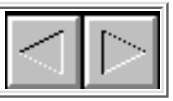

Chapitre VI : Expérimentations, évaluations et extensions de Kaomi

1. Introduction

Comme nous l'avons introduit précédemment, un des objectifs de la thèse est de valider la boîte à outils Kaomi au travers de la réalisation de plusieurs environnements auteur reposant sur des formalismes suffisamment différents pour montrer les possibilités d'utilisation de la boîte à outils.

Nous présenterons dans un premier temps les outils auteur de documents multimédias construits à partir de formalismes relationnels et événementiels (section 2). Dans un deuxième temps, nous présenterons deux expériences d'utilisation de la boîte à outils différentes des précédentes puisqu'elles visent à construire d'une part un éditeur de workflow (section 3), et d'autre part à utiliser Kaomi dans un contexte de base documentaire (section 4).

Ce travail que j'ai initié tant d'un point de vue de la conception que de la réalisation a impliqué de nombreuses personnes au sein du projet Opéra, tant d'un point de vue programmation (ingénieurs, doctorants, stagiaires) que d'un point de vue recherche. De manière à bien identifier les apports et les contributions de chacun, la section 5 présentera un bilan quantitatif des différentes participations dans la boîte à outils ainsi que dans les différents environnements auteur construits grâce à elle. Nous ferons dans la section 6 un bilan des différentes expériences réalisées avec Kaomi. De ce bilan et de l'expérience acquise au cours de ces réalisations nous dégagerons un ensemble de propositions pour étendre cette boîte à outils (section 7).

2. Les outils auteur de documents multimédias réalisés avec Kaomi

Les environnements auteur de documents multimédias construits au-dessus de Kaomi permettent d'éditer des langages qui couvrent un large échantillon des formalismes de spécification de documents multimédias (voir Chapitre II section 3).

Les environnements auteur réalisés recouvrent :

- le formalisme relationnel :
  - à base de relations : Madeus-Editeur (section 2.1) ;
  - à base d'opérateurs : SMIL-Editeur [Navarro00] (section 2.2);
- le formalisme événementiel : MHML, illustré par la réalisation de l'environnement MHML-Editeur (section 2.3).

2.1 Réalisation de Madeus-Editeur

Madeus-Editeur est un environnement auteur de documents au format Madeus (voir Chapitre II section 3.4.2). Nous avons dû, dans un premier temps étendre, le parseur pour permettre la construction de la structure de données de Madeus, et enfin, nous avons rajouté deux fichiers de ressources, pour définir la sémantique des relations et des opérateurs du langage Madeus. La création de cet éditeur a été simplifiée car les formalismes d'édition proposés par Kaomi et Madeus-Editeur sont très proches.

La traduction des différentes relations de Madeus en rélems a servi à illustrer les mécanismes de Kaomi dans le chapitre IV, nous ne présenterons ici que la traduction de la relation spatiale "centrer" en rélems (Figure VI-1).

CentrerHorizontal (A,B)	A.Gauche + d1 = B.Gauche A.Droite ± d2 = B.Droite d1=d2
CentrerVertical (A,B)	A.Bas + d1 = B.Bas A.Haut ± d2 = B.Haut d1=d2

Figure VI-1 : Traduction des relations spatiales "centrer"

L'environnement ainsi réalisé permet à l'auteur d'éditer le document dans un ensemble de vues, chacune de ces vues étant synchronisées avec les autres. Les services offerts par Madeus tirent parti se ceux de la boîte à outils : cohérence, édition par manipulation directe, formatage.
L'utilisation des résolveurs de contraintes nous a permis notamment de permettre l'édition par manipulation directe dans la vue temporelle.
Lors de la sélection d'un objet dans la vue temporelle, le système visualise l'intervalle de déplacement possible pour l'objet ou l'intervalle de retaillage possible. De plus, le système met en évidence les objets qui seront impliqués dans la déformation de l'objet (à cause des relations) par un changement de couleur, que ce soit lors un déplacement ou d'un retaillage.

Dans l'exemple de la Figure VI-2 l'auteur a sélectionné l'objet Photo4, le système visualise l'intervalle dans lequel l'auteur peut déplacer cet objet. Par exemple, le déplacement vers la gauche de l'objet Photo4 est limité par le fait que cet objet possède une relation *meet* avec l'objet Photo3. Lorsque l'auteur déplace l'objet (Figure VI-3) la vue se met à jour de manière continue. Le calcul des nouvelles positions se fait grâce aux résolveurs de contraintes. Les objets impliqués par la modification de l'auteur sont les objets Photo2 et Photo3. Le système interdira à l'auteur tout déplacement en dehors de l'intervalle permis. L'auteur peut aussi retailler son objet (Figure VI-4), comme pour un déplacement, la vue se mettra à jour en temps réel et interdira les déplacements non permis.

