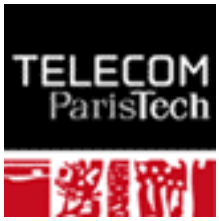
Scalabilité de scène multimédia

Benoît Pellan



Thèse

présentée pour obtenir le grade de Docteur
de Télécom ParisTech

Spécialité : **Informatique et Réseaux**

Benoît PELLAN

Scalabilité de scène multimédia

Soutenue le 7 octobre 2010 devant le jury composé de

Cécile Roisin	Président
Luiz Fernando Gomes Soares	Rapporteurs
Cécile Roisin	
Ansgar Scherp	Examinateur
Marc Brelot	Invité
Cyril Concolato	Directeurs de thèse
Béatrice Pesquet-Popescu	

Table des matières

Remerciements

Je tiens tout d'abord à remercier l'équipe Multimédia du département TSI de Télécom ParisTech que le hasard de la recherche (de mon premier emploi) m'a fait rejoindre au début de l'année 2004. C'est véritablement au sein de cette dynamique équipe que j'ai été initié au goût de l'innovation. Merci beaucoup Cyril, Jean, Jean(s)-Claude(s) et Béatrice pour la passion que vous savez transmettre tout aussi bien dans le cadre de projets scientifiques qu'à l'occasion de la pause café.

J'ai également une pensée particulière pour mes collègues 'précurseurs' qui m'ont fait la démonstration irréfutable, alors que nous partagions le même bureau, que l'on pouvait faire face sereinement à la mission du doctorant. Merci Philippe et Mariam d'avoir accompagné amicalement mes premiers pas dans le domaine de l'adaptation multimédia.

Si les conditions étaient effectivement favorables à l'été 2006 pour débuter cette aventure, il est certain que la mise en oeuvre de ce projet de thèse est véritablement le résultat de la détermination des dirigeants de RTL. Merci Charles-Emmanuel, Alexis et Frédéric de m'avoir fait confiance dans ce projet enthousiasmant et complexe qu'est la radio numérique. Je garde évidemment un souvenir inoubliable de notre équipe du 'RTLab'. Merci Jonathan et Pierre pour votre professionnalisme dans la mise au point des solutions et des démonstrations qui ont contribué au succès final de mes travaux. Merci Marc pour tes relectures toujours très pertinentes et créatives, c'est toujours un plaisir d'échanger avec une personne aussi 'visionnaire'. J'en profite également pour saluer ici tous mes anciens collègues de RTL avec qui j'ai eu plaisir à travailler.

La scalabilité de scène se déployant selon plusieurs axes, les dimensions académique et industrielle de mes travaux de thèse n'auraient pu s'épanouir sans une vie familiale heureuse. Je remercie tout naturellement ma famille pour son soutien attentionné et constant depuis la Bretagne et évidemment Emilie, ma femme, pour sa patience pendant la longue phase de rédaction de ce mémoire.

Je tiens aussi à souligner que les réflexions qui nourrissent ce mémoire sont issues de nombreuses rencontres au fil des années. Je ne peux malheureusement mentionner toutes ces personnes avec qui j'ai eu plaisir à travailler, notamment dans le cadre des projets DANAE et RADIO+, sans courir le risque d'omettre quelqu'un. Par ailleurs, je tiens également à noter l'effet finalement bénéfique des obstacles que certains ont pu mettre sur mon chemin. Là encore, il ne serait pas raisonnable de faire des remerciements personnalisés.

Je remercie également toutes les personnes qui m'ont aidé dans la dernière ligne droite. Merci Berthele, Brahim, Jonathan, Lina, Romain, Stéphane et Stanislas pour votre soutien efficace. Merci Olivier pour avoir si consciencieusement apporté une touche canadienne à l'anglais des extraits que je t'ai transmis. Merci Monique pour ta présence appréciée lors de ma soutenance.

Enfin, puisqu'il faut conclure ces remerciements, je vous soumets trois axes de réflexion sur le système d'enseignement par la recherche de Telecom ParisTech dont l'évolution m'interpelle:

- Au cours de ces sept années, j'ai pu constater certaines régressions qui dévalorisent, selon moi, la capacité d'innovation des doctorants. En particulier, je suis convaincu que je n'aurais pas pu mener à bien cette thèse dans les conditions désormais en vigueur.
- Il me plait parfois de penser que cette thèse a finalement commencer à germer dès 2002 lors de mon stage d'ingénieur sur le portage du codage hiérarchique JPEG 2000 sur les chipsets de Texas Instruments. Quelle est aujourd'hui la place pour la maturation de l'innovation ?
- A l'occasion de mon mariage, on a pu faire dire avec ironie à Philippe Bouvard "Entre la thèse et les enfants, il faut choisir". Le choix a été une évidence: cette thèse aura nécessité 4 années et Luce a fêté ses 15 mois le lendemain de ma soutenance. C'est en pensant à elle que j'aime à terminer ce mémoire.

Résumé long en français

Chapitre 1 : Introduction

Enjeux & besoins

La diffusion[1] de données multimédia associées aux services traditionnels de radio ou de télévision du paysage audiovisuel français constitue une évolution favorisant une interaction avec les auditeurs ou des téléspectateurs. Cette évolution est facilitée par la transition technologique qui s'opère avec le passage de l'analogique vers le numérique de la diffusion de ces services au travers de standards comme DVB-T [42], DVB-H [44] ou T-DMB [39]. Elle est également provoquée par le développement rapide de services complémentaires transmis par internet, comme les podcasts des émissions de radio ou encore les programmes disponibles en télévision de rattrapage. Cette évolution (ou révolution) annoncée des services de radio ou de télévision prend parfois le nom de "radio augmentée" ou de "télévision intelligente" (Smart TV). Les données multimédia associées aux stations de radio ou aux chaînes de télévision se doivent naturellement d'être à la hauteur des attentes des utilisateurs en termes de qualité des contenus mis à disposition mais elles doivent également proposer une interface attrayante et ergonomique, conforme aux exigences de l'utilisateur moderne. En particulier, cette expérience multimédia se doit d'être au moins équivalente à celle que l'on peut observer au travers des sites internet des stations de radio ou des chaînes de télévision, un des objectifs pour les éditeurs historiques de radio et de télévision étant de renforcer la pertinence de la plateforme hertzienne et de conquérir la prochaine génération d'utilisateurs.

La diffusion audiovisuelle par voie hertzienne est un domaine exigeant pour les services multimédia. Cet environnement présente plusieurs contraintes qui amplifient les difficultés que l'on peut rencontrer en téléchargeant un document multimédia ou bien en transmettant un service en streaming par internet. Par exemple, la bande passante dédiée à un service de radio numérique correspond à une part de la rare ressource hertzienne[2]: elle est donc relativement limitée et il n'est pas envisageable, dans les conditions actuelles, de faire l'hypothèse systématique d'une voie de communication depuis le récepteur de radio numérique vers un serveur distant (une voie de retour par internet, par exemple). En outre, le canal de communication utilisé pour la diffusion des services multimédia peut être perturbé et engendrer des pertes irrécupérables de données. Enfin, la diffusion impose, par nature, que le même service multimédia soit diffusé à tous sur une zone géographique donnée alors que le parc des récepteurs est souvent très hétérogène, les récepteurs les plus sommaires disposant uniquement des capacités nécessaires à la restitution sommaire du service.

Si le souhait premier d'un éditeur de service radiodiffusé est bien de rester accessible par son public quelque soit le récepteur utilisé, ces mêmes éditeurs cherchent également à se démarquer les uns des autres en proposant des fonctionnalités innovantes qui tirent parti des capacités techniques les plus avancées des récepteurs. Par conséquent, il est important que des services multimédia riches puissent être diffusés mais il est également impératif qu'ils ne provoquent pas d'incompatibilité avec le parc des récepteurs déployés. De la même façon, la multidiffusion de données multimédia visant différents récepteurs spécifiques constitue une valeur ajoutée pour un service. Cependant, il n'est pas acceptable que de telles déclinaisons d'un même service induisent des délais de transmission excessifs pour l'ensemble du service, notamment lorsque la bande passante s'avère trop limitée. De fait, le maintien de la qualité globale des services multimédia en diffusion au cours du temps nécessite un soin permanent de

[1] La diffusion est définie tout au long de ce mémoire comme étant une transmission unidirectionnelle de données, par voie hertzienne dans le cadre de nos expérimentations, à destination d'un grand nombre d'individus sans qu'il ne puisse être nécessairement fait l'hypothèse d'une voie de retour complémentaire.
[2] Le débit alloué aux données multimédia d'un service de radio représente typiquement un débit compris entre 8 et 32 kbps.

la part des éditeurs, y compris dans les différents environnements d'usage que l'utilisateur peut rencontrer, de façon à toujours préserver un lien de confiance et de proximité avec chaque téléspectateur. En pratique, le souhait d'un éditeur étant toujours de rendre son service multimédia accessible au plus grand nombre, une stratégie simple pour la publication de contenu consiste à minimiser les prises des risques quitte à réduire la qualité du service pour tous. Cette réduction de qualité peut prendre la forme de restrictions applicables seulement dans certains cas d'usage répertoriés, certains environnements pouvant parfois être simplement ignorés. Par exemple, il est courant qu'un site internet soit conçu en évitant les fonctions mal intégrées par certains navigateurs internet, voire même parfois, que celui-ci soit inaccessible à partir de certains navigateurs jugés peu utilisés. Cependant, ces solutions de contournement ne sont pas satisfaisantes en diffusion car il n'est pas acceptable qu'un service multimédia soit inaccessible depuis un récepteur compatible ou qu'un service multimédia souffre d'importantes limitations sur des équipements haut de gamme, alors que ces limitations ne s'appliquent pas pour des services similaires transmis par internet[3]. Par conséquent, il existe un besoin de techniques performantes d'adaptation qui permettent la diffusion de services multimédia robustes à tous les utilisateurs et qui facilitent l'enrichissement multimédia de leur présentation dès que les conditions d'usage le permettent.

Formulation du problème & principes de la solution proposée

La finalité de nos travaux consiste à fournir les outils techniques nécessaires à la diffusion de services multimédia adaptables aux conditions d'usage qu'un éditeur de service souhaite prendre en charge avec un soin particulier. Ainsi, l'objectif de nos travaux peut se formuler sous cette forme: il s'agit de spécifier *un processus de transformation générique, autonome et aux surcoûts réduits pour l'adaptation maîtrisée de services multimédia enrichis*. En effet, le marché horizontal, qu'est le marché des récepteurs de radio ou de télévision, nécessite un processus d'adaptation qui, en l'état actuel, ne peut faire l'hypothèse systématique d'une voie retour (*transformation autonome*). De façon à réduire le coût des récepteurs, un processus d'adaptation se doit de reposer sur des transformations standardisées (ou qui sont susceptibles d'être standardisées) de façon à pouvoir se généraliser à l'ensemble des gammes de produit (*transformation générique*). La diffusion de données multimédia associées aux programmes en direct implique un processus d'adaptation dynamique et continu (*service multimédia*) dont la qualité doit pouvoir être validée selon les règles éditoriales et la charte graphique définies par l'éditeur (*transformation maîtrisée*). Enfin, la captation de l'attention de l'utilisateur requiert une présentation attrayante et travaillée (*contenu multimédia riche*) qui doit pouvoir être adaptée par un processus de transformation interférant peu sur l'environnement d'exécution du service (*transformation aux surcoûts réduits*).

Le principe sous-jacent notre approche provient de la représentation scalable des codecs audiovisuels comme MPEG-4 SVC [104] ou JPEG2000 [85], aussi appelés codages hiérarchiques. Une telle structuration de la représentation d'un média permet la mise en oeuvre d'un processus d'adaptation générique et dynamique, fondé sur le filtrage de sous-ensembles du flux binaire. L'avantage essentiel de ce procédé de filtrage adaptatif est l'efficacité de son exécution en comparaison avec les techniques de transcodage. En pratique, toutes les options d'adaptation sont configurées pendant la phase d'encodage du média et classées dans des couches progressives dites « de scalabilité ». Ces couches de scalabilité correspondent donc à des enrichissements du média comme, par exemple, l'amélioration du rapport signal sur bruit d'une vidéo ou l'augmentation de la résolution d'une image. La représentation scalable des codecs audiovisuels présente donc des propriétés d'adaptation qui correspondent en partie aux exigences de la diffusion audiovisuelle de services multimédia. Ainsi, bien qu'une scène, chorégraphe organisant les différents média dans une présentation visuelle et interactive au cours du temps, n'ait pas les mêmes propriétés qu'une séquence vidéo ou qu'une image, nous défendons l'idée que le concept de scalabilité des média peut être transposé aux scènes multimédia, définissant ainsi la *scalabilité de scène multimédia.* Le principe défendu dans ce mémoire est donc que la présentation d'un service multimédia (ou d'un document multimédia statique) peut être modélisé en trois axes de scalabilité en répartissant

[3] Les services multimédia à la demande peuvent bénéficier d'une phase de négociation en amont de la transmission des contenus; or ceci est impossible dans le cas de la diffusion.

l'ensemble des caractéristiques d'une présentation en couches de scalabilité spatiales, temporelles et interactives. Grâce à la flexibilité des différents formats standardisés de description de scène multimédia comme SVG [118], MPEG-4 BIFS [59], SMIL [119], HTML [120], NCL [105] ou Flash, nous proposons d'organiser les propriétés des différentes scènes alternatives d'un même service (ou d'un document) multimédia en couches de scalabilité. Par conséquent, la solution que nous proposons permet la génération de services (ou document) multimédia adaptables où l'adaptation est conçue comme l'enrichissement d'une présentation existante et dont l'affichage progressif est possible. Ainsi, différents paramètres d'adaptation peuvent éventuellement être associés à ces couches de scalabilité : la dimension de l'écran, comme illustré dans l'exemple ci-dessous, ou encore le niveau d'intérêt de l'utilisateur pour le contenu, la durée de la présentation ou encore la capacité du processeur du récepteur à prendre en charge des animations complexes.

