification de cohérences quantitative et qualitative. Il faut par exemple vérifier que les objets qui sont la destination des événements soient présents au moment où l'objet source déclanchera l'événement.
- *Cohérence hyper-temporelle***:**la cohérence hyper-temporelle est liée aux liens :
  - Dans le cas de liens globaux, cela ne pose pas de problèmes de cohérence car la navigation hypertexte engendrée par ces liens est orthogonale à l'aspect temporel du document et donc ne crée pas d'incohérence.
  - Dans le cas de liens locaux, il faut vérifier que le lien pointe sur un objet actif. Dans le cadre de documents prédictifs on retombe dans le cas de la cohérence événementielle [Sabry99].

Au cours du processus d'édition, les erreurs sont détectées directement lors de la spécification. Les mécanismes pour détecter ce genre d'erreurs sont relativement simples à mettre en oeuvre. Les principaux algorithmes sont PC2 [Mackworth85], DPC [Meiri92] et DPCincrémental [Chleq95]. Le problème majeur de ces approches est le temps de vérification nécessaire.

Le formatage :

L'environnement auteur idéal doit posséder un formateur. En effet, un besoin essentiel de la phase d'édition est le formatage. La valeur d'un attribut lors de la présentation est le résultat d'un calcul entre les différentes spécification de l'auteur (attribut, relation,). Comme nous avons pu le voir dans les différents langages présentés, il existe différentes méthodes pour spécifier la valeur d'un attribut. La valeur d'un attribut au moment de la présentation peut dépendre de :

- Relations : par exemple si l'on a une relation *alignée à gauche* entre deux objets.
- Intervalle dans lequel il est défini : par exemple si l'attribut X d'un objet est contraint entre les valeurs 50 et100.
- Valeur spécifiée par l'auteur : par exemple si l'auteur a indiqué que l'objet doit se situer si possible en 80.
- Valeur héritée de la structure hiérarchique : par exemple, l'objet composite a l'attribut X qui a pour valeur 60, et tous les fils de cet objet ont un attribut X qui est plus grand que celui de leur ancêtre.

Prenons l'exemple d'un document où l'on possède les informations suivantes :

- Un composite C contenant A et B des objets de type texte (tous les objets contenus dans C sont dans la boîte englobante définie par les attributs X, Y, Hauteur, Largeur de l'objet C).
- L'attribut X de C vaut 30.
- L'attribut X de l'objet A est dans [50..160].
- L'attribut X de l'objet B est dans [130..360].
- L'attribut X de B vaut actuellement 190
- L'attribut X de A vaut actuellement 50

L'auteur ajoute ensuite une relation de la forme :

- A.X = B.X

Le système doit alors calculer une nouvelle valeur pour les attributs X des objets A et B.

Dans le cas ci-dessus, le système doit affecter par exemple la valeur 130 aux deux attributs. Cette valeur satisfait l'ensemble des informations que l'auteur a spécifiées. Sans ce nouveau calcul, le document aurait été quantitativement incohérent.

On peut noter que, lors d'une opération d'édition (la modification de la valeur d'un attribut ou de l'ajout d'une relation), le formateur doit calculer un nouveau placement (spatial ou temporel) pour les objets. Ce calcul doit être rapide lors du processus d'édition.

L'utilisation de méthodes ad hoc de formatage ([Quint98], [Tardif97b]) est aujourd'hui la plus répandue dans les systèmes auteur, cependant l'amélioration des techniques à base de contraintes semble offrir des perspectives intéressantes en termes de simplicité d'expression, d'évolution et de maintenance. Nous étudierons dans le chapitre V ces techniques et nous verrons comment les utiliser dans notre contexte.

Aide au diagnostique : le système doit apporter une aide à l'auteur pour expliquer ces actions dans deux situations :

- Explication lors de la détection d'une incohérence, il doit aider l'auteur dans la compréhension de celle-ci. Cela peut aller de la visualisation des relations impliquées dans l'erreur à la visualisation de toutes les relations induites sur les objets des relations impliquées dans l'erreur.
- Explication du formatage calculé par le système. En effet, le système calculant automatiquement le placement et la durée des objets, ce calcul peut être déroutant pour l'auteur dans certains cas. Il est donc important que le système puisse donner à l'auteur des explications concernant le choix des valeurs pour les différentes variables.

2.5 Synthèse sur la spécification de l'environnement idéal

Nous venons de définir un environnement idéal fournissant un formalisme d'édition, un ensemble de vues pour visualiser et éditer son document au travers du formalisme d'édition offert, et un ensemble de services d'aide tels que la vérification de cohérence, le formatage et le diagnostique pour faciliter la tâche de l'auteur.

Aujourd'hui aucun système auteur n'apporte de réponses pertinentes à ces différents besoins. Au cours des chapitres suivants nous apporterons des réponses concrètes à ces besoins.

L'ensemble des services décrits ci-dessus est complexe du fait qu'il doit à la fois permettre à l'auteur de manipuler facilement le document, tout en essayant de garder un lien avec le langage de présentation. Ce lien doit être conservé pour permettre de satisfaire les besoins liés à la vie du document ainsi qu'au cycle d'édition.

Nous allons maintenant valider ces propositions en indiquant comment elles permettent de répondre aux différents besoins de visualisation (section 3) définis dans le chapitre II, ainsi qu'aux différents besoins de navigation de l'auteur (section 4).

3. Visualisation d'information dans un contexte d'édition

Le système d'édition doit offrir la possibilité à l'auteur d'explorer les informations du document ([Jourdan97b, Jourdan98d]), tout en lui permettant d'adapter l'exploration de l'ensemble de ces informations en fonction de ses besoins (édition, compréhension du comportement de son document,).

Nous allons voir comment l'environnement idéal que nous venons de spécifier permet de répondre aux différents besoins définis dans le chapitre II liés à la fois à la visualisation des médias de base qui constituent le document (section 3.1), mais aussi à la visualisation statique des différentes dimensions qui le composent (section 3.2). Dans la section 4 nous verrons comment les différentes manipulations offertes dans les vues permettront de satisfaire le besoin de perception de l'espace de solutions.

Lors de ces présentations nous essaierons de voir pour chaque besoin quelles techniques permettent de le satisfaire. L'objectif de cette section est donc double :

- vérifier que l'environnement idéal que nous avons spécifié répond aux besoins de visualisation et de navigation définis dans le chapitre II section 2.2.2. Au cours de cette section nous serons amenés à préciser les besoins de visualisation en fonction de chacune des informations traitées.
- étudier quelles sont les solutions techniques existantes pour satisfaire ces besoins. Ceci afin d'évaluer les difficultés liées à la réalisation future de cet environnement idéal.

3.1 Accès aux médias et aux informations de base

Lors des manipulations de médias dans la phase d'édition l'auteur a besoin de visualiser :

- les attributs des médias ;
- la liste des médias manipulables dans son document ;
- la structure des médias complexes (vidéos structurées par exemple).