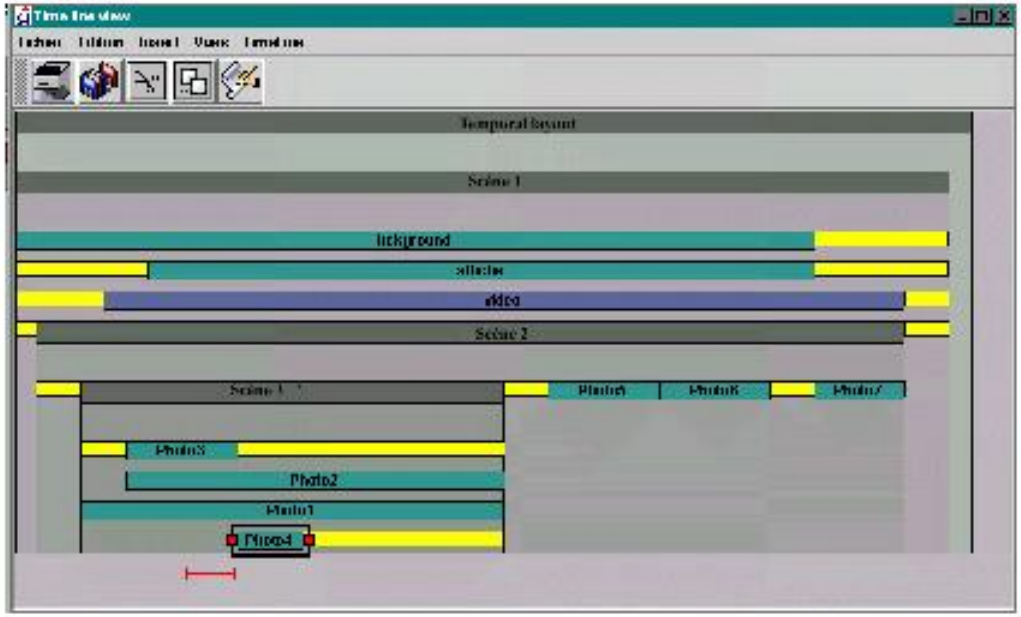

Figure VI-2 : Sélection d'un objet dans la vue temporelle de Madeus Editeur

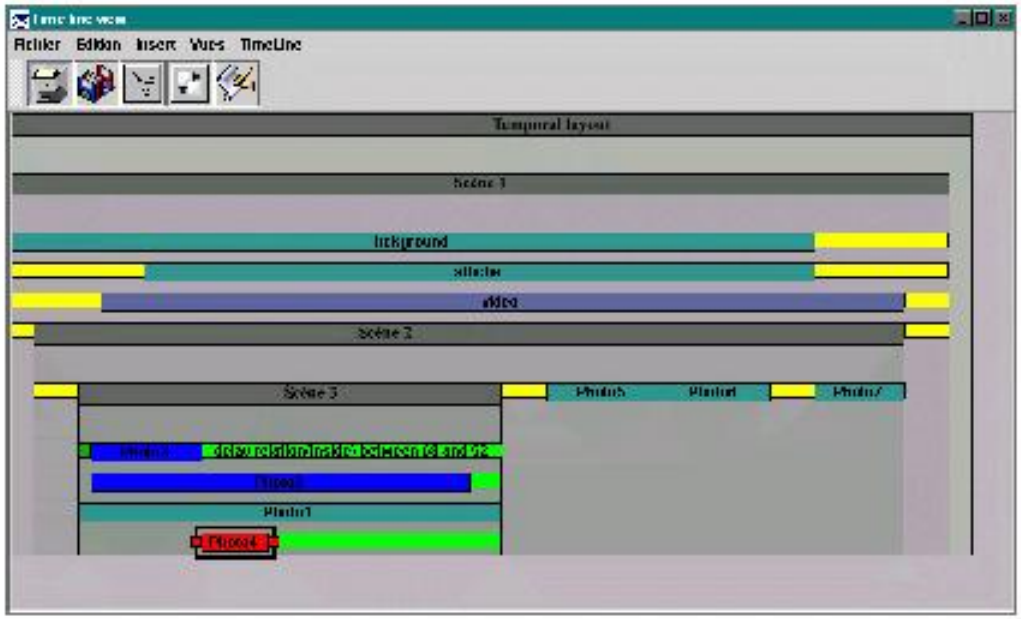

Figure VI-3 : Déplacement d'un objet dans la vue temporelle de Madeus Editeur

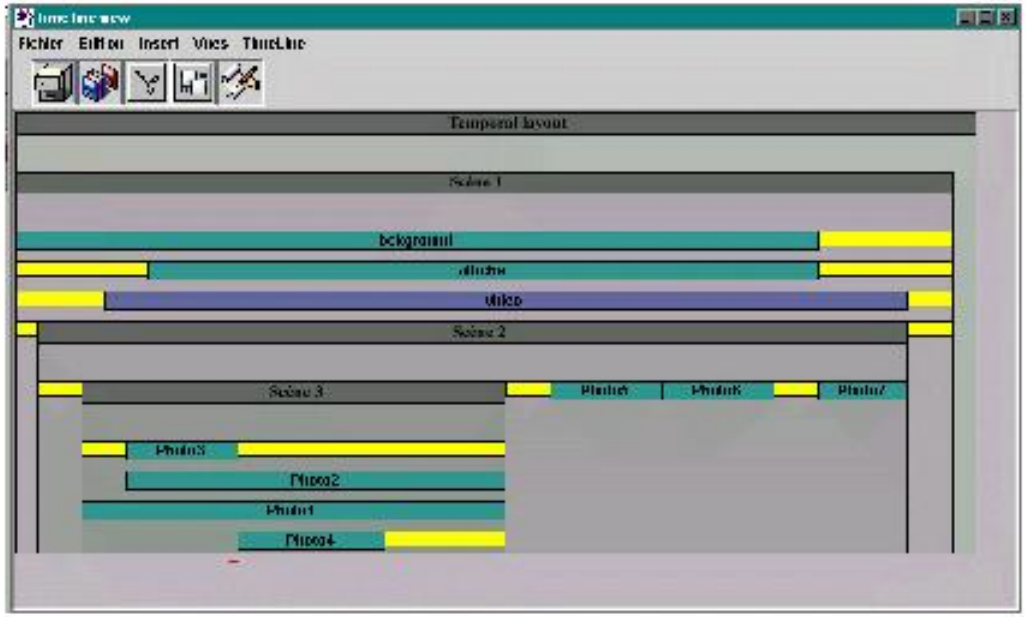

Figure VI-4 : Retaillage d'un objet dans la vue temporelle de Madeus Editeur

Par rapport à la version précédente de Madeus (Madeus-97), ce sont essentiellement les services d'aide à l'édition qui ont été ajoutés. Parmi ces services, on peut noter :

- *L'édition incrémentale*, c'est-à-dire que lors de l'édition toutes les vues sont synchronisées et mises à jour après chaque opération d'édition.
- *La vue temporelle* manipulable, qui permet à l'auteur de modifier intéractivement les différents attributs temporels de ces éléments ainsi que de naviguer dans l'espace de solutions.
- *Le formatage*, qui permet de dégager l'auteur de la spécification précise de la durée de ces médias quand cela est possible.
- *La vérification de cohérence qualitative et quantitative*après chaque opération d'édition.

2.2 Réalisation de SMIL-Editeur

SMIL-Editeur [Jourdan 99a, Navarro2000] est un environnement auteur de documents au format SMIL. Pour cet éditeur, nous avons dans un premier temps réalisé les mêmes opérations que dans le cadre du langage Madeus, c'est-à-dire que nous avons traduit les relations SMIL en termes de rélems. Cette traduction est contextuelle, c'est-à-dire qu'un élément est traduit en ayant une connaissance de ces fils et des différents attributs positionnés sur ceux-ci. Dans la Figure VI-5 nous pouvons voir les traductions des principales relations définies dans SMIL.

<Par dur="12s"> <txt id="A" ../> <txt id="B" ../> </Par>	Par.Début = A.Début A.Début = B.Début Par.Fin => A.Fin Par.Fin => B.fin /* car nous ne connaissons pas la durée de A et B */
<Par endsync="first"> <txt id="A" dur="3s" /> <txt id="B" dur="8s"/> </Par>	Par.Début=A.Début A.Debut = B.Debut A.Fin Þ B.Fin A.Fin Þ Par.Fin /* car la durée de A est inférieur à la durée de B */
<Seq > <txt id="A" dur="3s" /> <txt id="B" dur="8s"/> </Seq>	Seq.Début = A.Début A.Fin = B. Début Seq.Fin Þ B.Fin
<Par > <txt id="A" dur="3s" begin="end (B)"/> <txt id="B" dur="8s"/> </Par>	Par.Début = B.début A.Début= B.Fin Par.Fin = A.Fin
<Par dur="12"> <txt id="A" dur="10" begin="5"/> <txt id="B" dur="16"/> </Par>	Par.Début=B.Début A.Début = B.Début A.Début D 5 Par.Fin Þ A.Fin Par.Fin Þ B.Fin

Figure VI-5 : Traduction des relations SMIL en rélems

Dans un deuxième temps, nous avons étendu les opérations d'édition permises dans Kaomi de manière à assurer la cohérence d'un ensemble de comportements propres à SMIL. Par exemple, lors de l'ajout d'un objet, nous lui affectons une région spatiale par défaut, de même, lors de la suppression d'une région, nous vérifions que cette région n'est utilisée par aucun objet.

De plus, dans cet éditeur, nous avons étendu l'expressivité du langage SMIL pour faciliter l'édition spatiale. Pour cela, nous avons donné la possibilité à l'auteur d'ajouter des relations spatiales entre les objets. Ces relations sont stockées dans le fichier source en utilisant le mécanisme des espaces de noms [XML99]. Ces relations sont donc retrouvées lors de la prochaine session d'édition. L'utilisation des espaces de noms rend inexploitable ces informations dans les autres outils d'édition.

Cet éditeur profite, comme Madeus-Editeur, des différentes fonctionnalités offertes par Kaomi. Parmi celles-ci, on peut noter la possibilité d'éditer directement le placement spatial et temporel dans les vues de présentation et temporelle, chose qui n'est pas permise actuellement par les autres outils d'édition de documents SMIL (voir Chapitre II section 4.1.2 et 4.3.2).

L'édition de la structure temporelle est facilitée par la vue hiérarchique (Figure VI-6). Cette vue permet à l'auteur d'ajouter facilement un ensemble de médias organisés temporellement, organisation que l'auteur ajustera par la suite dans la vue temporelle.

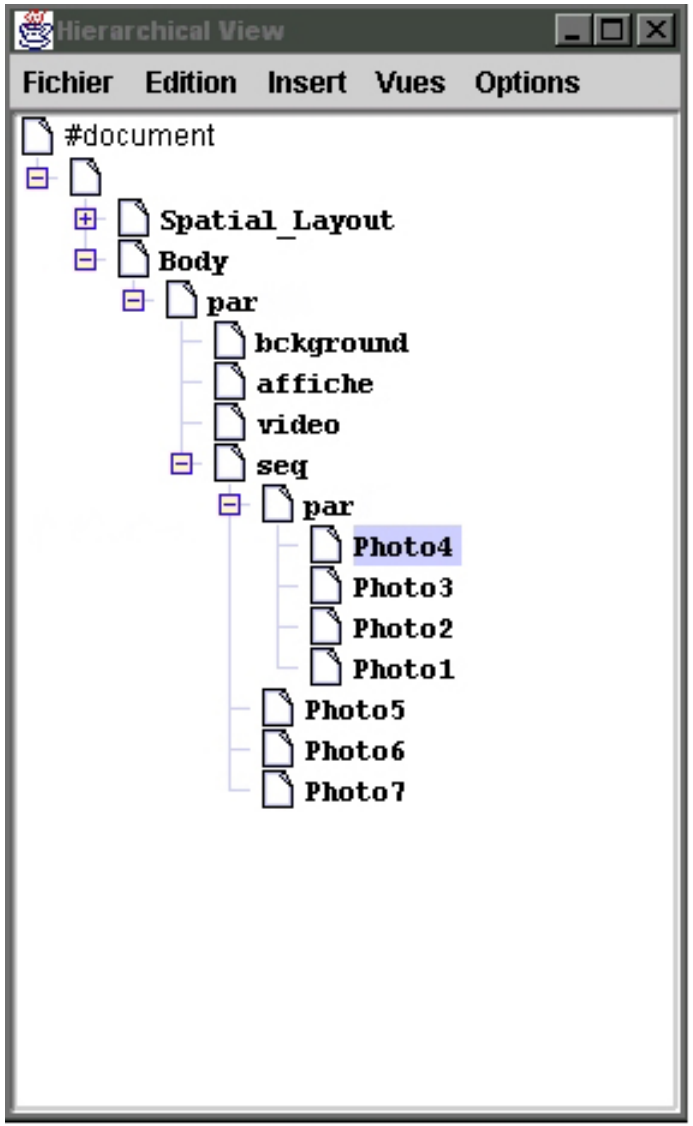

Figure VI-6 : Vue hiérarchique offerte par SMIL-Editeur

La vue temporelle permet de visualiser les différentes informations liées aux relations temporelles, mais aussi aux attributs temporels du langage SMIL. Dans la Figure VI-7, on peut voir les différentes informations temporelles présentées. On peut noter que les opérateurs de SMIL (*Par* et *Seq*) sont représentés par le mécanisme de boîtes englobantes fourni par Kaomi. Les attributs *begin* et *end* sont représentés explicitement par des délais reflétant les décalages temporels qu'impliquent les attributs (en jaune sur la figure). Enfin, les objets sont représentés par une boîte dont la longueur est proportionnelle à leur durée.