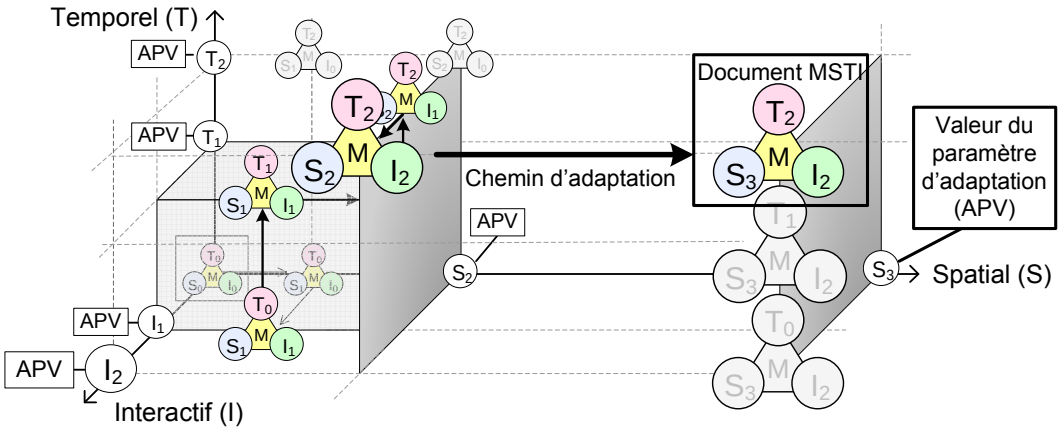
Exemple de scalabilité de scène et d'une adaptation à la dimension de l'écran.

Contributions

Le cheminement vers la modélisation de la scalabilité de scène multimédia nous a amené à une contribution que nous appelons le modèle *Scalable MSTI*. L'acronyme MSTI signifie *Média*, *Spatial*, *Temporel* et *Interactif*. Dans notre approche, les termes 'scalable' et 'scalabilité' sont directement empruntés de l'anglais 'scalable' et 'scalability' comme définis par H. Schwarz dans le contexte du standard MPEG-4 SVC [104]. Si l'on se réfère à l'office québécois de la langue française[4], la scalabilité fait référence à la notion d'extensibilité dans le domaine informatique et au codage hiérarchique dans le domaine des télécommunications. En effet, l'extensibilité y est définie comme 'l'aptitude d'un service à augmenter ou à diminuer son niveau de performance et ses coûts pour répondre aux changements dans la capacité de production ou dans la demande'. D'autre part, le codage hiérarchique y est défini comme 'une technique de codage du signal numérique, qui consiste à répartir les informations à transmettre dans des sous-ensembles hiérarchisés, de telle sorte qu'elles puissent être utilisées par ordre d'importance, au moment de reconstituer les sons ou les images.' […] 'Un sous-ensemble de base contenant les informations essentielles et des sous-ensembles secondaires sont définis. A la réception, le décodeur utilise d'abord les données du sous-ensemble de base, puis, s'il le peut, les données des sous-ensembles secondaires. La transmission s'adapte ainsi au débit disponible ou aux capacités du décodeur.' Ainsi, la scalabilité de scène multimédia, telle qu'envisagée dans notre contribution, intègre ces deux acceptions en proposant à l'utilisateur différents niveaux de présentation aux 'performances' croissantes en 'hiérarchisant' la scène en 'sous-ensembles' de manière à faciliter l'adaptation de la présentation à l'environnement d'usage lors de sa diffusion.

Si la scalabilité de scène multimédia peut être utilisée pour adapter un service multimédia à l'environnement d'usage en supprimant certains aspects de la présentation de la même manière que le codage hiérarchique d'un média audiovisuel le permet, l'élément innovant du modèle *Scalable MSTI* est la notion de couche de scalabilité pour une scène multimédia. Nous les appelons *couches STI* et celles-ci ont la particularité de traiter l'adaptation de plusieurs composants d'une scène en tant que groupe et non en tant que composants indépendants d'une même scène. Dans les paragraphes suivants sont décrites les deux principales contributions de nos travaux que sont le graphe d'adaptation et les mises à jour d'adaptation.

[4] http://www.granddictionnaire.com



Le graphe d'adaptation d'une scène *Scalable MSTI*.

Contribution – Le graphe d'adaptation

Dans ce mémoire, nous proposons un processus de prise de décision représenté à partir d'un graphe à trois dimensions structurant les capacités d'adaptation d'une scène multimédia scalable. Dans ce *graphe d'adaptation* illustré ci-dessus, chaque alternative de scène est positionnée sur des axes de scalabilité, appelé *axes de scalabilité STI* (axes *Spatiaux*, *Temporels* et *Interactifs*), selon des paramètres d'adaptation sélectionnés pendant la phase de création. Par conséquent, chaque alternative de la scène, représentée par un triplet (S_i, T_j, I_k), peut être associée aux valeurs de paramètres d'adaptation (APV) qui guident l'algorithme de prise de décision. Puisque des contraintes de progressivité s'appliquent aux propriétés de scène contenues dans les couches STI, le processus de prise de décision en vue de l'adaptation d'une présentation peut être représenté sous la forme de la sélection d'un chemin d'adaptation menant à une scène appropriée (et privilégiée par l'auteur) parmi l'ensemble des scènes possibles. En complément de ce parcours adaptatif, a priori sans retour possible autre que l'origine du graphe d'adaptation, des couches STI dites « de repli » peuvent être introduites. Ces couches de scalabilité de repli permettent, par exemple, qu'un lecteur multimédia soit en mesure d'identifier une configuration optimale 'en aveugle', c'est-à-dire sans nécessairement faire appel à des valeurs de paramètres d'adaptation explicites, par l'application progressive de couches STI jusqu'à ce que celles-ci excèdent les capacités du lecteur multimédia. En outre, ces couches STI de repli constituent de véritables points d'entrée dans le graphe d'adaptation, complémentaires à l'origine du graphe. De cette façon, un lecteur multimédia est aussi en mesure de traiter les modifications apportées à un service multimédia au cours du temps sans avoir systématiquement à retraiter l'ensemble de la scène courante.

Les avantages d'un graphe d'adaptation sont multiples. En particulier, cette représentation par graphe suggère un procédé d'édition guidé pour la génération de présentations adaptables à partir d'un jeu de scènes multimédia à renseigner. En diffusion, le graphe d'adaptation facilite la mise en œuvre d'optimisations pour la transmission progressive de scènes multimédia par la définition de priorités entre les différents aspects d'une présentation. En réception, l'adaptation dynamique d'une scène multimédia aux contraintes fluctuantes de l'environnement d'usage peut être gérée au travers d'un algorithme déterministe qui ne nécessite pas l'interprétation des propriétés de scène encapsulées dans les couches STI. Enfin, l'organisation hiérarchique des propriétés des différentes scènes alternatives d'une présentation adaptable ouvre un large champ de domaines d'application à partir du même moteur d'adaptation par la mise en correspondance des caractéristiques fondamentales d'une présentation (propriétés spatiales, temporelles et interactives d'un scène) avec des paramètres d'adaptation choisis en fonction des scénarios visés.

Contribution – Les mises à jour d'adaptation

Dans ce mémoire, nous proposons également une nouvelle forme de transformation de scène définie à partir de mises à jour qui modifient les caractéristiques d'une présentation dans la perspective d'une adaptation. Ces *mises à jour d'adaptation* sont des transformations similaires aux classiques mises à jour de scène définies dans Flash, MPEG-4 BIFS [59] ou MPEG-4 LASeR [62] et qui sont utilisées pour modifier la scène d'un service multimédia au cours du temps. En revanche, ces mises à jour nécessitent une validation supplémentaire à partir de l'environnement d'usage avant de pouvoir être appliquées à la scène. En d'autres termes, les mises à jour d'adaptation peuvent être filtrées lorsqu'elles ne sont pas compatibles avec l'environnement d'usage du document ou du service multimédia. Il est important de noter ici que la portée de ces mises à jour d'adaptation ne comporte pas de limite au sein de la scène autres que celles des formats utilisés pour les exprimer. Ainsi, des modifications portant sur de nombreux composants d'une scène peuvent être traités de façon unitaire en les encapsulant sous la forme de mises à jour d'adaptation au sein d'une même couche STI.

Les mises à jour d'adaptation se différencient essentiellement des autres types de transformation adaptative par le surcoût faible qu'elles nécessitent en diffusion en termes de puissance de calcul, de besoin en mémoire et de consommation en bande passante. Elles permettent également des transformations de scène aux niveaux les plus élémentaires des formats de scène et ne bride donc pas les capacités d'expression des langages de scène multimédia. Enfin, les mises à jour d'adaptation présentent la particularité d'avoir des coûts d'adaptation qui progressent avec la richesse du service multimédia. Ainsi, la scène multimédia conçue pour les environnements d'usage les plus exigeants (le service ou document dit « de base ») bénéficie de coûts d'adaptation très faibles, et donc de conditions d'accès privilégiées, alors que les présentations destinées à des environnements plus favorables pourront bénéficier d'enrichissements multimédia par la prise en charge de coûts d'adaptation plus élevés.

Contexte industriel des travaux

Les travaux présentés dans ce mémoire ont été menés dans le contexte industriel de RTL en tant qu'employé à plein temps entre octobre 2006 et janvier 2010 (dans le cadre CIFRE d'une Convention Industrielle de Formation par la REcherche). RTL, radio la plus écoutée en France, appartient à RTL Group qui est le leader de l'industrie du divertissement à l'échelle européenne avec 43 chaînes de télévision, 31 stations de radio dans dix pays différents et une production audiovisuelle internationale. Dans le cadre de mes activités de recherche, les quatre stations de radio du groupe RTL en France, à savoir RTL, RTL2, Fun Radio and RTL L'Equipe, ont été le terrain de jeu de mes expérimentations, visualisables tout au long de ce mémoire, mais elles ont surtout défini les fortes exigences de production multimédia qui ont guidé mes travaux.

Le rythme de mes travaux a été influencé par le calendrier légal du lancement de la radio numérique en France, qui a été défini progressivement par la loi du 30 septembre 1986 relative à la liberté de communication, au travers de ses évolutions successives en 2004[5], 2007[6] et 2009[7]. En mars 2008, le Conseil Supérieur de l'Audiovisuel (CSA) a lancé le premier appel à candidatures pour la diffusion de services de radio numérique sur les zones de Paris, Marseille et Nice-Cannes en T-DMB dans les bandes hertzienne III et L[8]. Tout au long de cette période, j'ai notamment contribué au nom du groupe RTL aux efforts de standardisation, au travers du groupe 'Systems' de l'organisation MPEG et du comité technique du WorldDMB, dans le but d'améliorer l'efficacité et l'interopérabilité des services de radio en T-DMB. Ces contributions, portées techniquement par Telecom ParisTech, ont abouti à la mise à jour du standard T-DMB publié par l'ETSI en avril 2009 [38] et par la définition d'un nouveau profil du standard MPEG-4 BIFS publié par l'ISO/IEC en juillet 2010 [60].