La visualisation des attributs des médias peut se faire via une palette d'attributs. Cependant, l'auteur a besoin de percevoir non seulement l'aspect quantitatif des attributs mais aussi un aspect qualitatif. De ce fait, visualiser ces informations sous une autre forme est essentiel. Par exemple, les attributs spatiaux seront visualisés dans la vue de présentation.

La visualisation de l'ensemble des éléments d'un document multimédia pose quant à elle des difficultés à cause du facteur d'échelle. L'ordre de grandeur d'un document multimédia est de mille objets pour une présentation d'une dizaine de minutes. L'idée la plus simple consiste à visualiser tous les médias dans une fenêtre comme dans Director en y ajoutant une visualisation des attributs de l'objet sélectionné (Figure III-13). Cette représentation très utile pour de petits documents se révèle inexploitable pour de gros documents. Des techniques à base de filtres peuvent être utilisées pour ne visualiser que les objets d'un certain type ou qui apparaissent dans une certaine scène ou encore les objets non encore utilisés. Ce sont des techniques issues de la visualisation d'informations qui pourraient être utilisées (visualisations en oeil de poisson [Sarkar94] ou sur les murs de perspective [Mackinlay91]). Malheureusement ces techniques ne sont encore que peu employées dans le cadre de l'édition de documents multimédias.

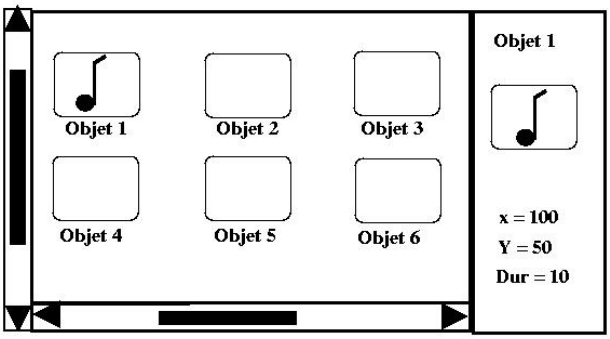


Figure III-13 : Vue des objets présents dans le document

On peut noter que pour des médias comme la vidéo, il est nécessaire de pouvoir visualiser de nombreuses informations. Outre les attributs classiques (spatiaux et temporels), il peut être utile d'avoir des informations sur le nombre de plans, de scènes ainsi que les dépendances temporelles ou spatiales de ces différents éléments.

Dans le domaine de l'édition et l'annotation de vidéo, on peut trouver de nombreux travaux qui permettent d'éditer des parties de vidéos et/ou de les annoter.

Le point commun de tous ces travaux est qu'ils manipulent une structure sur les médias. Parmi ces travaux, on peut citer les travaux de Dumas [Dumas00] qui permettent de définir un modèle d'annotation pour les vidéos, ceux de Chang [Chang97] qui permettent de rechercher de l'information dans des vidéos annotées et enfin ceux de Carrer [Carrer97] qui ont vu la réalisation d'un outil d'annotation. Dans le cadre du projet Opéra les travaux de Tien [Roisin00] visent à intégrer un outil de structuration et d'édition de vidéo au processus d'édition de documents multimédias, permettant ainsi à l'auteur de mettre des relations entre des objets du document et des scènes de la vidéo par exemple. Ces différents travaux n'offrent que peu de représentations graphiques novatrices pour visualiser ces informations et se ramènent souvent à la visualisation de la structure hiérarchique de la vidéo. Cette structure étant synchronisée avec une représentation graphique de la vidéo permettant ainsi à l'auteur de faire facilement le lien entre la structure ou l'annotation et l'image (ou partie de la vidéo) associée.

Dans le cadre de l'environnement idéal que nous avons proposé ces besoins sont couverts par les vues objets, attributs et hiérarchique.

3.2 Accès aux différentes dimensions du document

Dans un premier temps, nous allons nous intéresser à l'aspect visualisation d'informations sans parler des aspects de navigation. Pour chacune des dimensions, nous nous intéresserons dans un premier temps à l'étude de différents travaux permettant de visualiser le type d'informations correspondant sans nous limiter au contexte des documents multimédias, et dans un deuxième temps nous regarderons comment ces techniques sont employées dans notre contexte.

Nous présenterons différents travaux selon le type d'informations visualisé :

- la visualisation d'informations temporelles (section 3.2.1) ;
- la visualisation d'informations spatiales (section 3.2.2) ;
- la visualisation d'informations hiérarchiques (section 3.2.3) ;
- la visualisation d'informations hypertextuelles (section 3.2.4) ;

Nous nous intéresserons aussi à la visualisation des propriétés intrinsèques à notre formalisme d'édition :

- Les informations dynamiques (section 3.2.5) ;
- Les informations sur les relations (section 3.2.6).

3.2.1 Visualisation de l'information temporelle

Un des principaux besoins dans le cadre de l'édition de documents multimédias consiste à visualiser les informations temporelles contenues dans le document.

Ces informations sont de différentes natures :

- attributs temporels ;
- relations et groupes temporels ;
- enchaînement temporel des médias contenus dans le document.

Une solution utilisée classiquement pour représenter de telles informations est une vue temporelle linéaire communément appelée *vue temporelle*. Une telle visualisation est utilisée par exemple dans le cadre d'outils d'édition de vidéo (Adobe première [Adobe-Premiere00]) ou de musique et même dans certains éditeurs de documents multimédias ([Macromédia-Director00], [Jourdan97a]).

Dans le cadre de l'édition de documents à l'aide de relations, une des difficultés est liée à ce que l'auteur manipule un ensemble d'exécutions possibles de son document. Par une représentation statique, il est difficile de faire percevoir à l'auteur l'espace de solutions tout en préservant une réalité temporelle à la représentation. Des métaphores telles que celle des ressorts [Tardif97] peuvent être employées pour indiquer à l'auteur les zones du document qui contiennent de la flexibilité. Dans l'exemple de la Figure III-14 on peut voir un exemple d'une telle représentation.

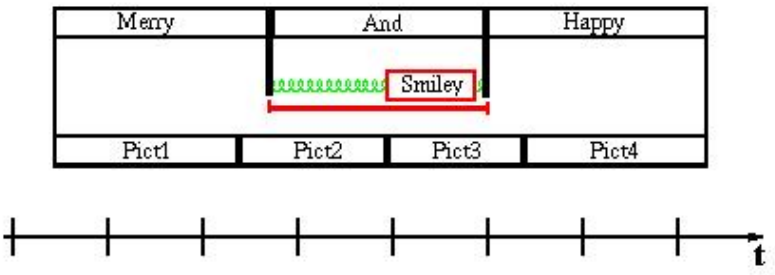


Figure III-14 : Visualisation de la flexibilité dans la vue temporelle

Le principal problème d'une telle représentation est la visualisation de gros volumes d'informations.