Les différentes manipulations permises sont le déplacement, le retaillage des objets et des composites. Ces fonctions sont directement issues de Kaomi.

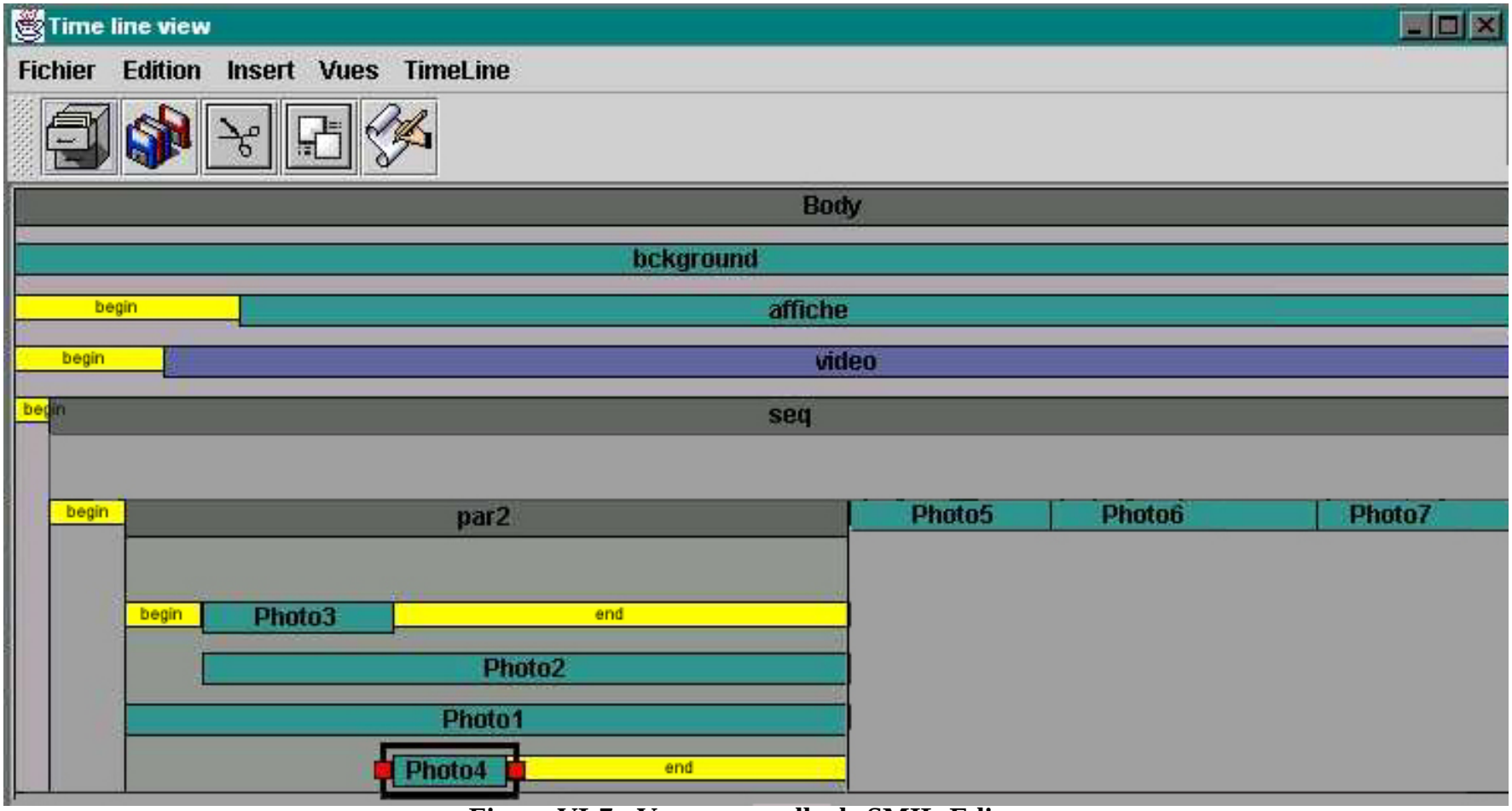

Figure VI-7 : Vue temporelle de SMIL-Editeur

Nous avons présenté dans le chapitre II des exemples d'environnement d'édition SMIL représentatif de ce qui existe aujourd'hui. Celui qui est le plus proche de notre éditeur SMIL-Editeur est certainement Grins (chapitre II section 4.3.2). Nous pensons cependant répondre plus complètement aux besoins des auteurs grâce aux fonctions suivantes :

- *L'édition incrémentale*, c'est-à-dire que lors de l'édition toutes les vues sont mises à jour.
- *La vue temporelle* manipulable, qui permet à l'auteur de modifier intéractivement les différents attributs temporels de ces éléments. Cette vue permet aussi de diminuer le nombre d'informations visualisées dans la vue temporelle grâce à l'ouverture/fermeture des éléments composites offerts par Kaomi.
- *La vue de présentation* qui permet à l'auteur d'éditer directement le placement spatial de ces objets.
- *L'intégration harmonieuse de deux formalismes d'édition* : celui du langage SMIL et celui de Kaomi pour le placement de relations spatiales et la définition des attributs dans un intervalle de valeurs.
- *Le formatage*, qui permet de calculer statiquement les instants de début et de fin des objets. Cela permet entre autre d'offrir une visualisation *a priori*

de l'enchaînement temporel défini dans le document.

- *Un premier niveau de vérification de cohérence qualitative*: nous vérifions qu'il n'y a pas de définition cyclique dans les attributs. Par exemple, lorsque la fin de l'objet A est définie relativement à la fin de l'objet B et réciproquement.

Néanmoins, l'outil proposé (contrairement à Grins) ne couvre pas complètement tous les traits du langage SMIL 1.0. L'opérateur *switch,* par exemple, n'est pas implémenté.

L'utilisation des techniques à base de contraintes a permis d'étendre facilement l'édition par manipulation directe dans les vues de présentation et temporelle pour le langage SMIL. De plus, ces techniques ont permis d'offrir un service de formatage qui permet de ne spécifier que partiellement les durées des objets de son document, laissant le soin au formateur de propager ces valeurs et de calculer une solution au document.

Dans la conclusion de cette thèse nous nous placerons par rapport à l'évolution de SMIL et notamment par rapport à la sortie de SMIL-2.0.

L'outil SMIL-Editeur a aujourd'hui servi de support à de nombreuses démonstrations que ce soit lors de conférences ou de réunions, néanmoins il manque une évaluation et un retour d'utilisateurs.

2.3 Réalisation de MHML-Editeur

L'éditeur MHML-Editeur, réalisé dans le cadre d'un contrat avec Alcatel, nous a permis d'expérimenter les modules liés à l'aspect imprédictif des médias et des événements.

Nous avons vu dans le chapitre II section 3.3 que la description des comportements temporel et spatial de MHML se fait via des événements. Parmi ceux-ci, certains sont prédictifs et donc traduits dans Kaomi sous forme de rélem, d'autres sont imprédictifs. Kaomi offre, de ce fait pour les événements prédictifs un service d'édition évolué avec manipulation dans les vues temporelle et de présentation. Par contre, même si Kaomi permet à l'auteur d'exprimer des comportements imprédictifs, elle n'offre pas de service d'assistance à l'édition (cohérence, manipulation directe).

Pour assurer un niveau acceptable de services d'édition, nous avons fait le choix de restreindre le langage MHML de manière à n'autoriser les événements imprédictifs que vers des îlots. Un *îlot* est une partie de document n'ayant pas d'événements imprédictifs entrant autres que ceux qui permettent de le démarrer et de l'arrêter.