[5] Loi n° 2004-669 du 9 juillet 2004 relative aux communications électroniques et aux services de communication audiovisuelle
[6] Loi n°2007-309 du 5 mars 2007 relative à la modernisation de la diffusion audiovisuelle et à la télévision du futur
[7] Loi n° 2009-258 du 5 mars 2009 relative à la communication audiovisuelle et au nouveau service public de la télévision
[8] Arrêté du 3 janvier 2008 relatif à la radio diffusée en mode numérique par voie hertzienne terrestre ou par voie satellitaire en bande L ou en bande S fixant les caractéristiques des signaux émis

Nos travaux de recherche sur la scalabilité de scène multimédia ont été publiés dans différentes conférences [2][3][4][5][6], un journal international [1] et ont été partagés avec les partenaires industriels de RTL pour certains [13]. En outre, nous avons été en mesure de faire la démonstration de la diffusion de scènes scalables dans des conditions réelles à partir de la tour Eiffel dans le cadre des expérimentations autorisées à Paris par le CSA et dans quelques salons français ouverts au grand public comme le Salon de l'Automobile (2008) ou encore la salon SATIS-Le RADIO (2009) grâce à une étroite collaboration avec l'opérateur de diffusion TDF dans le cadre du projet de recherche RADIO+[9]. Des résultats applicatifs issus de travaux menés dans le cadre de projet sont disponibles en appendice de ce mémoire.

Plan du mémoire

Ce mémoire se compose des chapitres suivants:

- Le chapitre 2 décrit des scénarios de radio numérique qui nécessitent une adaptation de la présentation du service multimédia en fonction de l'environnement d'usage du contenu mais également en vue d'une transmission progressive.

- Le chapitre 3 constitue une étude des modèles sous-jacents les principaux formats de scène à partir d'une classification des caractéristiques d'une présentation en propriétés spatio-visuelles, temporelles et interactives. Notre modèle *Scalable MSTI* s'appuie précisément sur cette décomposition des présentations en trois composantes: la composante *Spatiale*, *Temporelle* et *Interactive*.

- Le chapitre 4 décrit l'état de l'art des techniques de prise de décision pour l'adaptation d'une présentation multimédia en l'organisant en quatre catégories d'approche. Notre modèle *Scalable MSTI* est fondé sur la combinaison de *la sélection de scènes alternatives* et de *la plasticité de scène*, les autres approches ne remplissant pas les conditions de notre cahier des charges pour l'adaptation multimédia dans le domaine de la diffusion. De plus, ce chapitre décrit également les transformations de scène qui peuvent être utilisées pour modifier les caractéristiques d'une présentation multimédia. Ces transformations génériques constituent la boîte à outils normalisée pour l'expression des *mises à jour d'adaptation* utilisées dans l'approche *Scalable MSTI*.

- Le chapitre 5 présente notre cahier des charges pour l'adaptation des présentations multimédia et décrit deux contributions qui ont conduit à la définition de la scalabilité de scène multimédia. Premièrement, l'adaptation de média scalables (et non-scalables) en coordination avec l'amélioration de leur présentation a été expérimentée. Deuxièmement, les *mises à jour d'adaptation*, utilisées dans l'approche *Scalable MSTI*, ont été expérimentées dans le contexte de la diffusion en comparant leurs performances aux autres approches applicables à l'adaptation des propriétés spatiales d'une scène.

- Le chapitre 6 décrit notre approche *Scalable MSTI* qui s'appuie sur une séparation entre la structure de scène de média (*Média*) et les propriétés de la scène (propriétés *Spatiales*, *Temporelles* et *Interactives*). De plus, la flexibilité apportée par la scalabilité de scène a été illustrée au travers du concept de *graphe d'adaptation*. Cette représentation introduit un ensemble de règles qui guident le processus de prise de décision pour l'adaptation de la présentation vers une option adéquate.

- Le chapitre 7 conclut ce mémoire et ouvre des perspectives en vue de prochains travaux portant sur la scalabilité de scène multimédia.

[9] ANR ANR 08-CORD-018 - Radio Augmentée et Diffusion de contenus Interactifs Optimisés

Chapitre 2 : Scénarios multimédia de radio numérique

La radio numérique terrestre est l'évolution technologique de la radio analogique FM. Elle consiste à diffuser le son d'une station de radio en utilisant les nouvelles technologies numériques et propose également un enrichissement visuel des programmes au travers de services multimédia synchronisés avec le son. Ces services multimédia synchronisés avec les programmes donnent par exemple accès à des éléments complémentaires pendant les flashs d'information, permet l'identification du morceau musical en cours, propose des liens vers des offres commerciales pendant les coupures publicitaires ou permet l'affichage de messages d'auditeurs transmis à la station de radio par SMS. En complément de ces données viennent s'ajouter d'autres types de services multimédia qui ne sont pas nécessairement synchronisés avec le son de la station de radio et qui proposent différentes fonctionnalités multimédia qui enrichissent l'expérience de l'utilisateur. Par exemple, des services de météo, d'astrologie, d'information sur le trafic ou encore une vue d'ensemble de la grille des programmes de la station peuvent être proposés aux auditeurs équipés d'un récepteur numérique avec écran, comme un autoradio GPS ou bien un cadre photo-radio par exemple. Lors de nos travaux, nous avons volontairement cherché à différencier ces services « asynchrones » et les services « du direct » qui sont synchronisés avec le son de la station de radio car ils revêtent des problématiques techniques différentes. Une liste non-exhaustive, mais relativement complète, de scénarios multimedia de radio numérique et des problématiques techniques associées est décrite en appendice (Appendice A).

Exemples de données multimédia synchronisées avec un morceau musical.

Exemples d'un service d'astrologie et d'une grille des programmes en complément du direct.

Lorsque l'on considère l'écoute de masse que représentent les millions d'auditeurs quotidiens d'une station de radio comme RTL, la diffusion de la radio par voie hertzienne demeure aujourd'hui la seule technologie économiquement viable. Cependant, les scénarios multimédia de radio numérique doivent explorer les opportunités que présentent les postes de radio connectés aux réseaux IP par une connectivité 3G ou Wi-Fi. Cette hybridation des services multimédia de radio numérique peut s'opérer, par exemple, en considérant une simple voie de retour par internet ou bien en cherchant à développer une véritablement voie de communication réciproque permettant le téléchargement ou encore la transmission par streaming de données complémentaires venant éventuellement enrichir les services multimédia diffusés par voie hertzienne.

Exemple d'un service hybride alliant données multimédia en diffusion et transmises par internet.

Chapitre 3 : Les modèles de description de scène

Dans ce chapitre, nous présentons les modèles qui sous-tendent quelques un des principaux langages de description de scène qui sont actuellement pris en charge par les lecteurs multimédia (à savoir SVG [118], MPEG-4 BIFS [59], SMIL [119] et HTML [120]). Ces standards ont déjà fait l'objet de différentes études [21][25][29]. L'approche adoptée dans ce chapitre d'état de l'art consiste à établir les principes généraux des modèles sous-jacents ces langages à partir d'une classification des caractéristiques des présentations multimédia en aspects spatiaux, temporels et interactifs. Préalablement à cette étude, ce chapitre fournit également la définition de certains termes utilisés tout au long de ce mémoire: *scène*, *présentation*, *description de scène* mais également *modèles* qu'ils soient « *de présentation* », « *de document (ou de service)*» ou « *de description de scène* ».

Définitions

Une *présentation multimédia* constitue l'expérience que l'utilisateur perçoit dans la réalité du monde physique en visionnant un contenu, c'est-à-dire lorsqu'il est confronté à l'écran tactile de son poste de radio numérique, par exemple. Comme le définit L. Hardman [48][10], cette présentation est le résultat de l'exécution au cours du temps d'unités d'information qui sont présentées à l'utilisateur. Plus précisément, une présentation multimédia est le rendu d'un document multimédia comme le définit S. Boll [24][11] ou, plus largement, le rendu d'un service multimédia, où un service est défini comme un document pouvant être modifié partiellement au cours du temps.

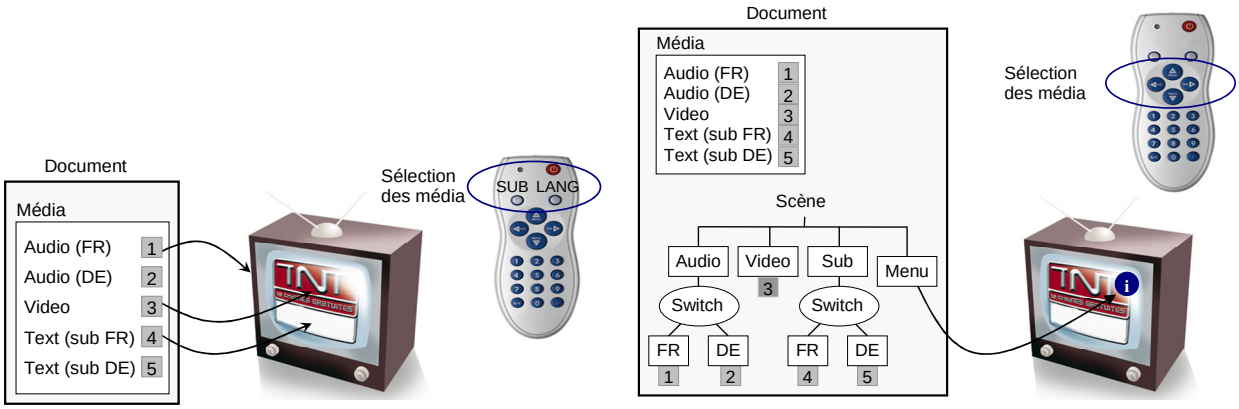
Un *document multimédia* constitue la captation d'une présentation sous la forme d'une description statique selon L. Hardman [48][12]. Un service multimédia constitue donc en quelque sorte un enregistrement en direct de l'expérience multimédia que l'utilisateur perçoit en exposition avec un contenu. Comme l'indique S. Laborie [78], 'un document multimédia est constitué d'un ensemble d'objets multimédia. Ceux-ci sont mis en page grâce à des techniques d'assemblage propres à l'auteur et forment la composition du document multimédia'. L'organisation des différents objets multimédia d'un service peut être implicite comme c'est le cas des services de television, par exemple, où l'affichage des sous-titres est géré automatiquement par les récepteurs TNT et où la sélection des pistes audio peut s'effectuer à partir de la télécommande du téléviseur en appuyant sur des touches dédiées.

Une *scène multimédia* constitue la description explicite de l'organisation d'une présentation multimédia. La composition d'un document (ou d'un service) multimédia peut donc faire appel à une scène de façon à organiser, dans l'espace et le temps, les objets multimédia auxquels l'utilisateur peut accéder par interactivité. Par exemple, un service de télévision peut inclure une scène multimédia et proposer à

[10] L. Hardman : « The term *presentation* refers to the runtime behavior of the information units presented to the user. »
[11] S. Boll : « A multimedia presentation is the rendering of a multimedia document. »
[12] L. Hardman : « The term document refers to static description of the presentation which can be stored. »

l'utilisateur une présentation spécifique à la chaîne pour l'activation des sous-titres et pour la sélection de la piste audio principale comme illustré ci-dessous.

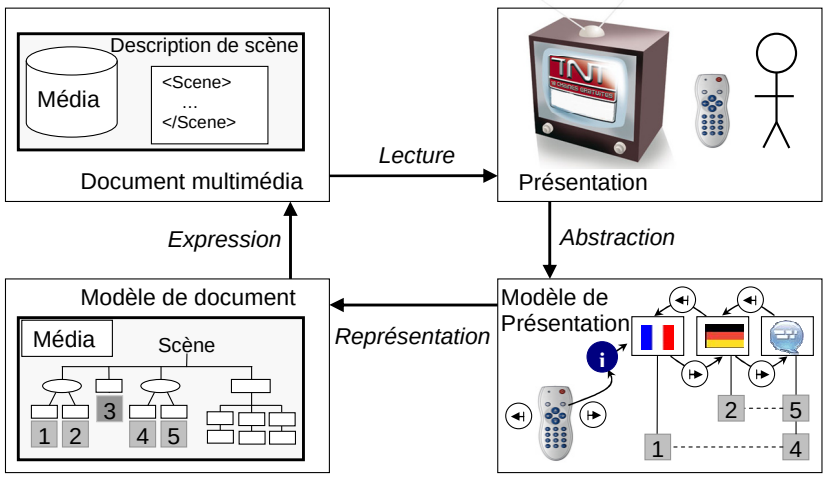
La *modélisation d'une présentation* consiste à abstraire le comportement perçu par l'utilisateur exposé à l'écran de son lecteur multimédia présentant un contenu multimédia. Cette modélisation reste cependant une notion théorique puisque qu'elle est en principe capable de modéliser l'ensemble des comportements multimédia possibles alors que l'expérience perçue par l'utilisateur est elle-même influencée par les limites techniques des récepteurs et les fonctions souhaitées par les éditeurs. Contrairement à la modélisation de phénomènes naturels par l'observation, la modélisation de présentations multimédia consiste à abstraire un phénomène artificiellement généré selon les souhaits d'un créateur du contenu.



Exemples d'un service de television sans scène multimédia (à gauche) et avec une scène multimédia (à droite)

Un *modèle de document* est la représentation d'un modèle de présentation dont les capacités d'expression informatique ont été définies de façon à répondre aux besoins d'un cycle de création-consommation du contenu. Par exemple, la compacité, l'exactitude, la robustesse, la réutilisabilité, l'adaptabilité de la description d'une présentation multimédia sont autant de critères qui conduisent à la conception de modèles de document différents qui représentent pourtant la même présentation. Ainsi, le processus de d'expression informatisée d'une présentation résulte d'un compromis qui varie selon les domaines d'application.