On peut noter que dans l'environnement idéal certaines informations sont représentées dans plusieurs vues. Par exemple, les attributs temporels sont visualisés dans les vues attribut et temporelle ce qui permet à l'auteur d'utiliser la vue la plus adaptée à son besoin. Par exemple, en phase de création d'un média la vue attribut lui permettra de positionner tous les attributs sur son média, la vue temporelle, lui permettant elle, d'ajuster le placement temporel entre les médias. De la même manière la structure de groupe temporel est visualisée dans les vues hiérarchique et temporelle.

Les problématiques de visualisation de relations étant communes à toutes les informations, nous les présenterons ultérieurement à la fin de cette section dans le cas de la visualisation des relations. Les techniques d'affichage de gros volumes d'information seront elles présentées dans la section visualisation de données hiérarchiques.

Dans la section 4 nous verrons comment des techniques de navigation peuvent améliorer la perception de cet espace de solutions.

3.2.2 Visualisation d'informations spatiales

Les informations spatiales contenues dans le document sont de différentes natures :

- attributs spatiaux ;
- relations et groupes spatiaux ;
- placement spatial des médias dans le temps.

Les méthodes de visualisation du placement spatial offertes dans l'environnement idéal pour les objets respectent le principe du WYSIWYG utilisé dans le cadre des éditeurs graphiques ainsi que pour l'édition de documents classiques. L'environnement auteur permet, par exemple, à l'auteur d'afficher certains instants précis du document dans la vue de présentation. Dans la Figure III-15 nous pouvons voir que l'auteur a stoppé la présentation de son document à un instant précis.

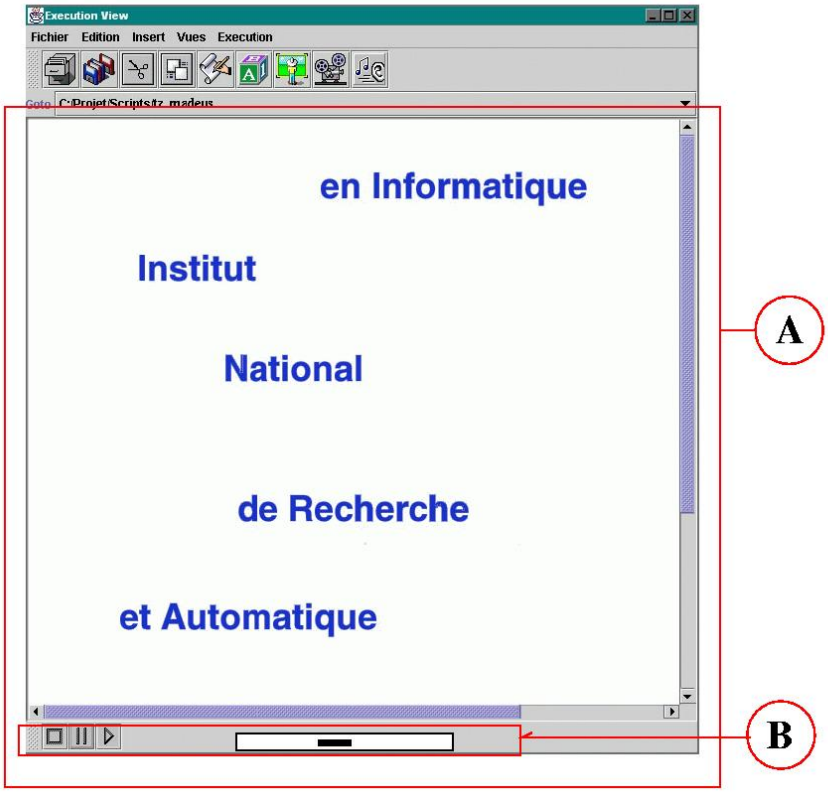


Figure III-15 : Vue de présentation

La limite de cette approche est que l'on ne peut pas voir comment deux objets qui n'ont pas d'instant de présentation en commun sont placés l'un par rapport à l'autre. Il faut donc pouvoir présenter plusieurs instants en même temps.

Dans Grins (Figure III-16), une vue permet de visualiser en même temps toutes les régions du document. On a ainsi sur la même vue, la

représentation spatiale d'objets ou de zones qui ne sont pas présents temporellement aux mêmes instants.

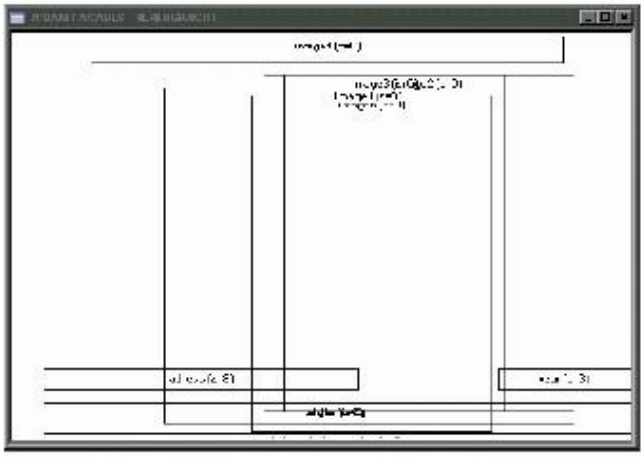


Figure III-16 : Visualisation des régions dans Grins

Ce principe peut être affiné de manière à pouvoir par exemple visualiser toutes les régions utilisées sur une période de temps, ou toutes les régions d'un sous-ensemble d'objets du document. Supposons que nous ayons un document composé de trois objets, chacun associé à une région (Figure III-17), si nous visualisons simultanément les 3 régions à l'écran nous obtenons la visualisation de toutes les régions du document. Cet affichage, fort utile pour des présentations de type transparent (pour permettre le placement d'objets comme les titres, aux mêmes endroits) se révèle vite inutilisable pour d'autres types de documents du fait du nombre de régions présentes dans le document. Pour remédier à cela des techniques de filtrage peuvent être appliquées lors de la visualisation pour par exemple ne visualiser que les régions utilisées pendant les 30 premières secondes, Il n'est malgré tout pas toujours facile de trouver un bon filtre, celui-ci étant dépendant de l'opération qu'est en train de faire l'auteur.