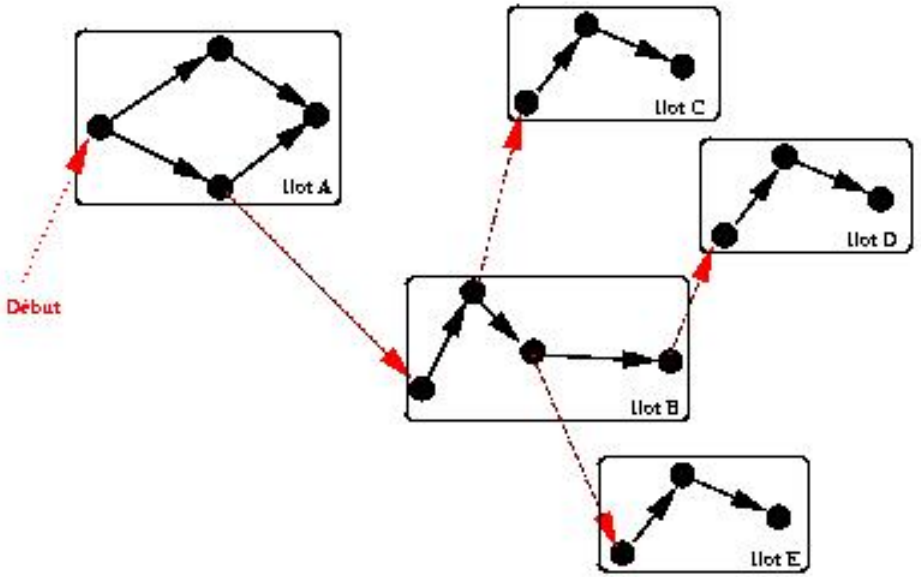

Figure VI-8 : Ensemble d'îlots d'un document MHML

Nous avons étendu Kaomi pour offrir une aide à l'édition de tels comportements. Cette édition se fait au travers d'une palette d'attributs qui permet d'éditer la structure des événements (Figure VI-9).

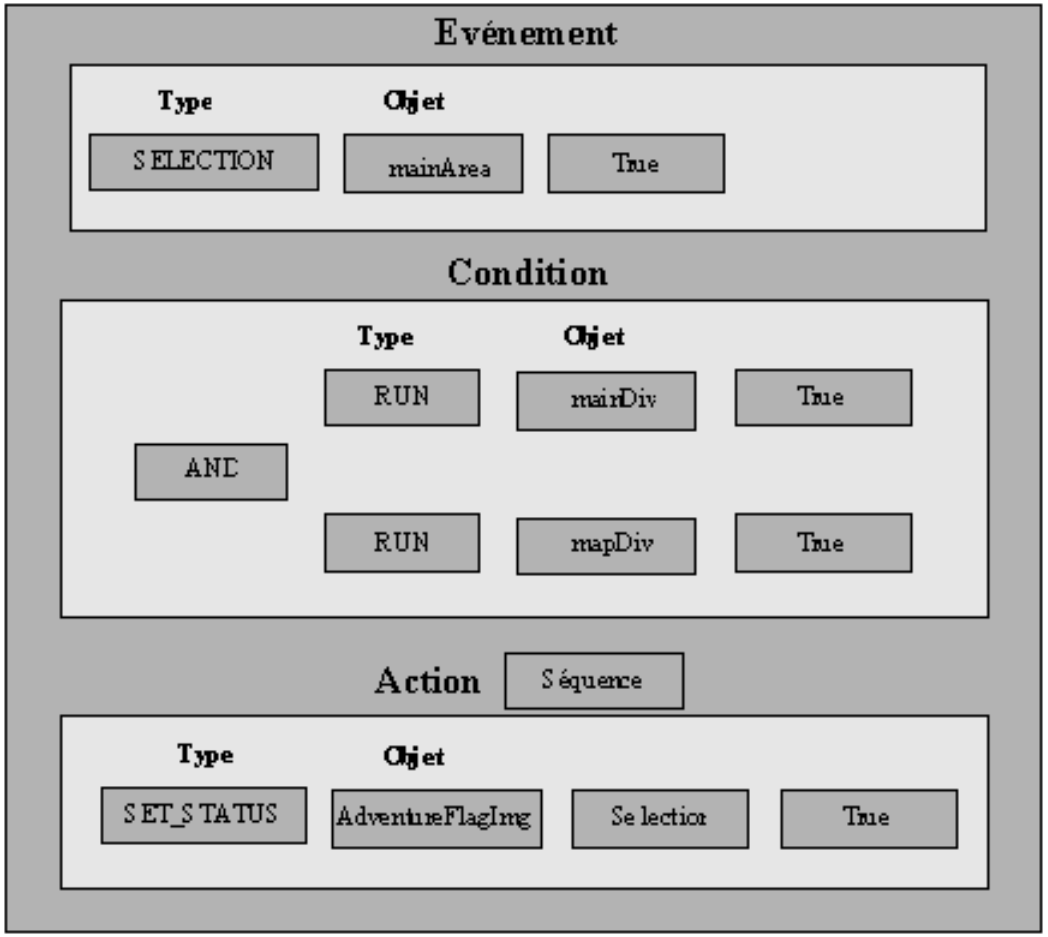

Figure VI-9 : Exemple de formulaire permettant le paramétrage d'événements

Par exemple, MHML-Editeur fournit un mécanisme de visualisation d'événements qui permet à l'auteur de percevoir des comportements imprédictibles, ainsi qu'un service de simulation qui permet d'obtenir des résultats d'exécutions possibles sans pour autant devoir exécuter le document (ce qui prendrait beaucoup plus de temps). Ce mécanisme de visualisation des événements repose sur un module de simulation qui permet de simuler des exécutions du document. Ce module a été réalisé au sein de Kaomi, permettant ainsi d'intégrer cette fonctionnalité dans d'autres outils auteur si le besoin s'en fait sentir. Ce module permettrait, par exemple, de simuler les comportements imprédictifs de SMIL 2.0.

Le choix qui a été fait permet de visualiser une solution possible parmi l'espace de solutions. La possibilité est donnée à l'auteur de choisir les événements qui ont lieu dans une boîte de dialogue, la date d'occurrence des événements étant choisie par le module de simulation. Dans la Figure VI-10, on peut voir un exemple de placement temporel avec quatre événements qui ont été déclenchés. Le premier est l'événement de début de document, les deux suivants ont été déclenchés, soit par une interaction utilisateur, soit par la fin d'un objet indéterministe et enfin le dernier est l'événement de fin de document qui dans ce cas a été déclenché par l'élément Body.