Un document multimédia peut s'exprimer (ou être publié) dans un ou plusieurs formats à partir d'une syntaxe spécifique (un langage). Un modèle de document rend donc éventuellement possible la multi-publication des documents multimédia. Pour les présentations faisant appel à une scène, le modèle de document relatif au langage utilisé pour la description de scène est appelé un *modèle de description de scène* dans ce mémoire.



La modélisation d'une présentation multimédia

Propriétés spatiales, temporelles et interactives des scènes multimédia

La définition d'un document multimédia proposée par C. Roisin [99] décrit une entité constituée 'd'un ensemble d'éléments d'information de base reliés par des relations de différentes nature (relations de composition, spatiales, temporelles et de navigation)'. La classification des modèles de description de scène que nous proposons dans ce chapitre s'appuie également sur les trois principaux aspects d'une présentation multimédia qui définissent l'expérience multimédia proposée aux utilisateurs, à savoir les propriétés spatio-visuelles, temporelles et interactives que l'on rencontre usuellement dans l'état de l'art [21][25][29].

Les modèles de description de scène permettent la représentation concrète des présentations, telles qu'elles peuvent être restituées à l'utilisateur au travers d'un lecteur multimédia compatible avec le format de scène modélisé. Notre étude s'est intéressée à l'expression des caractéristiques essentielles d'une présentation, à savoir la mise en scène des média qui la constituent, à partir des différents moyens d'accès à l'information dont dispose l'utilisateur:

- *l'observation des média affichés à un instant donné*. Ainsi, la dimension spatiale d'une présentation s'exprime informatiquement sous la forme de différents modèles de positionnement et de visibilité. Le style appliqué aux médias mais également les accessoires décoratifs qui les agrémentent font partie intégrante des propriétés spatio-visuelles de la scène dans notre classification puisqu'ils sont très souvent indéfectiblement liés à l'organisation spatiale de la présentation.
- *l'observation des média par leur renouvellement automatiquement au cours du temps*. Aux propriétés statiques des modèles spatiaux, s'ajoute la chorégraphie des média au sein d'une présentation qui nécessite la gestion de leur présence et de leur synchronisation. La temporalité de la scène dans son ensemble participe également à l'expérience multimédia de l'utilisateur face à une présentation.
- *l'observation des médias sélectionnés par interactivité*. Lorsque le lecteur multimédia le permet, la présentation des média est susceptible d'être dirigée par l'utilisateur lui-même. Au delà de la seule modélisation des actions de l'utilisateur, les différents modes de contrôle des média et de navigation définissent le comportement interactif de la présentation.

Cette classification des propriétés de scène a été utilisée dans notre approche *Scalable MSTI* pour constituer des groupes de transformation en vue de l'adaptation de la présentation d'un document (ou d'un service) multimédia. Ainsi, la sélection d'un groupe de propriétés spatiales d'une scène permet, par exemple, d'adapter l'ensemble des média affichés simultanément en fonction des caractéristiques de l'écran. De la même manière, la sélection d'un groupe de propriétés temporelles peut apporter un accès séquentiel à des médias qui ne peuvent s'afficher simultanément par manque d'espace sur l'écran. Enfin, la sélection d'un groupe de propriétés interactives ouvre l'accès à des média complémentaires à la demande de l'utilisateur, en fonction de ses souhaits par exemple.

Les modèles de description de scène introduits dans cette classification n'ont pas fait l'objet d'une comparaison systématique de leurs avantages et de leurs inconvénients. En effet, chaque modèle de description de scène présenté dans ce chapitre est susceptible de répondre aux besoins particuliers d'un domaine d'application. Parfois, ces modèles peuvent également être combinés avantageusement. Par ailleurs, cet inventaire ne constitue pas une liste exhaustive des modèles de description de scène. En effet, les modèles de document sont seulement une représentation partielle des présentations multimédia dont la complexité progresse rapidement avec l'augmentation des capacités des récepteurs (et des réseaux) et l'exigence croissante des éditeurs (et des utilisateurs). Les modèles de document sont donc amenés à évoluer régulièrement. Pour faire face à ce foisonnement, nous suggérons que les traitements appliqués aux documents multimédia reposent, dans la mesure du possible, sur des approches extensibles. Notre modèle *Scalable MSTI* prend en compte ce constat en s'appuyant notamment sur une transformation de scène conçue sur la seule séparation des propriétés spatiales, temporelles et interactives d'une scène.

Chapitre 4 : L'adaptation de scène multimédia

Dans ce chapitre, nous définissons la notion d'environnement d'usage telle qu'elle est envisagée dans notre étude et nous présentons différentes méthodes de prise de décision en vue de l'adaptation d'un document (ou d'un service) multimédia[13]. Nous présentons également les mécanismes de transformation de scène qui sont couramment utilisés pour modifier la présentation de documents multimédia.

Définition – Environnement d'usage

Les modèles de description de scène décrits au Chapitre 3 font l'hypothèse que les instructions d'une scène seront correctement traitées lors de la lecture du document (ou du service) multimédia. Cependant, ces conditions idéales de lecture ne sont pas toujours vérifiées et une telle hypothèse peut conduire à la dégradation de la présentation restituée. Ainsi, la qualité d'une scène multimédia ne se mesure pas uniquement à sa capacité à exprimer fidèlement une présentation claire et ergonomique. En effet, la restitution adéquate d'une présentation nécessite également une compatibilité entre la scène d'un document (ou d'un service) et les conditions de visionnage de celle-ci [134].

L'environnement d'usage d'un document (ou d'un service) multimédia est constitué de l'ensemble des contraintes imposées par l'infrastructure technique mise en œuvre pour la restitution d'une présentation mais inclut également les préférences de l'utilisateur. Cette différenciation régulièrement admise entre les contraintes techniques et les souhaits de l'utilisateur, comme par exemple chez S. Boll[14], ne transparait pas nécessairement dans les langages, comme MPEG-21 UED [66] ou encore W3C CC/PP [125], qui sont utilisés pour exprimer explicitement un environnement d'usage donné. En revanche, même si nous ne faisons pas cette différenciation, la personnalisation des média d'un document par rapport aux centres d'intérêt de l'utilisateur, telle que traitée par A. Scherp dans la plateforme MM4U [101], n'a pas été abordée lors de nos travaux car elle est difficilement envisageable dans le contexte de la diffusion. Les seules préférences de l'utilisateur prises en compte dans nos travaux concernent des souhaits généraux comme, par exemple, l'intérêt pour une partie de la présentation, la lisibilité de la présentation ou encore le temps qui peut être consacré au visionnage de la présentation.

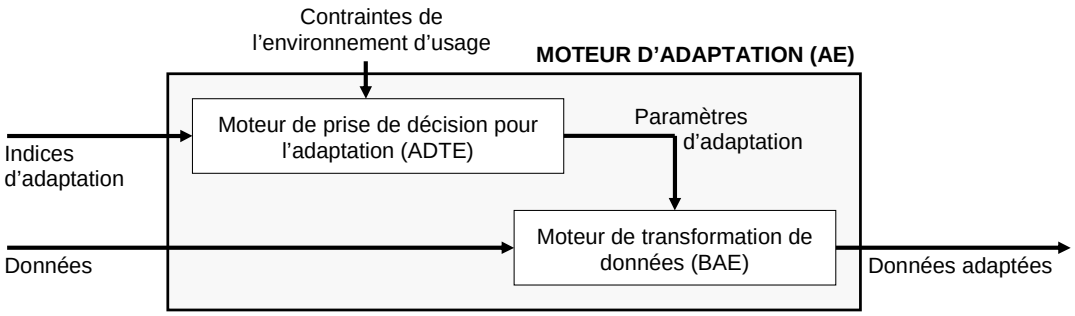
Architecture d'un moteur d'adaptation

Afin de prendre en compte la variété des environnements d'usage et garantir aux créateurs de contenu une restitution fidèle de leur production, une approche pragmatique consiste à concevoir des scènes multimédia uniquement à partir de fonctionnalités correctement prises en charge dans la majorité de ces environnements d'usage. Cette stratégie reste cependant frustrante pour les éditeurs qui doivent diffuser des documents multimédia dont l'attractivité se voit bridée par les environnements d'usage les plus contraignants. Cette pratique est également décevante pour les utilisateurs qui disposent d'équipements électroniques sophistiqués mais qui ne peuvent pas bénéficier de présentations multimédia avancées.

L'adaptation de scène multimédia répond à cette problématique en produisant des scènes multimédia conçues pour satisfaire les contraintes d'un environnement d'usage donné, exprimées sous la forme de paramètres d'adaptation (ou encore d'un contexte d'adaptation dans le standard MPEG-21 DIA [66]). Ainsi, l'adaptation de scène multimédia peut s'exprimer sous la forme d'un processus en deux étapes, comme décrit ci-après [91]. Premièrement, un moteur de prise de décision pour l'adaptation (ADTE) transforme les contraintes de l'environnement d'usage en paramètres d'adaptation en rapport avec la scène multimédia. Deuxièmement, un moteur de transformation de données (BAE) exécute la transformation de la scène.

[13] Bien qu'ils revêtent une définition différente exprimée au Chapitre 3, les termes 'document multimédia' et 'service multimédia' sont utilisés de façon interchangeable au cours de ce mémoire car notre approche s'attache à considérer les deux cas indifféremment.

[14] S. Boll : « For the extent of adaptability, we distinguish between adaptation to user interest, which adapts the contents of a document to the user's knowledge, professional background, and the like, and adaptation to technical infrastructure, which adapts to the technical infrastructure available to a user ».



L'architecture d'un moteur d'adaptation (de scène).

Taxonomie des techniques d'adaptation de scène

Une des difficultés majeures de l'adaptation de scène multimédia est la préservation des aspects sémantiques d'une présentation lors de la transformation de sa description de scène. Par conséquent, de nombreux travaux de l'état de l'art portant sur l'adaptation de scène ont été menés en vue de la mise au point d'une prise de décision pertinente pour l'adaptation des présentations multimédia. Ceux-ci sont organisés dans ce chapitre de façon à souligner quelques caractéristiques fondamentales qu'ils ont en commun. En revanche, les différentes approches de l'état de l'art utilisées par les moteurs de transformation pour l'adaptation de scène sont présentées dans ce chapitre selon des caractéristiques techniques abstraites issues de la représentation XML [116]: remplacement ou propagation de la valeur d'attributs et mise à jour d'éléments. Ces transformations sont volontairement non-spécifiques à l'adaptation de scène. En effet, le choix d'une transformation de scène générique facilite l'adoption des fonctionnalités qui en découle, à savoir l'adoption des fonctions d'adaptation de la scène multimédia dans notre cas. Cette conception de l'adaptation de scène en tant que transformation de scène usuelle, comparable à celles qui permettent l'évolution de la présentation d'un service multimédia, constitue un fondement de notre modèle *Scalable MSTI*.

Lors de cette étude, quatre grandes catégories d'approche pour l'adaptation de scène ont été identifiées:

- *L'adaptation de scène guidée par les média* consiste à configurer la transformation de la scène à partir de décisions d'adaptation prises au niveau des média. Ces approches sont notamment une réponse pragmatique aux besoins de certains média exigeant une infrastructure technique avancée mais réduisent aussi le champ d'application de l'adaptation de scène.
- *La publication de scènes adaptées* s'applique aux plateformes de génération de contenu où il est envisageable d'intégrer l'adaptabilité de la scène à la liste des objectifs du modèle de document utilisé pour la création des présentations. Ces approches permettent notamment l'automatisation de la production de présentations adaptables mais peuvent brider les fonctionnalités multimédia des présentations par une représentation trop abstraite ou par des règles d'adaptation trop rigides.
- *La sélection de scènes alternatives* consiste à choisir parmi un ensemble fini de présentations possibles celle dont la scène multimédia est compatible avec l'environnement d'usage. Ces approches garantissent aux créateurs de contenu une flexibilité éditoriale importante sur les présentations adaptées mais peuvent nécessiter en contrepartie un effort d'édition plus important pour la préparation de scènes multiples.
- *La plasticité de scène* permet la description des comportements adaptatifs de la présentation en tant que propriétés intrinsèques de la scène multimédia. Ces approches bénéficient de la flexibilité d'une formulation mathématique des scénarios d'adaptation d'une présentation mais souffrent d'une complexité d'édition plus importante qui peut notamment conduire à la limitation du champ d'application de l'adaptation de scène.