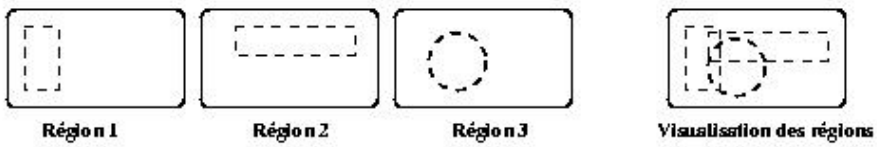
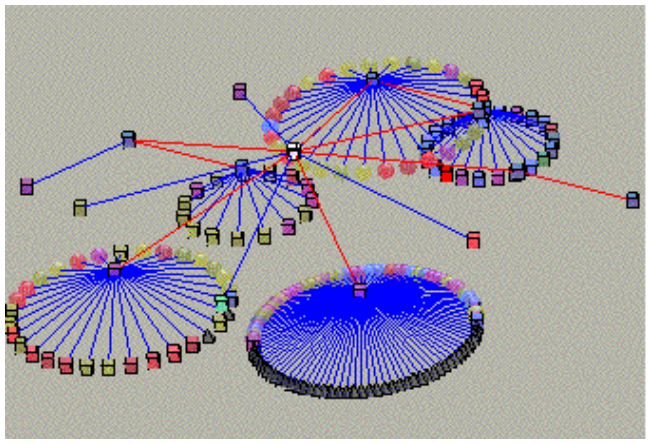


Figure III-17 : Visualisation simultanée des régions

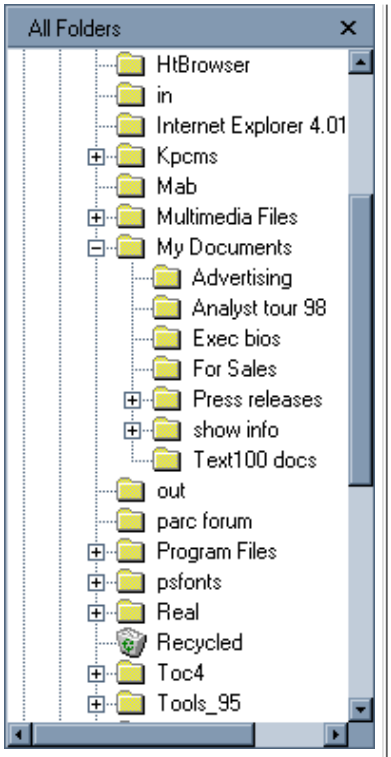
3.2.3 Visualisation d'informations hiérarchiques

La visualisation d'informations hiérarchiques est utile pour visualiser les structures hiérarchiques (spatiale, temporelle) du document.

Le problème que l'on rencontre communément dans ce contexte est celui du facteur d'échelle. Des techniques telles que l'utilisation de filtres (filtres sur le type des noeuds, sur la profondeur, sur la valeur de certains attributs) peuvent permettre d'alléger l'affichage. De plus, des techniques graphiques peuvent permettre d'améliorer la lisibilité de l'affichage. De telles techniques ont été développées essentiellement dans le cadre de la visualisation de systèmes de fichiers. Les techniques les plus connues sont à base d'arbres (Figure III-18a) ou d'arbre coniques (Figure III-18b, [Robertson91]). Dans le cadre de l'édition de documents, ce sont les techniques à base d'arbres qui sont le plus communément utilisées. On peut par exemple citer Amaya [Amaya00] qui utilise une telle représentation pour visualiser la structure XML du document (Figure III-19).



b) Représentation à base d'arbres coniques



a) Représentation à base d'arbres

Figure III-18 : Représentations hiérarchiques d'un système de fichier

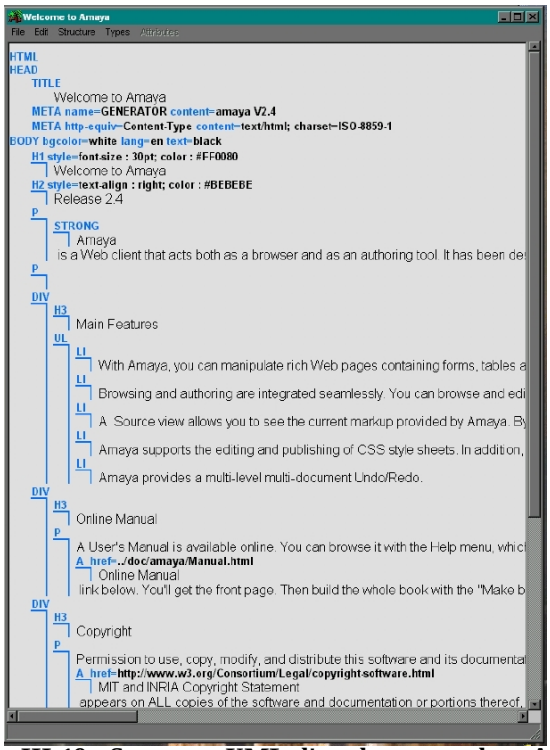


Figure III-19 : Structure XML d'un document dans Amaya

3.2.4 Visualisation d'informations sur la structure hypertexte

La dernière composante à laquelle nous allons nous intéresser est la visualisation de la structure hypertexte.

Les besoins dans le cadre de l'édition de documents multimédias sont similaires à ceux liés à l'édition hypertexte :

- visualisation des liens de navigations ;
- visualisation des liens locaux ;
- visualisation de l'interdépendance entre deux ou plusieurs documents ;
- visualisation de l'espace de navigation du lecteur.

Avec l'avènement du Web, l'explosion des pages HTML et la complexité de plus en plus croissante des sites Web, de nombreux travaux ont essayé de faciliter la perception globale des sites.

On peut noter que les besoins dans le cadre de documents multimédias recouvrent ceux qui sont liés à la dimension hypertexte mais la dimension temporelle soulève un nouveau problème. Il est important de visualiser la partie de document qui n'a pas été jouée à cause de l'activation du lien. Dans le cas de documents prédictifs, on a la connaissance de cette partie, et il est important de permettre à l'auteur de la visualiser. Ce besoin que l'on avait déjà dans la navigation hypertexte classique n'était pas satisfaisable dans ce contexte du fait que l'on n'avait pas la connaissance réelle de ce qu'avait vu (ou lu) le lecteur.

Dans la Figure III-20 on peut visualiser un exemple d'affichage de la structure d'un document hypertexte. Comme le montre cette figure, cette représentation sans doute très utile pour de petits documents devient vite inexploitable pour des documents complexes.

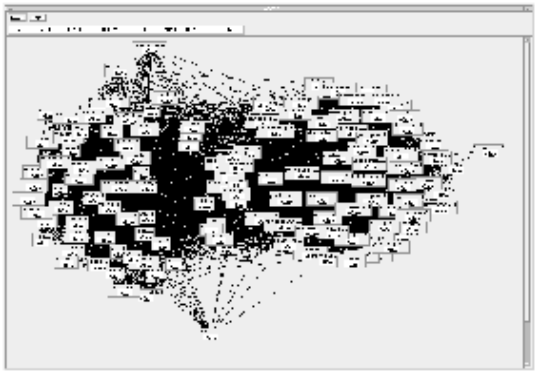


Figure III-20 : Visualisation de la structure hypertexte d'un document

Des techniques à base de filtres (sur le type des liens, ou sur le type de document, la localité des documents) doivent être utilisées pour alléger l'affichage.