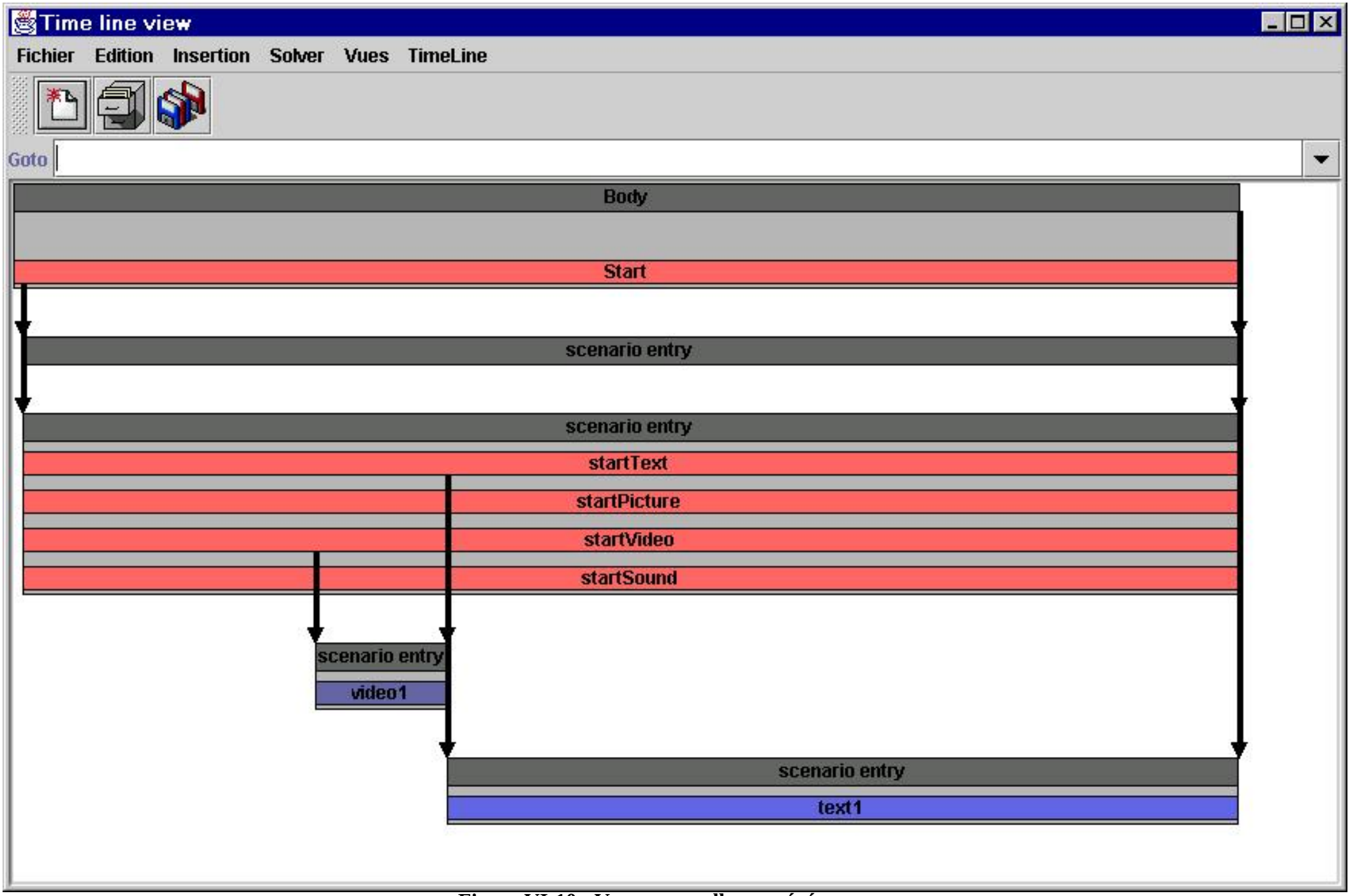

Figure VI-10 : Vue temporelle avec événements

À partir de cette représentation, l'auteur peut visualiser d'autres événements en les activant à partir de la boîte de dialogue. La vue s'adapte alors pour

prendre en compte l'occurrence de l'événement supplémentaire. Dans la Figure VI-11 l'auteur a cliqué sur StartVidéo pour déclencher un événement.

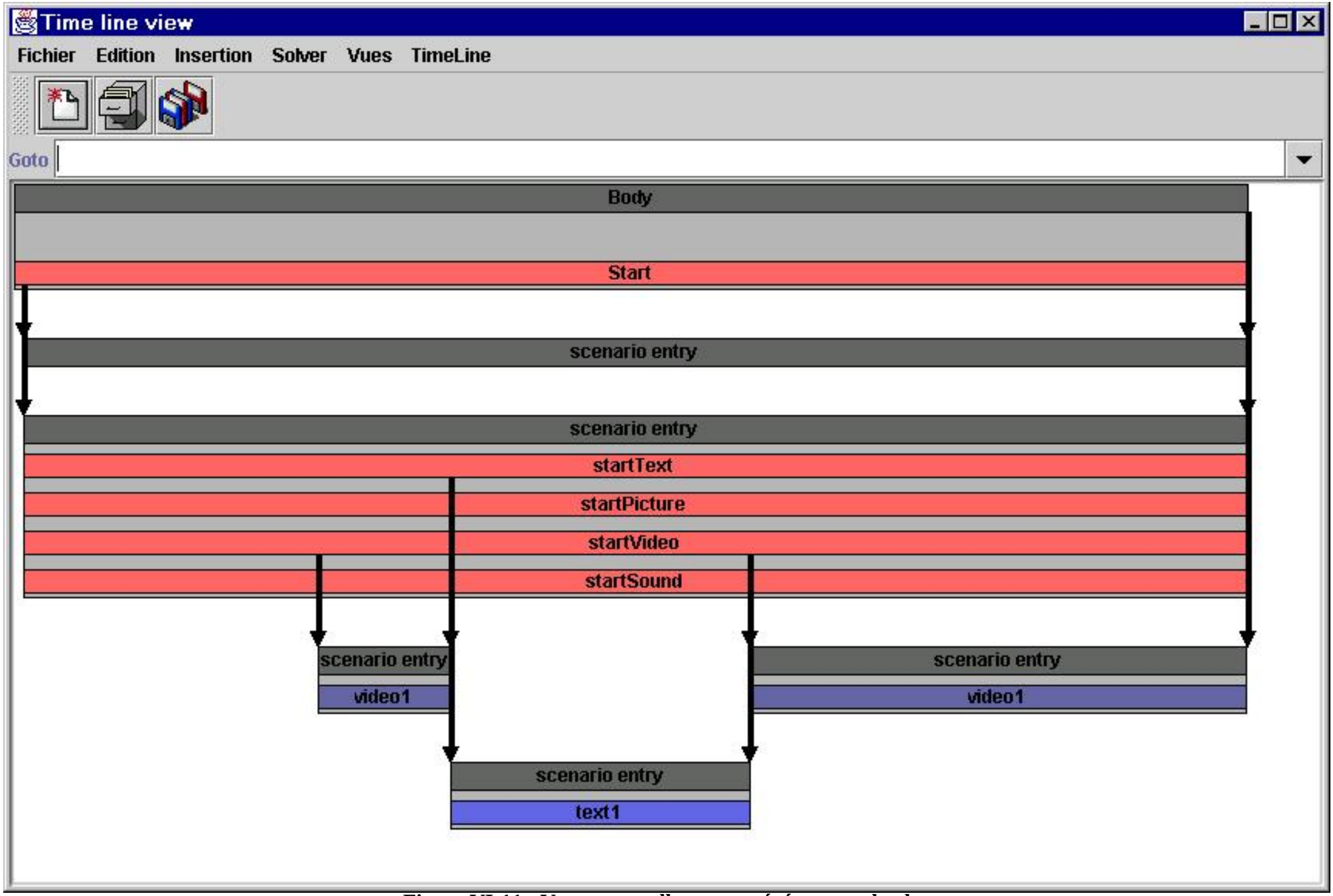

Figure VI-11 : Vue temporelle avec un événement de plus

Il serait intéressant d'étendre ce service en permettant à l'auteur de manipuler graphiquement les dates d'occurrence des événements.

Cet environnement peut être comparé à Lingo de Director (chapitre II section 4.3.1) de par l'expressivité qu'il permet à l'auteur. La différence essentielle vient de la facilité de spécification liée à l'utilisation de relations de haut niveau. Par rapport à Director, une autre différence essentielle est l'édition par manipulation directe avec maintien des relations (événements prédictifs) dans les vues temporelle et spatiale sur les parties prédictives du document.

L'environnement MHML-Editor permet à l'auteur de visualiser le résultat d'une exécution et de visualiser *a priori* plusieurs exécutions possibles dans différentes vues. Ces aides sont utiles à l'auteur de documents complexes pour visualiser le comportement de son document lors d'interactions avec le lecteur.

On peut noter, qu'il serait appréciable que des garanties puissent être fournies par le système auteur sans que l'auteur ne simule les différentes exécutions possibles (ce qui est impossible pour lui dans le cadre de documents imprédictifs) pour garantir certaines propriétés comme l'atteingnabilité de certaines parties de document, la durée minimum de présentation d'un média.

3. L'outil auteur de workflow réalisé avec Kaomi

Une expérience un peu différente a consisté à réaliser un outil auteur de workflow. Un workflow peut être défini comme un processus mettant en oeuvre un ensemble de tâches à réaliser. Chacune de ces tâches impliquant un ensemble de ressources humaines ou matérielles. Ces tâches ne sont pas indépendantes les unes des autres : le commencement d'une tâche T2 suppose par exemple la fin de la tâche T1. Il se peut aussi que pour des contraintes d'utilisation de ressources, les tâches T3 et T4 doivent se réaliser en séquence.

Les applications des workflows sont aujourd'hui nombreuses et variées :

- Organisation de l'emploi du temps d'une université, les ressources étant alors les enseignants, les élèves et les salles de cours, et les tâches étant les enseignements.
- Gestion du service des infirmières dans un hôpital.
- Gestion de la production d'une entreprise, les ressources seront les machines et le personnel, les tâches étant la production de biens.

Aujourd'hui, il existe quelques environnements d'édition de workflow : ActionWork, MicrosoftProject .

L'édition de worklfow dans ces environnements peut se décrire en trois étapes :

- Spécification de l'enchaînement temporel des différentes tâches.
- Allocation de ressources aux différentes tâches.
- Calcul de l'enchaînement des tâches de manière à prendre en compte la spécification de l'auteur et les contraintes sur les ressources. Par exemple, une ressource peut être partagée par deux tâches indépendantes temporellement, a priori, et de ce fait, ces deux tâches ne pourront s'exécuter en parallèle.

Au cours de l'expérience réalisée par Michael Tissot pendant son stage de maîtrise sous ma direction, nous nous sommes intéressés essentiellement à la spécification de l'enchaînement temporel des différentes tâches [Tissot00]. Aspect qui nous a semblé le plus proche de l'édition de documents multimédias, et qui n'était pas résolu de manière satisfaisante dans ces outils.

Nous allons, dans un premier temps, formaliser la définition d'un workflow (section 3.1), dans un deuxième temps nous dégagerons les principaux problèmes liés à l'édition de workflow (section 3.2) ceci afin de dégager les points communs avec les services offerts par Kaomi. Enfin dans la dernière partie de cette section (section 3.3) nous présenterons l'implémentation de cet éditeur ainsi que les apports et les limites de l'utilisation d'une boîte à outils telle que Kaomi dans ce contexte.