Face à la multiplicité des techniques d'adaptation de scène, une quête idéaliste consisterait à chercher à unifier toutes ces méthodes au travers d'une nouvelle proposition qui cumule leurs avantages identifiés ci-dessus tout en résolvant leurs difficultés. Selon nous, un tel objectif parait utopique car l'adaptation de scène est toujours un compromis entre le besoin d'une maîtrise éditoriale de la présentation résultant de

l'adaptation et la simplicité des nouveaux paradigmes d'édition à mettre en œuvre dans une chaîne de production multimédia pour prendre en charge l'adaptation. En revanche, ces différentes approches ne sont pas exclusives. Par conséquent, bien que la technique d'adaptation de scène adoptée par notre modèle *Scalable MSTI* repose essentiellement sur *la sélection de scènes alternatives*, elle peut être avantageusement complétée par *la plasticité de scène* ou s'articuler avec *l'adaptation de scène guidée par les média*.

Chapitre 5 : Le cheminement vers la scalabilité de scène multimédia

Dans ce chapitre, nous présentons les principaux facteurs qui nous ont conduits à la proposition d'un modèle de scène scalable. En particulier, nous établissons un bilan des différentes approches d'adaptation de scène présentées au Chapitre 4 au regard de nos problématiques relatives à la diffusion unidirectionnelle par voie hertzienne de services multimédia. Ensuite, nous présentons deux expérimentations préalables à la conception du modèle *Scalable MSTI* qui préfigurent le fondement notre proposition: la scalabilité de scène multimédia.

Analyse de l'état de l'art

Le croisement de la problématique de nos travaux présentée en introduction, à savoir la spécification d'*un processus de transformation générique, autonome et aux surcoûts réduits pour l'adaptation maîtrisée de services multimédia enrichis*, avec les résultats de notre analyse des différentes techniques d'adaptation de scène présentés au Chapitre 4 aboutit à quatre catégories d'approches qui sont envisageables dans notre cas:

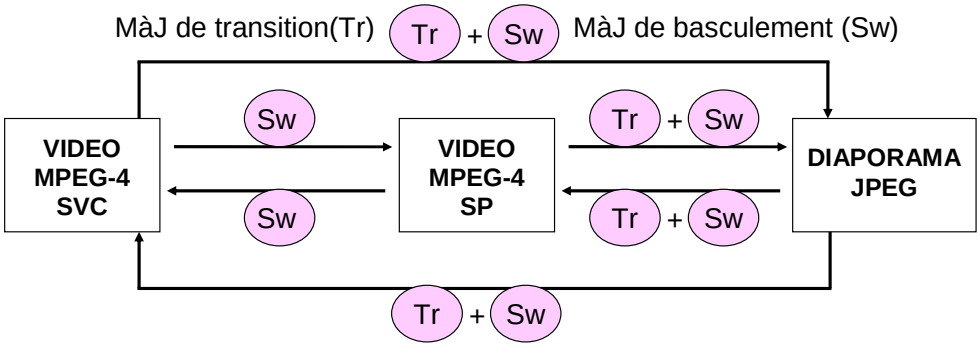
- *L'adaptation de scène guidée par les média* est pertinente lorsque l'éditeur a la possibilité de décrire des indices, voire de fournir des instructions, pour l'adaptation de la scène en fonction des transformations appliquées aux média. La limitation des surcoûts provoqués par le processus d'adaptation de la scène écarte, par exemple, certaines approches calculant itérativement l'ensemble des propriétés de scène à partir des nouvelles caractéristiques des média et d'indices d'adaptation laissés par l'éditeur [75][78]. De plus, la description des instructions d'adaptation de l'éditeur doit également répondre au besoin de la minimisation de la bande passante consommée par ce type de données supplémentaires décrivant l'adaptabilité de la présentation.
- *La sélection de scènes alternatives* se révèle être un excellent candidat, notamment lorsque l'éditeur peut explicitement maîtriser les critères de la prise de décision. En revanche, le poids que représentent les différentes alternatives d'une même présentation doit être optimisé afin d'atteindre un surcoût en bande passante acceptable.
- *La plasticité de scène* par la spécification de contraintes entre les différents éléments d'une scène constitue une option intéressante qui soulève cependant des difficultés lorsqu'il s'agit de considérer des contenus dynamiques et notamment des services multimédia. En outre, la maîtrise de la portée des contraintes d'adaptation de scène reste difficile pour l'éditeur et la résolution de celles-ci représente un surcoût de traitement qui n'est pas négligeable.
- *La plasticité de scène* par la spécification d'interpolations combine les avantages d'une représentation compacte et de la perspective d'optimisations pour le traitement des calculs correspondant aux configurations les plus usuelles. Cette approche peut être considérée comme une simplification de la plasticité de scène par contraintes et n'a finalement pour seul véritable défaut que sa capacité limitée à prendre en charge certains types d'enrichissement multimédia.

Sur la base de cette analyse, nous présentons dans ce chapitre deux contributions intermédiaires qui cherchent à étendre et à corriger certaines de ces techniques de façon à faire correspondre leurs caractéristiques au cahier des charges de nos travaux sur l'adaptation de scène multimédia. La première contribution associe *l'adaptation de scène guidée par les médias* à *la sélection explicite de scènes alternatives* de façon notamment à améliorer le compromis entre le besoin d'adaptation des média et de maîtrise de l'adaptation de scène. La deuxième contribution vise à optimiser l'approche par *sélection de scènes alternatives* en cherchant à réduire les surcoûts de l'adaptation de scène. Les expérimentations menées lors de cette seconde étude ont permis une évaluation comparative de ces surcoûts avec d'autres approches issues de la *plasticité de scène* notamment.

L'adaptation de scène guidée par les média

Nos travaux sur l'adaptation de scène guidée par les média nous ont permis de montrer comment une architecture d'adaptation conçue pour l'adaptation de média scalables peut évoluer par l'introduction d'un module d'adaptation de scène. Ce module d'adaptation de scène établit le lien entre un ensemble de média adaptés et une des présentations conçues par l'éditeur. Cette approche permet l'amélioration de la qualité globale du service multimédia adapté, telle que perçue par l'utilisateur, et ouvre la possibilité de nouveaux scénarios d'adaptation au niveau de la scène.

Le principal concept développé dans notre contribution est la modification guidée de la scène à partir de différentes versions adaptées d'un média comme illustré ci-dessous. En effet, la dégradation d'un média peut souvent être contrebalancée par un enrichissement de sa présentation, au travers de ce que nous appelons des *mises à jour pour le basculement de la présentation*. Par exemple, le basculement d'une vidéo vers un diaporama d'images dans le but d'économiser de la bande passante peut éventuellement être compensé par l'accès à un contrôle interactif sur les différentes images du diaporama. Des transitions fluides entre les différents états d'adaptation d'un média peuvent également être introduites dans notre approche par des transformations de scène que nous appelons des *mises à jour pour la transition de la présentation*. Par exemple, une vidéo au format VGA peut progressivement être redimensionnée avant de basculer vers une version QVGA. Les deux aspects, les mises à jour pour le basculement ou la transition de la présentation ont pu être combinées, lors de nos expérimentations, avec les autres transformations temporelles du service multimédia en s'appuyant sur le même type de mise à jour de scène introduite au Chapitre 3. En particulier, cette approche rend possible une prise de décision dynamique pour l'adaptation de la présentation et permet également une synchronisation entre les transformations des média et de la scène, ce qui contribue significativement au sentiment de qualité du service multimédia auprès de l'utilisateur. Ces mises à jour entre deux états de la présentation multimédia ont été retenues dans notre modèle *Scalable MSTI* au travers de la notion de chemin d'adaptation dans le processus de prise de décision pour l'adaptation. Par analogie, ces chemins d'adaptation peuvent également être utilisés pour définir des transitions entre les différents états d'une scène scalable.

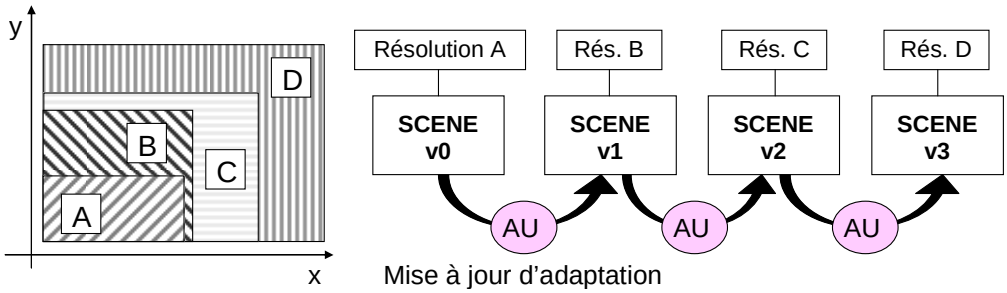


Modification de la scène entre les différents états d'adapation des média

La sélection de scène guidée par l'environnement d'usage

Nos travaux sur la sélection de scène guidée par l'environnement d'usage nous ont permis de montrer comment les mises à jour de scène peuvent être utilisées pour l'adaptation de la présentation de services multimédia en diffusion par rapport à l'environnement d'usage. En particulier, nos résultats d'expérimentaux montrent que l'adaptation par mises à jour de scène peut présenter de bon résultats sur les terminaux disposant d'un petit écran, de capacités de traitement réduites et de peu de mémoire. En outre, notre proposition permet aux éditeurs de configurer la granularité de la précision de l'adaptation des propriétés de la scène en fonction du surcoût en bande passante qu'ils jugent acceptable.

L'innovation principale de cette contribution sont les *mises à jour d'adaptation* qui constituent une transformation incrémentale de la scène permettant de dépasser les limitations dues à la redondance des scènes alternatives. En outre, la plasticité de scène par la spécification d'interpolations s'est révélée relativement coûteuse pour l'adaptation des propriétés spatiales d'une scène. Ces surcoûts augmentent a priori dans le cas de la plasticité de scène par la spécification de contraintes. Par conséquent, la production de différentes scènes alternatives, identifiées par des références explicites à des résolutions d'écran et générées dynamiquement de façon incrémentale, nous a permis de mettre au point une solution répondant au besoin d'adaptation d'une présentation multimédia vis-à-vis d'une résolution d'écran visée tout en respectant notre cahier des charge. Ces mises à jour incrémentales constituent le cœur de la transformation de notre approche *Scalable MSTI*.



Génération de scènes alternatives par des mises à jour d'adaptation

Chapitre 6 : Le modèle Scalable MSTI

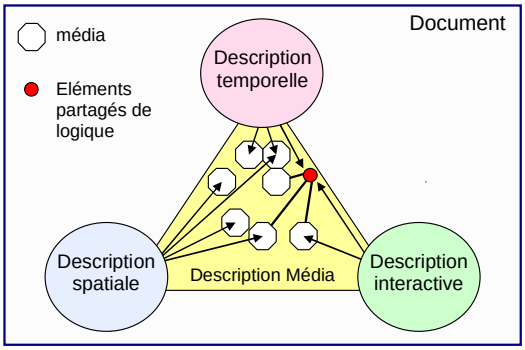
Dans ce chapitre, nous présentons le concept de scalabilité de scène multimédia. En particulier, nous présentons notre modèle *Scalable MSTI* qui définit une séparation entre la structure de scène dédiée aux média (description *Média*) et les propriétés de scène d'une présentation, qui sont elles-mêmes réparties en couches de scalabilité progressives (couches *Spatiales*, *Temporelles* et *Interactives*). Enfin, nous décrivons l'organisation hiérarchique des différentes options de la présentation d'un document (ou d'un service) multimédia scalable à partir des composantes STI de notre approche. Ces couches de scalabilité abstraites peuvent être représentées sous la forme d'un graphe d'adaptation qui guide l'adaptation de la scène en fonction de l'environnement d'usage.

La scène MSTI de base

La scalabilité de scène multimédia, telle que décrite en introduction de ce mémoire, consiste à organiser de façon hiérarchique la scène correspondant à une présentation adaptable. A l'origine de cette représentation hiérarchique, il existe un maillon primaire que nous appelons la scène « *de base* ». Dans notre modèle *Scalable MSTI*, cette scène de base repose sur les mêmes principes de description que n'importe quelle autre scène, qu'elle soit scalable ou pas. En particulier, elle est strictement exprimée selon le langage de description scène du document (ou du service) multimédia. Une scène *Scalable MSTI* est constituée de quatre composantes qui marquent une séparation, introduite au Chapitre 3, entre la structure de scène des média et les éléments de scène relatifs à leur présentation: une description *Média*, *Spatiale*, *Temporelle* et *Interactive*. Comme illustré dans la figure ci-après, les composantes relatives à la présentation multimédia, les composantes STI, se rapportent uniquement à la description *Média*. De cette façon, elles restent toujours indépendantes les unes des autres.