Dans le cadre de l'édition de documents, peu de systèmes intègrent de tels outils. La métaphore la plus communément utilisée est la visualisation du lien par une icône sur le média. Les systèmes n'offrent pas une vue globale de la structure.

Dans le cadre de notre environnement idéal, la vue hypertexte doit permettre de visualiser ces différentes informations (liens de navigation, interdépendance entre documents).

Aucun système ne propose une visualisation de l'espace précis de navigation du lecteur. Ce besoin, difficilement satisfaisable dans le cadre de document hypertexte classique du fait que l'on ne connaît pas ce que le lecteur a réellement lu d'un document, peut être satisfait dans le cadre des présentations temporisées.

La *vue rapport d'exécution*, permet à l'auteur de visualiser une exécution réelle de son document, et de ce fait, elle peut être utilisée pour visualiser l'espace de navigation accédé au cours de la présentation par le lecteur.

3.2.5 Visualisationde l'exécution du document

Le système auteur idéal doit permettre à l'auteur de visualiser les différentes informations liées à l'exécution de son document. Ces informations sont :

- Visualisation d'une exécution a priori : le système auteur au travers de la vue temporelle offre une visualisation a priori de l'enchaînement temporel du document. Le choix de la solution présentée peut-être paramétré pour choisir par exemple quels liens le lecteur activera, et à quel moment se fera leur activation.
- Visualisation d'une exécution et donc de la dynamicité du document : la vue de présentation du document permet à l'auteur de visualiser une exécution. Il est opportun d'offrir à l'auteur des fonctions telles que :
  - *Joue, qui permet à l'auteur de lancer l'exécution du document à partir du début.*
  - *Joue_à(instant) démarre l'exécution à un instant donné.*
  - *Joue_à(objet) lance l'exécution du document à partir de l'objet spécifié*

Ces trois fonctions sont très utiles lors de la phase d'édition car elles permettent à l'auteur de visualiser le document selon une localité temporelle.

- Visualisation du résultat d'une exécution : la vue rapport d'exécution permet à l'auteur de visualiser le résultat réel d'une exécution (activation de liens). Se pose alors le problème de l'édition dans cette vue rapport d'exécution. Editer le document au travers de cette vue peut paraître plus simple pour l'auteur, par exemple pour corriger un comportement non désiré. Cependant cela pose le problème de la pertinence de cette édition vis-à-vis du document qui, lui, représente un ensemble d'exécutions. En effet, quelles garanties l'auteur peut-il avoir sur sa modification vis-à-vis de tous les scénarios possibles ?

3.2.6 Visualisation des relations contenues dans le document

Une des difficultés rencontrées dans les différentes dimensions est de visualiser les relations que l'auteur a spécifiées entre les différents objets et leurs conséquences.

Cette visualisation peut se faire à différents niveaux. On peut, par exemple, enrichir chacune des vues pour afficher cette information, c'est le cas de la visualisation des arcs de synchronisation dans la vue temporelle de Grins. On peut aussi visualiser explicitement le graphe de dépendance, comme c'est réalisé dans le logiciel FireFly [Buchanan93] (Figure III-21). Dans cette figure, on voit que le système représente explicitement les instants de début et de fin des objets, les synchronisations étant représentées par des liens entre les noeuds. L'inconvénient majeur de ce genre de représentation est qu'il ne donne pas une représentation temporelle de l'enchaînement des événements.

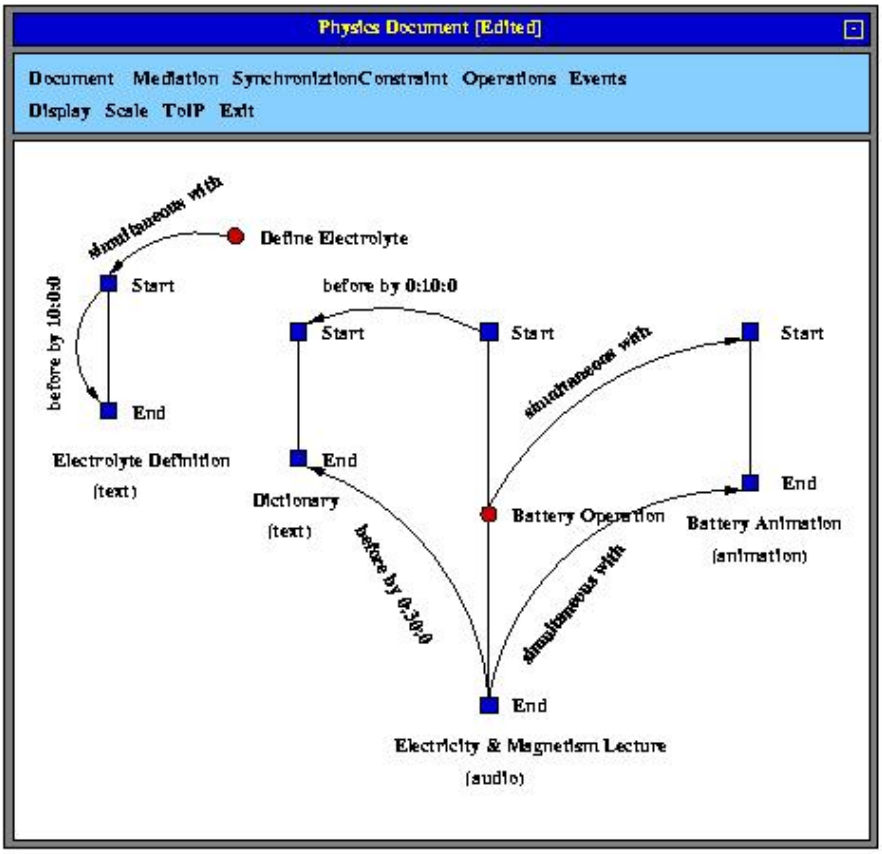


Figure III-21 : Graphe de synchronisation de Firefly

Parmi les outils qui offrent une visualisation de relations dans des domaines autres que l'édition multimédia, on peut citer Action Works d'IBM (Figure III-22) qui permet à l'auteur de visualiser de manière abstraite l'ensemble des relations qui composent un Workflow. Un workflow est la description d'un ensemble de tâches qui sont liées les unes par rapports aux autres temporellement. Nous donnerons une définition plus précise dans le chapitre VI section 3.

Sur la Figure III-22 on peut voir que différentes métaphores graphiques sont utilisées en fonction du type de relations que l'on visualise. Le système représente les différentes possibilités par des icônes différentes (rendez-vous, condition, enchaînement automatique, enchaînement en cas d'erreur). Cette représentation permet facilement de voir les dépendances entre les tâches, cependant elle ne permet pas de percevoir l'enchaînement temporel de ces dernières.