3.1 Modélisation d'un workflow

Un workflow peut se décomposer en tâches élémentaires et en tâches composites :

- Une tâche élémentaire peut être définie par :
  - Un intervalle pour définir l'instant de début de la tâche, par exemple la tâche doit commencer entre le 5 et le 10 juin.
  - Un intervalle pour définir l'instant de fin de la tâche, par exemple la tâche doit finir entre le 22 et le 24 juin.
  - Un intervalle de durée : la durée de la tâche est comprise entre 5 et 10 jours.
  - La contrôlabilité de la tâche, une tâche est contrôlable si avant de la commencer, on peut fixer sa durée, par exemple, la réalisation de la tâche T1 durera 6 jours, dans le cas contraire on dit que la tâche est incontrôlable. C'est le cas, par exemple, si sa réalisation dépend de conditions externes que l'on ne peut pas contrôler.
  - Un ensemble de ressources nécessaires à la réalisation de la tâche.
  - Un ensemble de ressources allouées à la réalisation de la tâche.
- Une tâche composite est définie par :
  - Un ensemble de tâches élémentaires ou composites.
  - Un ensemble de relations entre ces tâches : les tâches commencent en même temps, une tâche doit impérativement être finie à telle date, deux tâches doivent se dérouler en séquence.
  - Un intervalle pour définir l'instant de début de la tâche, par exemple la tâche doit commencer entre le 5 et le 10 juin.
  - Un intervalle pour définir l'instant de fin de la tâche, par exemple la tâche doit finir entre le 22 et le 24 juin.
  - Un intervalle de durée : la durée de la tâche est comprise entre 5 et 10 jours.
  - La contrôlabilité de la tâche, une tâche composite est contrôlable si toutes les tâches élémentaires qui la composent sont contrôlables.
  - Un ensemble de ressources nécessaires à la réalisation de la tâche.
  - Un ensemble de ressources allouées à cette tâche.

On peut remarquer que la définition d'un workflow ressemble à la définition d'un document multimédia, avec une priorité plus importante sur l'aspect gestion de ressources. Cependant, on note que l'on peut assimiler les besoins liés au partage de ressource aux aspects de qualité de services. Aspect que l'on retrouve dans le cadre des documents multimédias avec les besoins en débit du réseau, mémoire et CPU de la machine.

3.2 L'édition de workflow

Des différents systèmes auteur de workflow que nous avons étudiés [Tissot00], nous pouvons dégager un ensemble de caractéristiques communes :

- La représentation des informations temporelles :
  - Soit à base de vue temporelle proche d'un paradigme de temps absolu (Microsoft Project), c'est-à-dire que les objets sont visualisés sur un axe temporel, et leur longueur est proportionnelle à leur durée.
  - Soit à base d'une visualisation d'un graphe de dépendance entre les tâches (ActionWork).
- Les outils auteur de workflow se limitent souvent à la spécification du workflow et à sa résolution. Dans les outils étudiés, il n'y a aucune aide au diagnostique de la cohérence du workflow, les outils se contentent de dire, à la fin de la spécification que le workflow est cohérent ou non et de produire une solution dans le cas d'une spécification cohérente. De la même façon, il n'y a aucun moyen de paramétrer la résolution du workflow, de manière à, par exemple, minimiser la durée totale du workflow ou pour maximiser l'utilisation de certaines ressources.
- Les outils auteur de workflow offrent une édition directe très pauvre, c'est-à-dire que lorsque l'auteur manipule la représentation graphique du workflow et les contraintes entre les tâches ne sont pas maintenues graphiquement.

C'est pour essayer d'améliorer les différents points ci-dessus que nous avons construit un environnement de workflow au-dessus de Kaomi pour tirer parti des services d'édition/visualisation de la vue temporelle, et des services de vérification de cohérence incrémentaux. En effet, la présentation ci-dessus montre que la spécification temporelle de l'enchaînement des tâches pour un éditeur de workflow est très proche de l'édition d'un document multimédia. Les différences se situent au niveau de l'allocation de ressources, ainsi que dans la datation absolue de certains événements.

3.4 Implémentation de Workflow Editeur

La première étape a été de définir une DTD XML pour modéliser un workflow. A partir de cette étape, il a ensuite fallu définir une structure de données permettant de modéliser sous forme de *Documents* (format pivot de Kaomi) un workflow. Pour cela, nous avons dans un premier temps défini pour chaque tâche un média texte qui contenait le nom de la tâche et la liste des ressources nécessaires, ensuite nous lui avons défini des propriétés spatiales arbitraires, et enfin nous avons défini les propriétés temporelles entre objets en suivant la spécification du workflow. L'implémentation de cette tâche utilise la classe ElementMultimédia qui offrait tous les mécanismes pour stocker et éditer les informations nécessaires.

Au cours de la traduction de la spécification d'un workflow vers un ElementMultimédia, nous avons été limité par l'expressivité de notre modèle temporel, nous n'avons par exemple pas la possibilité de spécifier des dates absolues. En effet, dans Kaomi, nous utilisons des dates relatives au début du document, et nous n'avons pas le moyen, sans introduire un facteur d'échelle trop important pour les résolveurs, d'avoir un système de date absolue. De ce fait, la phase de vérification de cohérence et de formatage à chaque opération d'édition prendrait un temps trop important pour l'auteur.

Cette limitation n'est pas spécifique au domaine du workflow, dans des documents conçus pour des domaines d'application comme l'enseignement, on retrouve le même besoin. Par exemple, les étudiants ne peuvent accéder à un cours ou une correction d'exercice qu'à partir d'une date précise.

Une fois la définition d'une classe tâche réalisée, il a fallu définir le fichier de ressources nécessaires à l'édition de tâches et notamment à l'édition des ressources. Dans la palette résultante, l'auteur peut choisir les ressources utilisées et leur taux d'utilisation (Figure VI-12).

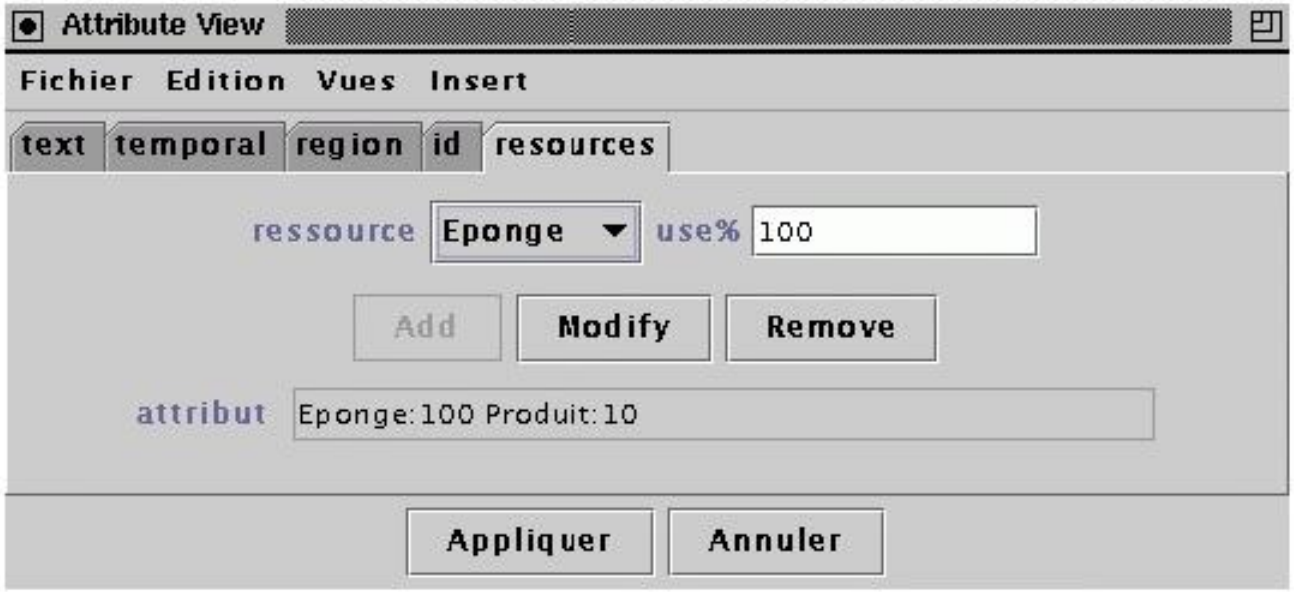

Figure VI-12 : Palette d'attributs de Workflow Editeur

Une fois la traduction du workflow dans une structure de données compatible avec la boîte à outil effectuée, nous avons pu utiliser tous les services de celle-ci : visualisation de l'enchaînement temporel, vérification de la cohérence, aide au formatage pour des tâches déterministes uniquement.