Nos diverses expérimentations ont permis de mettre en évidence que cette séparation des propriétés de scène est possible pour les standards SVG [118], MPEG-4 BIFS [59], SMIL [119]. Elle est de plus en cohérence avec les caractéristiques intrinsèques d'une présentation puisqu'un utilisateur peut accéder aux média d'une présentation soit au travers d'un affichage statique (propriétés spatiales), au cours du temps (propriétés temporelles) ou encore par actions (propriétés interactives). Par ailleurs les difficultés levées par la séparation des composantes STI pour certains modèles de description de scène multimédia ont

également été traitées, notamment par l'introduction d'une logique partagée au sein de la description *Média*.

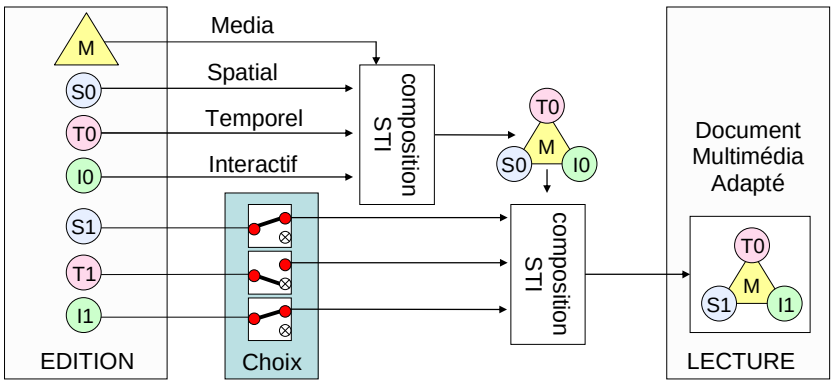


Un document multimédia dans le modèle Scalable MSTI.

L'enrichissement de la scène par scalabilité

La description *Média* d'une scène *Scalable MSTI* peut être vue comme la structure d'un document multimédia qui ne serait pas encore habillé de sa présentation. La description *Média* dispose donc d'identifiants permettant de lui assigner des propriétés de scène (`ref_id`). Le processus qui permet la transformation d'une description *Média* en une scène *Scalable MSTI* est appelée la composition STI. Celle-ci s'appuie sur les mises à jour de scène issues de l'état de l'art portant sur les transformations de scène décrites au Chapitre 3 au travers de commandes d'insertion (`<Insert/>`), de remplacement (`<Replace/>`), de suppression (`<Delete/>`) ou encore de déplacement (`<Move/>`).

Une des propriétés fondamentales de la composition STI pour la scalabilité de scène multimédia est sa récursivité, le résultat d'une composition STI pouvant être utilisé en tant que description *Média* d'une nouvelle composition STI comme illustré ci-dessous. Cette cascade est une conséquence directe de l'indépendance des composantes STI entre elles et tient également au fait qu'une description *Média* est exprimée dans le même format que la description de scène résultant de la composition STI, à savoir un format standardisé comme SVG [118], MPEG-4 BIFS [59], SMIL [119], HTML [120], NCL [105], ou Flash. Dans notre approche scalable, le processus de cascade peut être utilisé pour définir un enrichissement progressif de la présentation à partir d'une scène de base constituée du triplet (S_0,T_0,I_0). Etant donné qu'un objectif pour la scalabilité de scène multimédia est de minimiser la redondance qu'il peut exister entre deux scènes aux descriptions proches, l'une pouvant être considérée comme l'enrichissement de l'autre, une description *Spatiale* (S_{i+1}) ne remplace pas nécessairement l'entièreté des propriétés introduites par la précédente description *Spatiale* (S_i). Au contraire, celle-ci introduit uniquement les mises à jour nécessaires à l'évolution de la présentation du document multimédia. Ces mises à jour de scène sont rassemblées au sein d'une entité que l'on appelle *couche de scalabilité*.

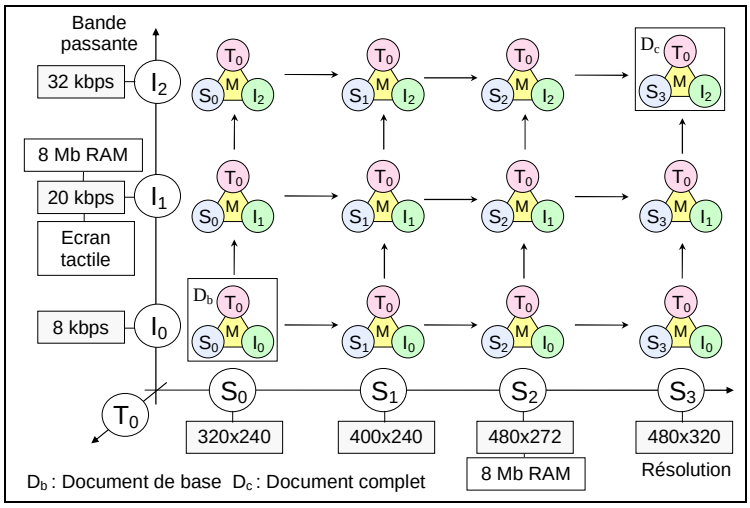


Cascade de compositions STI.

Le gaphe d'adaptation

L'édition d'un document *Scalable MSTI* consiste à définir et à remplir successivement des couches *Spatiales*, *Temporelles* et *Interactives* qui correspondent aux différentes alternatives de sa présentation. Ces couches STI incrémentales correspondent à des améliorations de la présentation multimédia du document et nécessitent parfois le relâchement progressif de contraintes de l'environnement d'usage. Ainsi, les différentes couches de scalabilité d'une scène peuvent être ordonnées selon trois axes de scalabilité où une relation d'ordre est définie pour chacun à partir d'un type de paramètre d'adaptation sélectionné par l'éditeur. Ainsi, des exemples typiques de paramètres d'adaptation en rapport avec les capacités des récepteurs sont la résolution (axe *Spatial*), les capacités de traitement (axe *Temporel*) et la consommation en mémoire (axe Interactif). Cependant, de nombreux autres paramètres d'adaptation sont également envisageables comme, comme par exemple, certains paramètres relatifs aux souhaits des utilisateurs: l'accessibilité aux personnes malvoyantes (axe *Spatial*), la durée de la présentation (axe *Temporel*) et le niveau d'expertise de l'utilisateur (axe *Interactif*). A partir de ce choix éditorial, les axes de scalabilité peuvent se voir attribuer des valeurs portant sur les paramètres d'adaptation leur correspondant et deviennent alors de véritables axes d'adaptation. Il est cependant important de noter que les valeurs de ces paramètres d'adaptation n'ont pas nécessairement besoin d'être précises. En effet, l'ordonnancement relatif de ces parameters peut suffire à lui seul pour définir la scalabilité d'une scène multimédia car cela permet, entre autres, d'éluder la difficile question de l'estimation précise de certains paramètres liés aux capacités des terminaux en proposant une évaluation in situ (la consommation mémoire varie d'un lecteur à un autre, par exemple).

Le processus d'adaptation qui consiste à sélectionner la présentation correspondant à un environnement d'usage peut être représenté dans le contexte d'un graphe à trois dimensions comme celui illustré ci-dessous. Il s'agit donc de progresser le long de chaque axe d'adaptation jusqu'à ce que les valeurs des paramètres d'adaptation excèdent les contraintes fixées par l'environnement d'usage: la résolution et la bande passante dans cet exemple. Etant donné que seuls trois paramètres primaires d'adaptation peuvent être utilisés au maximum, compte-tenu de notre représentation à partir de trois axes de scalabilité, des paramètres auxiliaires peuvent être introduits pour écarter certaines options potentiellement incompatibles avec l'environnement d'usage (la mémoire vive et la disponibilité d'un écran tactile, par exemple). En complément de ce mode de fonctionnement usuel du graphe d'adaptation, quatre extensions ont été introduites afin de mieux prendre en compte certains cheminements adaptatifs particuliers. Ainsi, des couches STI d'accès direct peuvent être introduites dans le graphe pour constituer des raccourcis (`randomAccess`). Des couches STI optionnelles peuvent également être définies afin de pouvoir être évitées lors d'un parcours adaptatif (`skippable`). La règle d'indépendance entre couches STI peut être ponctuellement enfreinte afin de limiter volontairement les possibilités du graphe aux seuls cas pertinents (`dependsOn` et `requiredFor`). Enfin, des couches STI bloquantes permettent de prendre en compte certains cas extrême d'adaptation où une des composantes STI prédomine (`blocks`).



Exemple de graphe d'adaptation.

Lors de nos travaux, nous avons été en mesure de valider expérimentalement que le graphe d'adaptation permet toujours l'adaptation d'une présentation à l'environnement d'usage dans le respect des souhaits de l'éditeur, le pire cas étant la sélection de la scène dite « de base ». Lors de nos tests, nous avons été amenés à développer plusieurs algorithmes. Un algorithme simple consiste, par exemple, à parcourir complètement l'axe *Spatial* jusqu'à épuisement avant de parcourir l'axe *Temporel* puis *Interactif* avant de recommencer. Un autre algorithme simple consiste à appliquer successivement une couche *Spatiale*, puis *Temporelle*, puis *Interactive* et ainsi de suite. Même si nos expérimentations nous conduisent à émettre quelques recommandations s'agissant du traitement des extensions de notre graphe d'adaptation, nous ne définissons pas d'algorithme de prise de décision pour l'adaptation. En effet, l'exploitation qui peut être faite du graphe d'adaptation dépend fortement des différentes contraintes qui pèsent sur le moteur d'adaptation selon les domaines d'application. Par exemple, certains algorithmes doivent pouvoir s'exécuter sans mémoire alors que d'autres peuvent tirer parti des étapes précédentes dans le graphe pour déterminer chaque nouvelle progression. De la même manière, certains algorithmes doivent être en mesure de traiter les couches de scalabilité dans leur ordre de réception alors que d'autres peuvent se permettre de construire une représentation complète du graphe avant de prendre une première décision d'adaptation.

Les différentes expérimentations de services multimédia que nous avons menées ont permis de mettre en œuvre la scalabilité conjointe des média (MPEG-4 SVC) et de la scène (MPEG-4 BIFS) par streaming dans le cadre de plateforme MPEG-21 DIA [66] et également de réaliser la diffusion progressive de scènes scalables dans le cadre du standard T-DMB [38]. Que ce soient la gestion conjointe de plusieurs graphes d'adaptation, la fusion de deux graphes d'adaptation ou encore la prise en charge de la modification dynamique d'un graphe d'adaptation, de nombreuses pistes d'exploration et d'optimisation restent ouvertes pour que la scalabilité de scène multimédia s'épanouisse pleinement.

Chapitre 7 : Conclusion

Dans ce mémoire nous avons présenté le concept de scalabilité de scène multimédia en introduisant une nouvelle méthode de prise de décision pour l'adaptation de la présentation de documents (ou de services) multimédia (notion de *graphe d'adaptation*) et en faisant évoluer les transformations de scène par mise à jour pour les besoins de l'adaptation (notion de *mises à jour d'adaptation*). Le modèle *Scalable MSTI*, que nous proposons, est un modèle de description de scène s'articulant autour de quatre composantes (la composante *Média*, *Spatiale*, *Temporelle* et *Interactive*, respectivement) qui fournissent une représentation flexible d'un document (ou service) multimédia adaptable. En particulier, le modèle *Scalable MSTI* correspond au besoin d'*un processus de transformation générique, autonome et aux surcoûts réduits pour l'adaptation maîtrisée de services multimédia enrichis* issu de la diffusion de services de radio multimédia en mode numérique par voie hertzienne terrestre.

Modèle de description de scène

Notre étude de l'état de l'art et des différentes approches traitant de l'adaptation de présentations multimédia au Chapitre 4 aboutit à une opposition forte entre les modèles de document généralistes comme ZYX [24], AHM [48] ou Madeus [79] et les modèles de document abstraits comme dans les approches MSSA [75], MM4U [101], celles basées sur des contraintes [84] ou des interpolations [36]. En effet, les modèles de document généralistes, aussi appelés méta-modèles dans notre classification, sont conçus pour identifier différents besoins d'adaptation et pour réconcilier ces besoins en un unique modèle de représentation d'une présentation multimédia. En revanche, les modèles de document abstraits, aussi appelés méta-format dans notre classification, sont issus de l'analyse d'un ou plusieurs formats de scène existants et représentent certaines propriétés de leur modèle de présentation, voire toutes leurs propriétés comme dans le cas du méta-format MM4U. Ces deux approches ont des avantages et des inconvénients. L'approche par méta-modèle permet notamment une flexibilité d'adaptation importante mais réduit considérablement les possibilités d'expression d'une présentation. L'approche par méta-format, quant à elle, permet la pleine exploitation des possibilités multimédia des formats de scène

standardisés lorsqu'elle ne cherche pas à couvrir plusieurs formats, mais la flexibilité de l'adaptation peut être limitée par leur modèle de présentation sous-jacent.