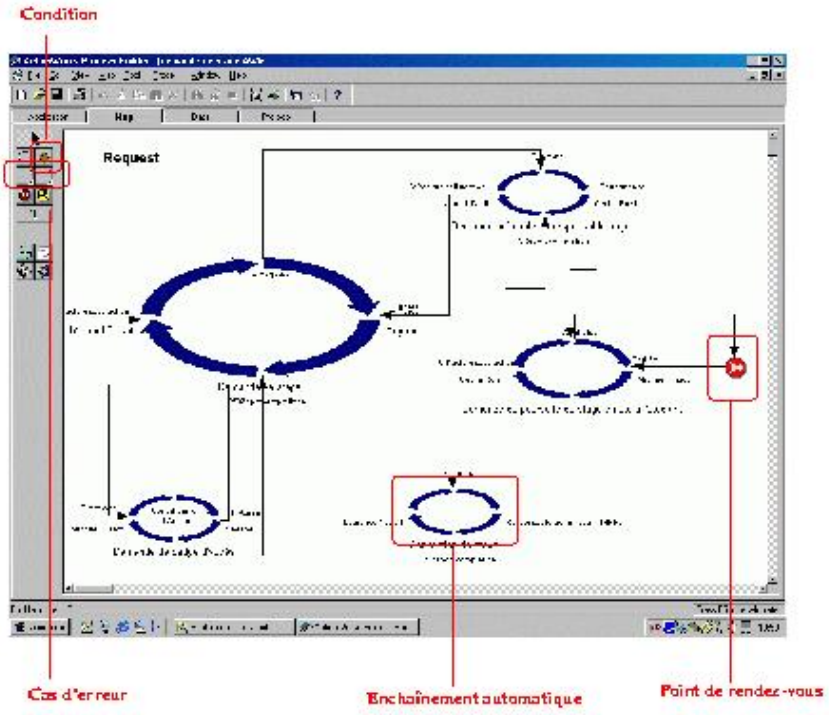


Figure III-22 : Vue d'édition d'Action Works Project Builder

Le système doit aussi fournir des mécanismes de synchronisation forte entre chacune des vues et la liste des relations. Par exemple, lorsque l'on

sélectionne un objet dans la vue spatiale, les relations impliquées dans son placement doivent être aussi visualisées.

On retrouve ce type de fonction dans le logiciel Cabri [Laborde95]. Cabri permet entre autres de dessiner des figures géométriques en spécifiant des relations entre les objets. Ainsi, l'extension réalisée par Bellynck [Bellynck99] permet de synchroniser la liste des relations avec les objets manipulés.

4. Parcours dans le document multimédia

Dans de nombreux domaines (hypertexte, recherche d'informations) des métaphores graphiques (boîtes, ressorts, arbres coniques) sont utilisées pour permettre à l'auteur de mieux appréhender les gros volumes d'information comme c'est le cas de certains documents multimédias. Cependant ces métaphores graphiques ne sont pas suffisantes ([Morse99], [Wilcox97]). Il est important de compléter ces métaphores graphiques statiques par des moyens de parcours à l'intérieur du volume d'informations offert à l'auteur. La barre de défilement en est un exemple simpliste, mais d'autres mécanismes peuvent être proposés comme les arbres hyperboliques. La représentation graphique choisie permet d'afficher à l'auteur l'information qui est essentielle à son travail en cours, tout en lui permettant d'accéder aux autres informations par des manipulations simples.

Nous allons nous intéresser dans cette partie à des techniques de navigation développées dans l'environnement idéal de manière à aider l'utilisateur dans sa tâche de compréhension d'un gros volume d'information.

4.1 Type de parcours offerts dans l'environnement idéal

Des techniques ont été développées pour permettre à l'auteur de naviguer de façon plus rapide à l'intérieur de son document. On peut classer ces techniques en trois catégories :

- Navigation basée sur la structure du document : c'est, par exemple, les types de navigation que l'on retrouve dans Thot et Word qui permettent de naviguer de titre en titre, de figure en figure, ou, plus généralement, sur le type ou sur la valeur d'un des attributs des noeuds du document.
- Navigation basée sur la temporalité du document : on peut citer les fonctions d'avance rapide, de retour, de pause/resume qui permettent à l'auteur de naviguer temporellement dans le document. Ces fonctions peuvent être mises en oeuvre dans la vue de présentation mais aussi dans la vue temporelle.

Les modes de parcours peuvent également tirer parti des services de synchronisation entre les vues. On peut par exemple imaginer une synchronisation sur les objets en cours de visualisation entre ces vues de navigation et les autres vues de l'application. Cela pourrait se faire au moyen d'une métaphore graphique qui, par exemple, dans la vue hiérarchique se traduirait par le changement de couleur des objets en cours de visualisation.

On peut citer aussi les techniques de navigation plus évoluées qui se basent sur les instants clés du document (images clés représentant les instants de début ou de fin d'objets (voir Figure III-23) ([Layaida99])), les techniques qui permettent de compresser certaines parties temporelles. On peut noter que ces techniques sont largement utilisées dans les outils de création et de manipulation de vidéo [Adobe-premiere00].

Figure III-23 : Visualisation des instants clés

4.2 Parcours de l'espace de solutions du document

Nous avons vu au cours du chapitre précédent que lors de l'édition de documents à base de relations flexibles, l'auteur ne spécifie pas une présentation de son document mais un ensemble de présentations. Le système doit donc permettre à l'auteur de naviguer dans cet espace. Un moyen simple de réaliser cela est de permettre à l'auteur de manipuler directement la flexibilité des relations [Tardif97]. Un besoin important lors de ces opérations est la perception de l'espace des déplacements possibles des objets avant toute manipulation. Par exemple, lorsque l'auteur sélectionne un objet dans le but de le déplacer dans la dimension spatiale ou temporelle, il est important que le système l'informe de l'intervalle dans lequel il pourra effectuer ce déplacement sans enfreindre les différentes relations du document qu'il a déjà spécifiées. Le système auteur doit donc dans un premier temps calculer l'intervalle de déplacement possible de l'auteur et dans un deuxième temps le rendre perceptible à ce dernier, par exemple en l'affichant [Tardif97].

Cet intervalle n'est pas seulement dû aux propriétés intrinsèques de l'objet, mais aussi à son contexte dans le document et notamment aux différentes relations directes ou indirectes qui le lient aux autres objets.
Par exemple dans la Figure III-24, si l'auteur définit le document suivant :

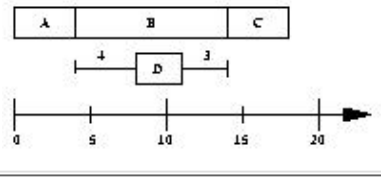
Objets	Relations	
A: [4] B: [10] C: [4] D: [1 3 6]	A Suivi_de B B Suivi_de C A Avant D D Avant C	

Figure III-24 : Besoin d'anticipation en cours d'édition

La solution courante est affichée dans la Figure III-24, avec A=4, B=10, C=4, D=3. Maintenant supposons que l'auteur déplace l'objet D. Il est important de l'informer que ce déplacement peut se faire uniquement de quatre unités vers la gauche ou de trois unités vers la droite (par une borne de déplacement comme sur la figure, par exemple).