Dans l'exemple de la Figure VI-13, nous pouvons voir le résultat du placement temporel d'un workflow. Ce workflow organise la journée de Michael qui s'installe dans son nouvel appartement. Son après-midi est organisé en deux grandes parties : le repas et le nettoyage de l'appartement. Pour la réalisation de la deuxième tâche, Michael sera aidé d'un camarade.

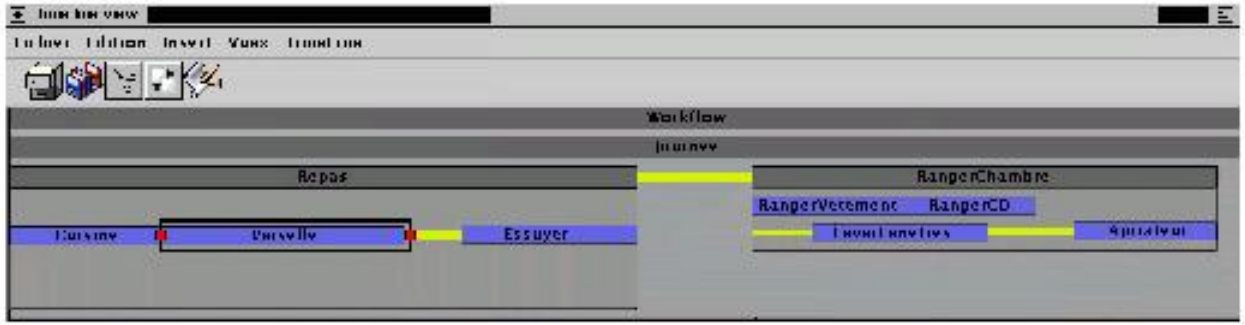

Figure VI-13 : Vue temporelle de Workflow Editeur

Par rapport à l'utilisation des résolveurs de contraintes, une première expérience a été menée pour intégrer la prise en compte des ressources lors du formatage. L'idée de cette expérience était d'intégrer les dépendances liées aux ressources, ainsi que le calcul de l'allocation de ressources dans le système de formatage. Les premiers travaux ont donné des résultats encourageants car ils permettent d'avoir un formatage qui prend en compte l'utilisation de

ressources, cependant ils n'ont pu être menés à terme à l'heure actuelle.

3.4 Bilan de Workflow Editeur

Les principaux apports de l'utilisation d'une boîte à outils comme Kaomi dans l'édition de workflow sont :

- Aide à la visualisation des informations temporelles.
- Edition directe par manipulation avec maintien des relations temporelles lors de l'édition, ce qui permet à l'auteur de mieux identifier la source de l'erreur.
- Vérification des incohérences et détection incrémentale des erreurs.
- Aide au formatage du workflow.
- Simulation de la dimension temporelle du workflow.

Parmi les limitations de la boîte à outils rencontrées, on peut citer la nécessité d'intégrer un formatage temporel avec gestion des ressources physiques ainsi que la nécessité d'intégrer des dates absolues. Un autre problème est la disparité dans les échelles de temps.

4 Les bases de données documentaires

Dans le cadre d'une coopération avec l'Aérospatiale ([Aérospatiale99]) nous avons utilisé Kaomi pour réaliser un prototype permettant d'éditer et de gérer un fond documentaire [Duluc00, Duluc00b]. Ce prototype a été réalisé par Frank Duluc dans le cadre d'une thèse Cifre en coopération avec les membres de l'équipe Opéra. Ce prototype permet de couvrir les différentes phases de la chaîne documentaire : l'édition, la gestion du fond, la production et la présentation des documents multimédias. Ce travail a été publié collectivement par Franc Duluc et les personnes d'Opéra [Duluc00].

L'idée était, dans un premier temps, de manipuler dans la base de documents un ensemble de fragments de documents multimédias, appelés *granules*, c'est-à-dire que l'on ne stocke pas seulement des médias dans cette base mais aussi des parties de documents. L'intérêt est de stocker un ensemble de médias agencés temporellement et spatialement. Dans un deuxième temps, à partir de ces fragments, l'auteur (et à plus long terme le système) génère une présentation qui doit être visualisée. L'apport d'une boîte à outils dans un tel processus est de fournir un mécanisme de création pour les différents fragments de la base documentaire, et, dans un deuxième temps, de permettre une édition et une visualisation des différents fragments pour en faire des documents à part entière. Ce prototype a été réalisé en utilisant les fonctionnalités Java pour faire l'interface avec les bases de données.

Ce prototype a permis de mettre en évidence la capacité d'utiliser une technologie multimédia dans les circuits de production documentaire. Cependant, il est nécessaire d'utiliser des processus automatiques de génération à partir des fragments de base comme pour les documents textuels. Pour réaliser cela, il est nécessaire d'utiliser des modèles de documents génériques qui prennent en compte le modèle temporel.

5 Evaluation quantitative de Kaomi et des environnements auteur

Comme j'ai pu le dire au début du chapitre IV, Kaomi est le résultat d'un travail que j'ai initié et qui a évolué grâce à la participation de nombreuses personnes. Afin de clarifier les contributions de chacun j'ai reporté, dans la Figure VI-14, une évaluation du nombre de lignes de code de Kaomi, des

différents environnements auteur construits à l'aide de Kaomi ainsi que de la contribution de chacun. On peut voir que le nombre de lignes de code spécifique est mineur par rapport à la boîte à outils et ce code consiste essentiellement à la définition d'un analyseur lexical spécifique. De plus, dans cette figure, j'ai reporté ma participation directe et indirecte (via l'encadrement de stagiaires) à cette boîte à outils et aux différents environnements auteur.

	Kaomi	MadeusEd.	SmilEd.	MHMLEd.	WorkflowEd.
Nombre de lignes de code	110000	5000	6000	3000	4000
% développé personnellement	20%	20%	35%	35%	30%
% développé par des stagiaires sous ma responsabilité	40%	30%	60%	0%	70%
Autres : doctorants, ingénieur	40%	50%	5%	65%	0%

Figure VI-14 : Code de Kaomi et des environnements auteur

Au cours de ces différentes expériences, la boîte à outils s'est enrichie et a été rendue de plus en plus extensible et générique. Cette extensibilité et cette généricité constituent sans aucun doute un point fort qui laisse présager de bonnes possibilités d'évolution dans le futur. Nous reviendrons sur ce point dans la conclusion.

6. Bilan des expérimentations

Nous venons de voir, au cours de ce chapitre, la présentation des expérimentations menées avec la boîte à outils Kaomi. Je vais en faire un bilan en trois points :

- *Modèle d'édition proposé.* Nous avons vu dans les chapitres III et IV qu'un environnement auteur devait, pour faciliter la tâche de l'auteur, lui fournir un formalisme d'édition de plus haut niveau que celui du format de présentation, tout en lui permettant d'exprimer tout ce que le format de présentation lui permettait. Au cours des expériences avec SMIL-Editeur et MHML-Editeur, nous avons vu comment se passe l'édition en combinant ces deux formalismes. Nous avons vu de plus, que les services de l'environnement auteur étaient dépendants du formalisme utilisé par l'auteur. Les services d'aide sont optimaux pour les parties spécifiées (ou compatibles) avec le formalisme de l'environnement auteur, et ils offrent un service minimal dans le cadre d'une édition spécifique. La boîte à outils permet au développeur de l'environnement auteur de spécialiser ces services pour le

langage utilisé.

- *Proposition d'environnement ayant plusieurs vues synchronisées*. Dans le chapitre III nous avons proposé un environnement permettant à l'auteur d'éditer son document dans plusieurs vues, utilisant pour chaque opération la plus adaptée à son besoin. Les différentes expérimentations menées ont permis de vérifier que ce concept de multivues est parfaitement adapté aux différents formalismes de spécification de documents multimédias et que le mécanisme d'extension des vues est un moyen d'adaptation aux formats de présentation (MHML).
- *Bilan de l'utilisation de Kaomi comme base de conception de systèmes auteur*. Trois systèmes auteur de documents multimédias ont été réalisés au-dessus de Kaomi. Nous avons vu dans la section précédente que le code spécifique nécessaire pour chacun de ces environnements auteur était mineur par rapport au code de la boîte à outils. Cela montre la pertinence de l'utilisation de Kaomi dans ce contexte.

Le bilan des expérimentations faites avec Kaomi ne nous permet de relever que très peu de points négatifs. Cela est principalement lié au fait que les manques rencontrés lors de la création des différents environnements auteur ont été directement implémentés dans la boîte à outils. C'est le cas par exemple du module de simulation réalisé initialement pour MHML-Editeur. La validation de cet environnement est plus poussée que les autres du fait que nous avons eu un retour de la part d'utilisateurs.

On peut noter cependant l'impossibilité de spécifier des dates absolues, ainsi que la réalisation de services de diagnostique des incohérences qui est quasi-inexistant.