Lors de nos travaux, nos choix de conception ont été guidés par l'idée que la capacité d'adaptation d'un service multimédia (ou d'un document multimédia) constitue une valeur ajoutée pour celui-ci. En effet, la démarche d'adaptation provient souvent du constat que plusieurs utilisateurs ne peuvent ou ne pourront profiter de l'expérience multimédia, telle que prévue par l'éditeur, avec un niveau de satisfaction suffisant. Une telle hypothèse de 'valeur ajoutée' disqualifie les approches par méta-modèle parce que leur mise en oeuvre nécessite la remise en cause des plateformes de production existantes qui ont pourtant fait l'objet d'investissements importants pour être en mesure de rendre un service de qualité à la majorité des utilisateurs. Au contraire, nous proposons un modèle de document abstrait, notre modèle *Scalable MSTI*, qui ne constitue cependant un méta-format de multi-publication, puisque seules les caractéristiques génériques (propriété spatiales, temporelles et interactives) d'un format de scène donné, expression concrète d'une description de scène, sont abstraites dans le but de concevoir des scénarios d'adaptation comme dans l'approche CSVG [17] ou pour permettre un affichage progressif de la présentation [102]. Le modèle *Scalable MSTI* est donc un modèle de description de scène qui structure la scène de documents (ou de services) multimédia indépendamment de leur format en sous-ensembles hiérarchisés.

Adaptation de scène multimédia

Le cœur de notre approche repose sur une définition de la scalabilité de scène multimédia qui vise à enrichir la présentation d'un document multimédia existant. Dans le cas particulier de l'adaptation, un document initial est conçu pour être compatible avec la majorité des environnements d'usage, si ce n'est tous. Ensuite, la scalabilité de scène multimédia est mise en oeuvre pour enrichir l'expérience multimédia parfois médiocre de ce document dit « de base » par l'application de propriétés de scène supplémentaires compatibles avec l'environnement d'usage.

Etant donné notre cahier des charges issu de la diffusion de services de radio multimédia en mode numérique par voie hertzienne terrestre, comme décrit dans le Chapitre 2, nos choix ont été essentiellement guidés par la maîtrise des surcoûts occasionnés par le traitement de l'adaptation sur des récepteurs portatifs mais également par la minimisation de la bande passante nécessaire à la signalisation des éléments destinés à la mise en oeuvre de l'adaptation. En effet, le surcoût occasionné par la prise en charge d'un processus d'adaptation au niveau du récepteur (par une puissance de calcul supérieure, par une batterie à la durée allongée ou par une mémoire à la capacité étendue) doit toujours rester en cohérence avec le gain final procuré à l'utilisateur qui devra le financer. De la même façon, la séparation d'un service multimédia de radio numérique en plusieurs niveaux de présentation en vue de sa diffusion à partir d'une bande passante limitée, mais constante, est un compromis entre la qualité du service pour tous (en d'autres termes, le temps d'accès au service de base et le poids de ses données multimédia) et les capacités d'adaptation de ce service. En conséquence de ce cahier des charges issu de la diffusion de services multimédia, notre modèle *Scalable MSTI* repose sur deux processus successifs. Le premier est une transformation de la scène pour l'adaptation de la présentation d'un service multimédia (les mises à jour d'adaptation). Le deuxième est un processus de prise de décision qui oriente l'adaptation de la présentation en fonction des souhaits de l'éditeur du service (le graphe d'adaptation).

L'origine des capacités d'adaptation du modèle *Scalable MSTI*, introduit au Chapitre 6, provient de la sélection appropriée d'une scène multimédia parmi plusieurs alternatives. Dans notre approche, la représentation de ces scènes alternatives a été optimisée en rassemblant leurs propriétés spatiales, temporelles et interactives dans des descriptions progressives (les couches de scalabilité *Spatiales*, *Temporelles* et *Interactives*, respectivement) qui enrichissent une description initiale dite « de base » (la description *Média*). De façon à répondre aux contraintes de l'environnement d'usage et guider l'adaptation de la présentation, chaque couche STI peut être associée à la valeur de plusieurs paramètres d'adaptation liés aux besoins des scénarios d'application visés (une résolution d'écran minimale de 320x240 pixels, par exemple). Par conséquent, l'adaptation de plusieurs composants d'une scène est traitée par groupe de transformations atomiques dans l'approche *Scalable MSTI*, au sein des couches de

scalabilité, au lieu d'être traitée de façon individuelle pour chaque composant, comme c'est le cas de plusieurs approches de présentés au Chapitre 4. Cette approche a l'avantage de prendre en compte explicitement les liens sémantiques qui unissent les média élémentaires d'une même présentation.

L'expression concrète des couches de scalabilité à destination des scénarios d'adaptation, appelée mises à jour d'adaptation, est définie au Chapitre 5 et a été expérimentée au sein de deux plateformes de tests. Premièrement, l'adaptation simultanée d'un contenu vidéo scalable et de la présentation à laquelle il appartient a été explorée et testée dans le cadre d'une plateforme MPEG-21 de streaming sur IP [6]. Deuxièmement, les performances de mises à jour d'adaptation appliquées aux scènes multimédia enrichissant un service sonore de radio ont été évaluées par rapport aux approches comparables issues de l'état de l'art (voir Chapitre 4) dans le cadre de l'adaptation des dimensions spatiales de la présentation de services multimédia destinés à une diffusion en T-DMB [5]. Dans les deux cas, les mises à jour d'adaptation se sont révélées faiblement consommatrices en capacité de traitement, peu gourmandes en mémoire et économes en bande passante. En outre, ces expérimentations ont montré la remarquable flexibilité d'adaptation qui est offerte aux fournisseurs de documents ou de services scalables. En effet, le surcoût croissant des mises à jour d'adaptation au fur et à mesure de leurs applications successives permet d'apporter une solution efficace à destination des récepteurs les plus limités en reportant l'essentiel du surcoût de l'adaptation sur les couches d'amélioration les plus hautes, couches correspondantes aux terminaux les plus avancés dans les scénarios de diffusion de la radio numérique. Enfin, la bande passante consommée par la scalabilité d'une scène multimédia peut être contrôlée par la fusion des mises à jour d'adaptation, ce qui revient à réduire la granularité de la scalabilité d'une scène multimédia après sa génération, ou bien en introduisant des sous-canaux de transmissions à plusieurs débits [13].

Le modèle *Scalable MSTI* est une transposition de la scalabilité des média, comme MPEG-4 SVC [104] pour la vidéo ou bien encore JPEG 2000 [85] pour l'image, au domaine des documents ou services présentant une scène multimédia. Par conséquent, notre modèle inclut des axes de scalabilité (l'axe *Spatial*, *Temporel*, *Interactif*) auxquels sont associées des relations d'ordre. La correspondance entre les propriétés d'une scène (couches de scalabilité) et les caractéristiques envisagées pour l'environnement d'usage du contenu (paramètres d'adaptation) n'est pas spécifiée par notre modèle. Ceci permet notamment une configuration spécifique en fonction des scénarios d'application. En revanche, nous proposons de guider les algorithmes de prise de décision pour l'adaptation de la présentation en fléchant le processus de sélection entre les différentes versions de la scène. Dans le cas des scénarios d'adaptation où les environnements d'usage sont explicites, ces algorithmes de décision peuvent être représentés sous la forme d'un graphe d'adaptation où la sélection des couches de scalabilité de la scène est guidée par un ensemble restreint de règles définies au Chapitre 6 [3]. Ainsi, plusieurs algorithmes de prise de décision ont été expérimentés sur des scénarios d'adaptation typiques. Par exemple, les paramètres d'adaptation des axes STI ont été configurés avec des caractéristiques relatives aux capacités techniques des terminaux en s'appuyant sur la résolution (*Spatial*), les capacités de traitement (*Temporel*) et la mémoire vive (*Interactif*) [1]. Pour le besoin de la génération de résumés multimédia, les paramètres d'adaptation ont été configurés avec des caractéristiques relatives aux attentes de l'utilisateur [2], s'agissant notamment des zones visuelles d'intérêt de la présentation (*Spatial*), des moments d'intérêt (*Temporel*) et des actions d'intérêt (*Interactif*). Nos multiples expérimentations ont conclu que la flexibilité de notre graphe d'adaptation est suffisante pour définir des chemins d'adaptation menant à des versions pertinentes des scènes scalables tout en conservant un processus de sélection simple.

Scalabilité de scène multimédia

Le terme 'scalable' ou 'scalabilité' utilisé pour la définition de la scalabilité de scène multimédia dans le modèle *Scalable MSTI* doit être interprété comme défini dans le standard MPEG-4 SVC [104] et fait référence à la suppression de sous-ensemble du flux (flux de scène, dans notre cas) en vue de l'adaptation aux différents besoins de l'utilisateur et faire face à la multiplicité des capacités des terminaux ainsi qu'aux fluctuations du réseaux. Au travers de nos expérimentations, la scalabilité de scène multimédia a été abordée sous des angles différents mais compatibles.

Premièrement, l'adaptation d'une présentation multimédia, y compris l'adaptation des média scalables qu'elle présente, constitue une première étape vers la scalabilité de scène multimédia [7]. Nous avons montré dans le cadre du projet DANAE[15] que des décisions d'adaptation prises au niveau des média scalables pouvaient être utilisées pour guider l'adaptation de la scène en utilisant les mises à jour d'adaptation au sein de la plateforme d'adaptation MPEG-21. De plus, la scalabilité de scène multimédia, telle que définie dans le modèle *Scalable MSTI*, peut être combinée à la scalabilité des média, soit en définissant des graphes d'adaptation dépendants (en associant, par exemple, la résolution de la scène à la résolution d'une des ses vidéo) ou soit en définissant des graphes d'adaptation indépendants aux paramètres d'adaptation décorrélés.

Deuxièmement, l'adaptation de scènes *Scalable MSTI* qui intègrent des média non-scalables introduit également la scalabilité de scène multimédia en proposant un ensemble de média mis en scène dans l'espace et le temps au sein d'une présentation interactive compatible avec l'environnement d'usage. Cette forme de scalabilité de scène multimédia couvre la scalabilité des éléments graphiques comme ceux définis en SVG [118], la scalabilité introduite par les différents niveaux de détail d'un univers en trois dimensions comme défini en VRML [55] ou encore la scalabilité d'une police de caractère telle que définie dans le format TrueType [63]. Dans ce cas, les propriétés de scène (et plus particulièrement les propriétés spatiales) des documents (ou services) *Scalable MSTI* peuvent bénéficier d'une plasticité sans conséquence au niveau du graphe d'adaptation.

Troisièmement, le décodage et le rendu progressif de scènes *Scalable MSTI* peut faire appel à la scalabilité de scène multimédia de façon à fournir une présentation multimédia pertinente et cohérente à l'utilisateur pendant le chargement du contenu. Nous avons montré dans le cadre du projet RADIO+[16] que la diffusion de couches de scalabilité concaténées sur quelques flux élémentaires MPEG-4 pouvait être mise en place pour optimiser le compromis entre la qualité du contenu et les délais de transmission dans l'environnement T-DMB.

Perspectives

Le travail présenté dans ce mémoire peut être prolongé dans différentes directions qui vont de l'amélioration de la production de scènes scalables jusqu'à l'adaptation des scènes scalables aux préférences de l'utilisateur en passant par l'optimisation de la scalabilité des scènes multimédia et l'exploration de la scalabilité de scène pour la diffusion de la télévision.

Perspectives – Amélioration de la production de scènes scalables

En termes de production de contenu, les caractéristiques relatives à la scalabilité de scène multimédia introduites dans l'outil d'édition *BIFSEdit*, présenté en appendice (Appendice D), se sont cantonnées aux optimisations automatisables permettant l'amélioration de la production de services multimédia affichables progressivement. Par conséquent, les patrons MSTI générés lors de la phase d'édition ont permis de faciliter la production de contenu en présentant un ensemble de canevas multimédia prêts à l'emploi. En outre, ces canevas multimédia facilitent également la transmission progressive de services multimédia par l'identification explicite des informations les plus pertinentes qui doivent être diffusées en priorité. Dans le cadre d'un outil d'édition de véritables scènes *Scalable MSTI*, ces patrons MSTI pourraient également mentionner à l'utilisateur de l'outil les scénarios d'adaptation qu'ils intègrent. Ainsi, une approche multi-vue présentant une visualisation appropriée d'une scène scalable au sein d'un outil d'édition bénéficierait certainement des transformations incrémentales XML telles que définies par L. Villard [112].