4.3 Synthèse sur le parcours des informations contenues dans le document

Les différentes vues spécifiées dans la section 2.3 répondent aux besoins de visualisation définis dans le chapitre précédent (section 2.2.2), en permettant à l'auteur de comprendre et de manipuler les différents aspects de son document. De plus, certaines vues, comme la vue temporelle et spatiale, permettent à l'auteur d'éditer simplement le document, par exemple, en plaçant directement les objets à leur emplacement temporel ou spatial.

Le parcours des informations contenues dans le document au travers des vues permet à l'auteur, entre autre, de visualiser l'espace de solutions défini dans son document et lui permet de parcourir, selon ses besoins, les différentes parties du document. De plus, la combinaison des vues et des fonctions de navigation permet à l'auteur d'avoir à sa disposition plusieurs manières pour effectuer chacune des opérations d'édition. Il peut alors choisir celle qui convient le mieux à l'opération qu'il fait ou la plus adaptée à la succession d'opérations qu'il est en train de faire. Nous illustrerons cela dans l'exemple de session d'édition dans la section qui suit.

5 Illustration au travers d'un exemple

De manière à illustrer ce chapitre et le type d'environnement auteur que nous proposons, nous allons terminer ce chapitre par un exemple de session d'édition avec un tel environnement.
Nous allons utiliser l'environnement auteur idéal construit au dessus de Madeus. Ce choix, est purement arbitraire, l'édition se faisant de manière similaire dans tous les environnements auteur.
Nous allons donc reprendre le document INRIA présenté dans le chapitre précédent.
Dans un premier temps, l'auteur au moyen de l'interface va créer ces cinq médias texte (`"Institut", "National", "de Recherche ", "en Informatique", "et Automatique"`).
Dans l'exemple de la Figure III-25, on peut voir cette création. L'auteur a plusieurs possibilités pour créer ces objets. Par exemple, il peut insérer l'objet directement dans la vue spatiale ou hiérarchique (flèches 1 de la figure), et ensuite positionner les attributs soit par manipulations directes dans les vues (attributs spatiaux dans la vue de présentation et attributs temporels dans la vue temporelle) ou grâce à la vue attributs (flèches 2 de la figure).

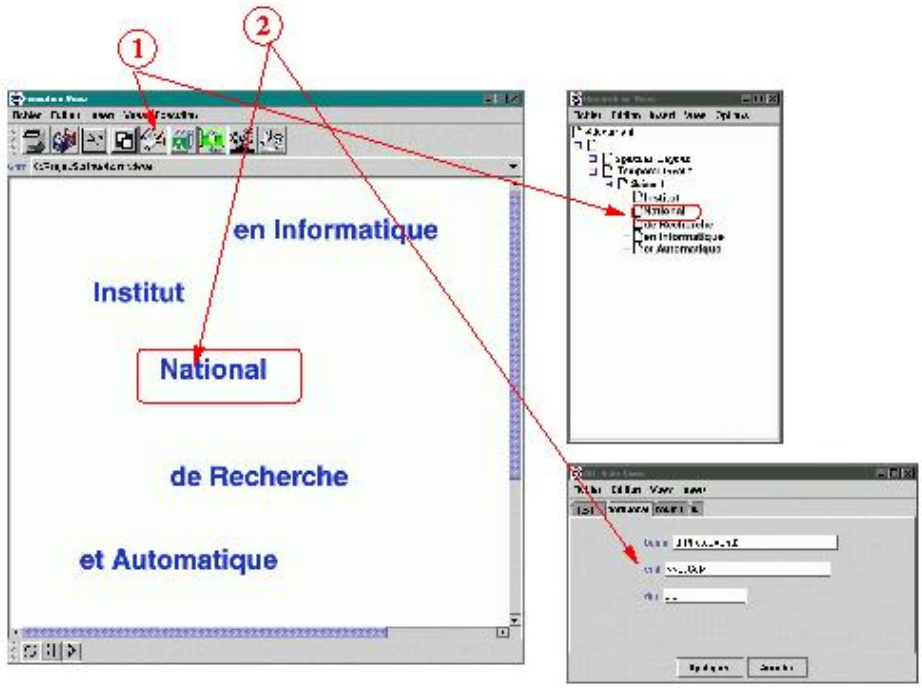


Figure III-25 : Ajout d'un élément texte

Dans un deuxième temps, il va définir la structure spatiale et créer un composite qui contient ces cinq médias. Il va ensuite mettre quatre relations pour aligner ces objets horizontalement (`<AligneAGauche param1="media1" param2="media2" delta="+60">`), et quatre relations pour aligner ces objets verticalement (`<Under param1="media1" param2="media2" delta="+20">`). Dans l'exemple de la Figure III-26, l'auteur insert une relation spatiale entre les objets "National" et "et Automatique". Pour cela il peut sélectionner les objets dans la vue qu'il préfère (dans l'exemple, la sélection

s'effectue à l'aide des vues spatiale et hiérarchique), et il positionne une relation à l'aide de la palette de relations.

Une fois le positionnement spatial défini, il peut définir la structure temporelle en définissant un composite qui contient les cinq médias et placer les objets les uns par rapport aux autres avec des relations spatiales et/ou temporelles (exemple : `<start param1="media1" param2="media2" delay="10s"> ).`

Dans l'exemple de la Figure III-27, l'auteur insert une relation temporelle entre deux objets. Comme pour les relations spatiales il peut sélectionner les objets dans la vue qu'il désire, et positionne une relation grâce à la palette temporelle.

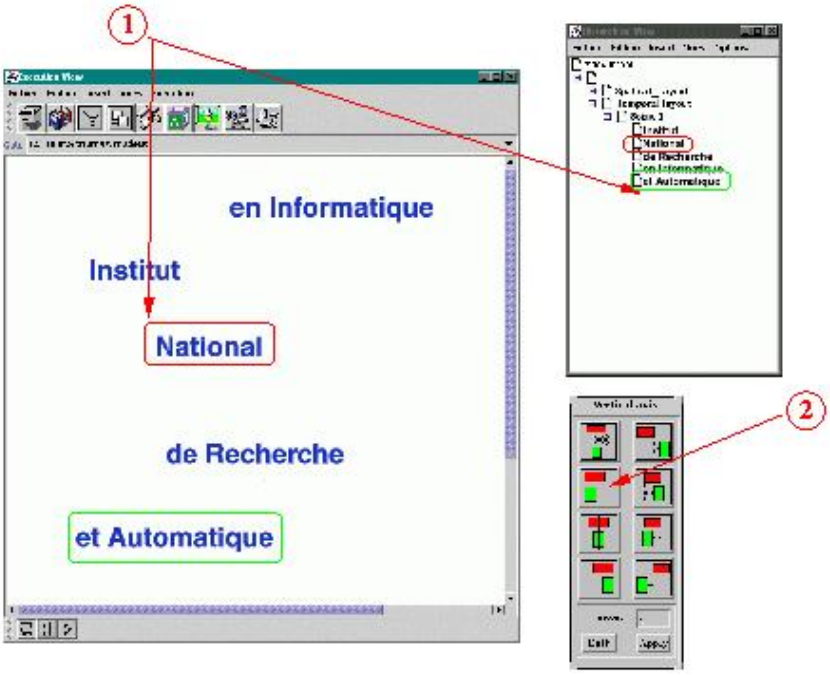


Figure III-26 : ajout de relation spatiale

Une fois ces définitions faites, il peut placer spatialement le média qui apparaît en premier là où il le désire dans la vue présentation. Du fait que le système auteur maintient les relations, les autres objets seront toujours placés en cascade les uns par rapport aux autres. Il fait de même pour le premier objet temporel.