7. Travaux futurs

Malgré les différents aspects positifs, Kaomi reste une boîte à outils qui doit s'étendre et s'enrichir. Les différentes extensions que l'on peut envisager sont classées ci-dessous en cinq catégories :

Médias : les deux premières extensions peuvent être considérées comme un travail d'implémentation et ne poseront aucun problème d'intégration dans la boîte à outils. Les deux dernières extensions nécessiteront par contre un travail de réflexion plus important pour les intégrer complètement au processus d'édition.

- *Intégrer l'édition des médias* : il est contraignant pour un auteur d'utiliser des outils différents pour la création des différents types de médias. L'environnement auteur doit fournir un ensemble de primitives élémentaires pour éditer les différents médias et leur affecter des transformations ou des effets de style élémentaires (Director, PowerPoint).
- *Intégrer une édition plus fine des médias* en identifiant des sous-parties à l'intérieur de ceux-ci, par exemple, par la définition de scènes dans une vidéo. Cela permettra une synchronisation de ces éléments avec d'autres médias du document. La principale difficulté, outre la définition de sous-parties dans les médias est le facteur d'échelle que cela introduit dans la boîte à outils. Un travail sur l'édition de vidéos structurées est actuellement réalisé par Tien Tran Thuong, étudiant en thèse. L'objectif de ce travail est de permettre à l'auteur de définir une structure pour une vidéo et de l'utiliser pour synchroniser des scènes de la vidéo avec des médias du document [Roisin00]. Ce travail se concrétise actuellement par l'introduction d'une vue vidéo structurée dans Kaomi.
- *Accéder à des bases de données*: le couplage d'un environnement auteur avec une base de données pour accéder aux médias permettra dans un premier temps d'étendre les capacités de recherche et d'insertion des médias lors de l'édition. Dans un deuxième temps, les documents pourront contenir des requêtes vers des bases de données intégrant ainsi dynamiquement ces médias tout en maintenant un ensemble de propriétés spatiales et temporelles au document. On retrouve là la notion de granule de Franck Duluc[Duluc00b]. Cependant, cela posera un problème de formatage

dynamique du document pour prendre en compte cette intégration de manière optimale.

- *Implémenter un ensemble plus large de médiateurs* : les bibliothèques libres utilisées permettent de jouer l'essentiel des standards de médias. Cependant, dans le cadre d'un environnement auteur ayant un objectif de diffusion large, il est nécessaire de permettre l'ouverture vers de nouveaux standards de médias (SVG) ainsi que l'intégration d'un ensemble de médias dans des formats propriétaires (Word, Excel, Director). Les difficultés de cette intégration sont liées à la granularité de l'insertion de tels médias et donc à la synchronisation fine d'une partie de ces médias avec d'autres éléments du document.
- *Intégrer des comportements indéterministes :* l'intégration de médias indéterministes nécessite une adaptation de la notion de formatage et de cohérence [Layaïda97]. Vu la complexité théorique du traitement de l'indéterminisme, une première approche (traitant partiellement le problème) serait d'introduire une notion de cohérence locale. Une première expérience d'édition /présentation a été menée dans le cadre de MHML-Editeur.

Hypertexte:

- *Intégration du processus de lecture dans le document*. Cette extension permettra de ne plus considérer un document comme une entité isolée, mais comme faisant partie d'un tout comme, par exemple, d'un processus d'apprentissage. On retrouve là les idées rencontrées dans le workflow. L'environnement de lecture doit être capable de gérer le flux de lecture, c'est-à-dire l'enchaînement de parcours entre les différents documents faisant partie de ce processus. Cela peut se réaliser par l'utilisation de liens conditionnels.
- *Visualisation*: on peut diviser l'extension des services de visualisation de la structure hypertexte en deux étapes :
  - *Intégrer un service minimum de visualisation de la structure hypertexte* : il est nécessaire de permettre à l'auteur de visualiser cette structure et si possible, de naviguer facilement à l'intérieur de celle-ci, en offrant par exemple une visualisation graphique des liens hypertexte du document.
  - *Intégrer un service évolué de visualisation et d'édition de la structure hypertexte*. Les principales difficultés seront de permettre à l'auteur de percevoir cette structure pour de gros documents. Les représentations actuelles des structures hypertexte de document sont limitées par le facteur d'échelle (voir Chapitre II).

Document: au cours de cette thèse nous avons considéré les médias comme des éléments de base, nous avons restreint le pouvoir d'expression de l'auteur pour faciliter l'édition. Il est nécessaire d'étendre ces deux limites pour offrir un environnement auteur complet.

Permettre l'accès à un langage de programmation : il existe toujours des situations où, quel que soit l'environnement auteur utilisé, certains auteurs spécialisés auront besoin d'accéder ponctuellement à la puissance d'un langage de programmation (lancement de programme externe, tests sur la configuration de l'environnement de lecture du document). Il est donc nécessaire de fournir de tels points d'entrée. Cette intégration nécessitera la vérification de l'intégrité d'une telle édition par rapport au reste de la spécification du document.

Modèle de document / feuille de style : la notion de modèle de document utilisée dans l'édition textuelle n'est pas encore intégrée à l'édition de document multimédia. Cette intégration soulève deux problèmes :

- *Intégrer la notion de modèle de documents* aux documents multimédias en s'inspirant des travaux réalisés pour les documents textuels ([Quint87], [Furuta88]). Cela permettra de définir des classes de documents et de faciliter l'édition du document par l'utilisation de feuilles de style, en dégageant ainsi l'auteur de la définition du positionnement spatial et temporel de ces objets.
- *Définir des modèles de documents* :l'intégration de modèles de documents permettra de faciliter l'édition, cependant l'écriture de ces modèles de documents ne se fait pas encore de manière conviviale. Il faudra fournir des moyens interactifs pour définir de tels modèles que ce soit pour les sources de données ou pour les feuilles de styles. Un premier travail dans ce sens est en cours au dessus de Kaomi : il est réalisé par Lionel Villard,

étudiant en thèse [Villard00] et vise principalement a intégrer les transformations des données incrémentales pour l'intégrer au processus d'édition.

Souplesse de présentation: nous avons vu que la définition relative du document permet de simplifier la tâche d'édition. Cette souplesse définie dans le document peut être aussi utilisée lors de la présentation du document.

- *Formatage dynamique (en cours de présentation) du document :*l'intégration dynamique de médias ou l'adaptation du formatage temporel et spatial à l'évolution des conditions de présentation nécessitent la mise en place de formateurs prenant en compte cette composante.
- *Intégration d'un service de gestion de la qualité de service* pour la vue de présentation : la manipulation de médias dynamique nécessitant de nombreuses ressources physiques (mémoire, bande passante), une politique de pré-chargement et d'accès aux médias efficace est nécessaire. L'utilisation d'un graphe temporel dans la vue de présentation permet de connaître statiquement le cours de l'exécution et permet ainsi au système, à un instant t, d'anticiper le chargement des médias qui seront utilisés dans le futur. Un travail est mené par Maximilien Laforge pour améliorer le service proposé par la vue de présentation [Laforge00]. La difficulté sera de généraliser ce travail aux documents imprédictifs.
- *Alternative sur les médias :*une définition relative du document devrait faciliter entre autre le changement des médias dynamiquement. De ce fait, l'écriture de documents multilingues pourrait se réaliser en associant aux objets multimédias de Kaomi un ensemble de médias (propre à chacune des langues), et le système pourrait ainsi, au moment de la présentation choisir l'ensemble de médias adapté tout en conservant la spécification du comportement temporel et spatial entre les objets multimédias.

Fonctions utilisateur avancées :enfin, il est aussi nécessaire d'étendre certaines fonctions d'édition pour notamment prendre en compte l'édition coopérative de documents.

- *Gestion d'utilisateurs sur le document et de leurs droits*: les documents peuvent s'adapter en fonction des lecteurs, cette adaptation dépend en partie des droits que le lecteur possède au moment de l'accès au document. Par exemple, dans le cas d'un document représentant un cours, on peut imaginer qu'un étudiant n'ait pas le droit de regarder la correction d'un exercice avant une certaine date.
- *Gestion de l'édition coopérative de documents multimédias :*de nombreux travaux portent sur l'édition coopérative de documents textuels ou hypertexte ([Decouchant96], [Byzance00], [Séraphine01]). L'édition de documents multimédias met en oeuvre de nombreuses personnes travaillant à différentes étapes du processus de conception. Il est donc naturel de souhaiter un environnement qui leur permette de collaborer efficacement entre elles.

8. Conclusion

Dans ce chapitre nous avons d'une part décrit les différentes réalisations effectuées avec Kaomi, et d'autre part, à partir d'un bilan de ces expérimentations, porté à la fois sur le formalisme d'édition proposé et sur l'utilisation de la boîte à outils, nous avons dégagé un ensemble de pistes pour étendre les capacité d'une telle boîte à outils.