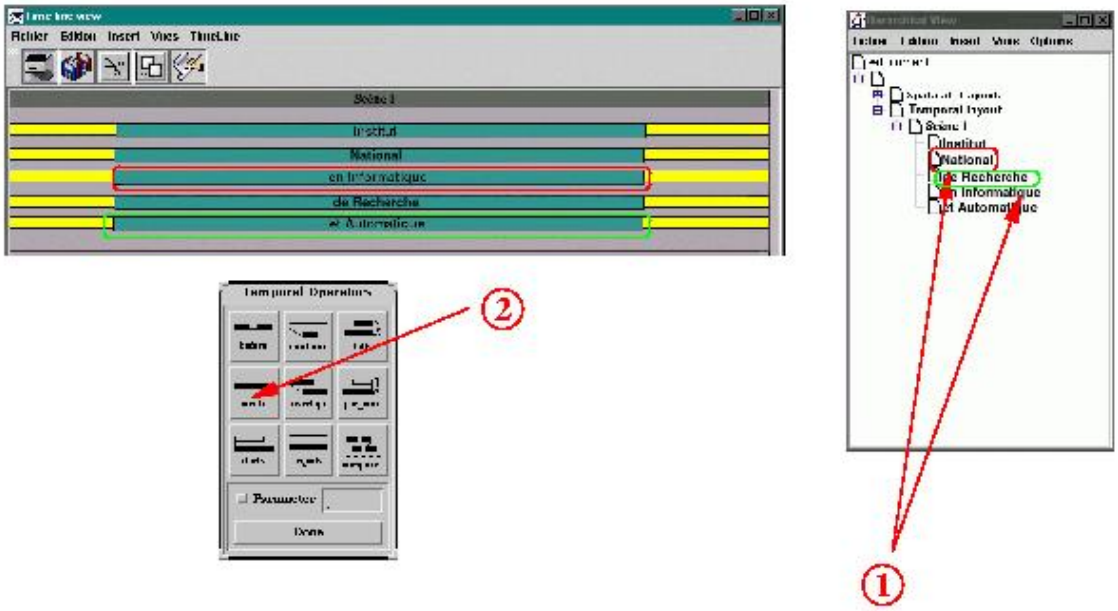


Figure III-27 : Ajout de relation temporelle

A tout moment au cours de ce processus d'édition l'auteur peut exécuter son document de manière à avoir un aperçu de son document.

On peut noter que s'il désire rajouter une image de titre, il lui suffit de créer le média, de le placer spatialement et temporellement, et si nécessaire de mettre une relation avec un des autres objets pour que la présentation s'adapte automatiquement. De la même façon, s'il désire que sa présentation fasse 10 secondes de plus, il lui suffit de spécifier cette durée sur le document, et le formateur modifiera les durées sur les objets pour prendre en compte cette modification.

On peut aussi imaginer que l'auteur veuille utiliser cette présentation pour réaliser un autre document, par exemple la présentation de la boîte à outils Kaomi. Pour cela, il remplace le contenu des cinq médias texte par ("Kaomi :", "une boîte", "à outils", "pour la construction",

"d'environnements auteur". Le système ajustera alors le placement spatial des cinq objets en conservant les relations spatiales spécifiées.

6 Bilan et perspectives de l'environnement idéal

Au cours de ce chapitre nous venons de proposer un formalisme d'édition et un environnement auteur pour la spécification de documents multimédias. Dans l'exemple terminant ce chapitre, nous avons pu voir qu'il était relativement simple de définir le document, et qu'il était relativement simple de le modifier.

L'hypothèse que nous avons faite, qui était de fournir un environnement auteur basé essentiellement sur un formalisme d'édition relationnelle (tout en permettant à l'auteur d'utiliser le formalisme du fichier de sauvegarde), répond dans un premier temps aux différents besoins énoncés dans le chapitre II (section 2.1 et 2.2).

Les différentes propositions qui ont été faites dans ce chapitre permettent aussi de répondre aux différents besoins de visualisation et de présentation définis dans le chapitre II.

L'environnement résultant permet donc d'apporter à la fois une réponse sur la simplicité d'édition ainsi que sur les services d'aide à l'auteur tel que la visualisation, la vérification de cohérence et le formatage. Ce sont généralement ces points qui limitent l'utilisation des outils actuels.

Nous allons nous intéresser maintenant à la réalisation d'un tel environnement. L'environnement que nous allons proposer ne satisfait pas tous les points décrits dans ce chapitre. Cette limitation porte essentiellement sur les trois points suivants :

- visualisation de la structure hypertexte ;
- visualisation de gros volumes d'information ;
- aide au diagnostique.

Pour ces trois points de nombreux travaux sont en cours et ils ne sont pas liés à la dimension temporelle des documents multimédias.

Nous apporterons par contre une réponse à tous les autres points. La volonté de valider les principes d'édition pour de nombreux langages représentatifs des différentes approches de spécifications a motivé le choix de réaliser une boîte à outils de manière à mettre en commun le maximum de compétence. Les points difficiles dans la réalisation de cette boîte à outils sont :

- la mise en oeuvre du formalisme d'édition ;
- la réalisation des différentes vues ;
- la gestion du placement temporel et spatial des éléments ;
- la vérification de la cohérence du document.

La présentation des réponses à ces problèmes se fera en trois temps. Nous présenterons dans un premier temps l'architecture de la boîte à outils ainsi que la mise en oeuvre du formalisme d'édition et des différentes vues dans le chapitre IV.

Dans un deuxième temps (chapitre V), nous présenterons la réalisation des deux derniers points, que nous avons choisi de réaliser à l'aide de technologies à base de contraintes. Le choix d'utiliser des techniques à base de contraintes est motivé par le fait que nous voulons apporter une réponse globale aux problèmes de placement et de cohérence pour les différents langages de documents multimédias que nous allons expérimenter.

Dans un troisième temps (Chapitre VI) nous présenterons la réalisation de différents environnements auteur de documents multimédias.