Bien que la prise en compte des propriétés spatiales, temporelles et interactives des scènes multimédia dans les paradigmes d'édition a déjà fait l'objet par le passé d'études poussées, comme par exemple dans

[15] EU IST-1-507113 - Dynamic and distributed Adaptation of scalable multimedia coNtent in a context-Aware Environment http://danae.rd.francetelecom.com
[16] ANR ANR 08-CORD-018 - Radio Augmentée et Diffusion de contenus Interactifs Optimisés

le projet Madeus [79], la composition de plusieurs documents *Scalable MSTI* en un seul et même document scalable nécessite des travaux supplémentaires. Plusieurs approches peuvent déjà être identifiées pour la composition de deux scènes scalables. Premièrement, une des deux scènes scalables peut être adaptée et intégrée, en tant que scène non-scalable, dans une scène scalable maîtresse. Dans ce cas, une partie des capacités d'adaptation de la présentation est perdue lors de la composition des deux documents. Deuxièmement, les couches de scalabilité de chaque scène peuvent être distribuées dans une nouvelle scène scalable. Une telle fusion de la scalabilité de deux scènes conserve la totalité des capacités d'adaptation de la présentation dans le document final mais nécessite des axes d'adaptation compatibles. Enfin, une des deux scènes scalables peut être référencée dans la scène maîtresse, introduisant ainsi une sous-scène scalable. Dans ce dernier cas, le traitement de plusieurs graphes d'adaptation dépendants doit être exploré.

Perspectives – Optimisation de la scalabilité de scène multimédia

Notre contribution à la standardisation [8][9][10][11] d'un nouveau profil MPEG-4 BIFS permettant la description de mises à jour d'adaptation par utilisation de l'outil `EnvironmentTest` [60] nécessite encore une adoption par le standard T-DMB pour que l'adaptation de présentations par la scalabilité de scène multimédia soit possible sur les récepteurs de radio numérique dotés d'un écran (actuellement, seul Core2D est le profil MPEG-4 BIFS du T-DMB). D'ailleurs, la spécification du transport des données MPEG-4 BIFS en DAB+ [39], en cours de développement au sein du forum WorldDMB, ouvrirait le champ d'application de la scalabilité de scène multimédia à l'ensemble de la famille DAB [41]. Le profil MPEG-4 BIFS ExtendedCore2D a été défini comme une extension du profil Core2D de façon à garantir une rétrocompatibilité avec les récepteurs T-DMB déjà commercialisés. En revanche, la diffusion en T-DMB de services multimédia hiérarchiques, où une scène au profil Core2D peut être enrichie par des éléments de scène au profil ExtendedCore2D, doit encore faire l'objet de précisions, en faisant éventuellement appel à la possibilité de définir plusieurs descripteurs d'objet MPEG-4 initiaux (`Initial Object Descriptors`) dans un flux de transport MPEG-2. Par conséquent, la scalabilité de scène multimédia définie à partir d'une organisation hiérarchique des différents profils d'un format utilisé par un service (ou un document) multimédia, comme introduit par les profils scalable SMIL [119], reste à évaluer.

Lors de notre étude, nous avons pris le parti de spécifier la description de nos couches de scalabilité directement au sein du flux de scène, par exemple, au moyen du descripteur `EnvironmentTest` pour le format MPEG-4 BIFS, de transformations XSLT [122] pour le format HTML [120] ou encore de code JavaScript [64] pour le format SVG [118]. Cependant, une signalisation au moment du transport des scènes multimédia, au niveau MPEG-2 TS [54], MPEG-4 SL [56] ou RTP [51], pour l'adaptation de scène optimiserait encore davantage la transmission de scènes scalables en permettant la mise en place d'algorithmes de correction d'erreur inégale [26], la transmission distribuée de contenu [97] ou l'entrelacement inégal des couches de scalabilité [133]. De telles extensions (ou spécialisations) de ces standards pourraient être entreprises en appliquant le concept de couche de réseau abstrait (ou Network Abstraction Layers) à la scalabilité de scène multimédia [106].

Enfin, la nature dynamique des services multimédia est prise en compte dans notre approche *Scalable MSTI* par la mise à jour directe des couches de scalabilité STI du graphe d'adaptation au cours du temps. Ainsi, à l'occasion d'une modification de la scène, le graphe d'adaptation est partiellement réévalué à partir du triplet de couches STI précédent toutes celles qui ont été modifiées. Un nouveau chemin d'adaptation peut alors être décidé et mener à la modification partielle de la présentation courante en s'appuyant sur les couches de scalabilité dites « d'accès rapide » (RAL) les plus proches. Cependant, une modification portant sur la description *Média* d'un document *Scalable MSTI*, qui contient notamment les références aux média, peut nécessiter le rafraîchissement de l'entièreté de la présentation. D'ailleurs, cette limitation de notre approche se retrouve lors de la diffusion de flux de transport MPEG-2 lorsque le numéro de version de la table PMT est incrémenté (`version_number`) ou encore lors d'une transmission en streaming sur IP avec RTP à cause de la nature statique de la description SDP [52]. Des solutions de contournements ont été mises en œuvre dans le cas de la diffusion de services multimédia de

radio numérique T-DMB en exploitant la mise à jour de descripteurs d'objet MPEG-4 pour reconfigurer dynamiquement les flux de transport MPEG-2 du service. Cependant, la continuité du service multimédia dans le temps reste une problématique ouverte, particulièrement lorsque la bande passante disponible nécessite un chargement progressif du contenu.

Perspectives – Scalabilité de scène pour la diffusion de la télévision

Les exigences qui ont guidé nos choix dans nos travaux proviennent de la diffusion de services de radio multimédia en mode numérique par voie hertzienne terrestre. Bien que nos contributions puissent être appliquées directement au domaine de la télévision, la scalabilité de scène multimédia revêt des besoins qui peuvent être différents dans ce cas. En effet, la ressource allouée pour la diffusion de services de télévision en mode numérique étant supérieure à celle nécessaire aux services de radio numérique, un débit substantiel peut éventuellement être dégagé pour la diffusion de services multimédia. De plus, les capacités de traitement d'un boîtier TNT peuvent être plus avancées que celles d'un récepteur de radio mobile fonctionnant sur batterie. Par conséquent, une étude supplémentaire semble nécessaire pour explorer la scalabilité de scène multimédia lorsque de très nombreuses couches de scalabilité sont définies et pour évaluer l'impact que peut avoir les scénarios hybrides de la télévision diffusée par voie hertzienne et transmise par internet.

En particulier, la présence d'un moteur JavaScript dans le standard HbbTV [40] et les capacités de traitement grandissantes des boitiers de TNT[17] posent à nouveau la question de l'adaptation de scène par contraintes introduite au Chapitre 4. En effet, les capacités d'organisation spatiale de la scène en vue de l'adaptation restent limitées dans notre modèle *Scalable MSTI*, s'agissant notamment de la granularité de l'adaptation par rapport à la taille de l'écran qui est nécessairement non-infinie. Pour autant, l'expression de contraintes d'adaptation nécessite souvent l'utilisation de langages procéduraux qui peuvent s'avérer difficiles à manipuler [84]. L'association des deux approches peut donc constituer une solution performante dans ce cas.

Perspectives – Adapation de scènes scalables aux souhaits de l'utilisateur

L'adaptation de la présentation multimédia aux souhaits de l'utilisateur a été volontairement écartée du champ de notre étude de façon à limiter le domaine d'adaptation à l'environnement d'usage. Cependant, notre modèle *Scalable MSTI* présente des chemins d'adaptation qui « digressent » ou « fourchent » (voir Chapitre 6) et qui pourraient être utilisés pour renseigner des paramètres d'adaptation optionnels en relation avec les préférences de l'utilisateur.

En particulier, l'approche *Scalable MSTI* permettrait la mise en œuvre d'algorithmes de sélection à partir de retours d'expérience implicites de la part de l'utilisateur [95] en définissant notre graphe d'adaptation comme un espace configurable en fonction des préférences de l'utilisateur. Ainsi, les média supposés être préférés par l'utilisateur feraient partie des couches de scalabilité les plus basses (c'est-à-dire les plus prioritaires) alors que les nouveaux sujets nécessiteraient davantage d'actions de la part de l'utilisateur pour être visualisables. Ainsi, cette organisation hiérarchique de la scène évoluerait perpétuellement en fonction des choix de l'utilisateur en interaction avec le document (ou service) multimédia. Les préférences des utilisateurs pourraient également se voir utilisées pour précharger les couches de scalabilité les plus populaires et optimiser ainsi les délais de chargement des contenus.

[17] On notera, par exemple, que le boitier TV Freebox Revolution dispose d'un processeur Intel Atom CE 4100.

Chapter 1 Introduction

1.1 Background and motivations

The broadcasting of multimedia services is an evolution of the broadcasting of radio and television services. This transition is supported by the introduction of digital broadcasting standards such as DVB-T [42], DVB-H [44] or T-DMB [38] and the rapid development of complementary IP-based services such as radio podcasts or catch-up television services. Multimedia services associated with radio (or television) programs have to meet user expectations in terms of content but must also offer an attractive presentation. Therefore, the multimedia production of a broadcaster is driven by strong editorial rules and graphic standards that contribute to the character of the radio station. In fact, multimedia broadcast should be seen as a gateway to on-demand multimedia services. Such a multimedia front-end must offer an advanced visual and interactive experience to the user in order to enable the success of the technological upgrade and to conquer the next generation of users.

Broadcast scenarios imply challenges for multimedia services since broadcast environments gather specific constraints. In particular, these constraints heighten the difficulties raised when progressively downloading a multimedia document or streaming a multimedia service. For instance, the digital radio bandwidth dedicated to multimedia applications remains very low[18], and a return channel from digital radio receivers cannot always be assumed. Furthermore the radio communication channel may suffer transmission losses and the same multimedia service has to be broadcast to all while the cheapest handheld radio receivers only have limited processing power and memory capabilities.

On the one hand, the goal of a radio broadcaster is to constantly remain accessible to all auditors, whatever multimedia device they may use. On the other hand, broadcasters would like their multimedia service to maximize the satisfaction of their auditors by making use of the full technical capabilities of the delivery chain. As a consequence, it is imperative that multimedia services being broadcast do not suffer a breakdown due to a receiver with incompatible multimedia capabilities. In the same way, unbearable transmission delays due to narrow-bandwidth environments are not acceptable. In fact, the quality of multimedia services needs to be closely controlled by broadcasters over time and in various user's contexts in order to preserve an end-to-end communication channel with auditors.

In practice, the objective of a broadcaster is always to maximize its potential audience share. As a result, a usual content publication strategy consists in reducing the multimedia quality of the service for all. This quality reduction can take the form of restrictions identified in some particular usage environments. In that case, the other specific environments that cannot be made compatible are simply ignored. However, this workaround is not satisfactory since broadcast multimedia services undergo important limitations on high-tech receivers compared to on-demand services. Indeed, on-demand services usually imply content negotiations prior to delivery, which is not possible in broadcast scenarios. As a consequence, there is a need for high-performance adaptation techniques that would ensure a robust multimedia service for all while providing multimedia enhancements whenever the usage environment makes it possible.

1.2 Objectives and overview of the proposed solution

The motivation of our work is to provide technical support for the deployment of multimedia services optimized for the usage environments a broadcaster wants to address. Hence, our main objective is to specify *a generic, autonomous and low-overhead transformation process for the controlled adaptation of advanced multimedia services*. In fact, the horizontal market of broadcast receivers, such as radio and television receivers, requires an adaptation process that does not assume any return channel *(autonomous)*. To lower deployment costs, the adaptation process should use a standardized transformation of multimedia services *(generic)*. The live nature of multimedia broadcast implies a continuous adaptation process *(service)* whose quality needs to be validated according to the editorial

[18] A typical bandwidth from 8 to 32 kbps can be allocated to multimedia services

rules and the graphic standards of the broadcaster *(control)*. Finally, attracting the user attention requires carefully designed and fancy presentations *(advanced multimedia)* that must be adapted using a transformation process that does not significantly interfere with the user's environment *(low-overhead)*.

The underlying principles of our proposal lay in the scalable representation of visual media codec such as MPEG-4 SVC [104] or JPEG2000 [85]. Such a scalable or hierarchical representation of a media permits a generic and dynamic adaptation process based on the self-sufficient filtering of scalability layers. This simple filtering process can be performed with acceptable processing overheads compared to media transcoding. Furthermore, all adaptation options are configured during content encoding and classified into progressive scalability layers. These scalability layers correspond to media enhancements such as signal-to-noise ratio improvement or image resolution increase. Although the multimedia scene- which is the choreographer that organizes several media into a visual and interactive presentation over time- does not have the same properties as a video sequence or an image, we argue that media scalability concepts can be transposed to multimedia scenes, thus defining *multimedia scene scalability*. In few words, the principle defended in this dissertation is that the presentation of a multimedia service (or of a static document) can be modeled into three scalability axes by dividing all scene properties into progressive *Spatial*, *Temporal* and *Interactive* scalability layers. Thanks to the flexibility offered by standardized multimedia scene description formats such as SVG [118], MPEG-4 BIFS [59], SMIL [119], HTML [120], NCL [105], or Flash[19], we propose to organize the presentation of different versions of the same multimedia service (or document) into progressive scalability layers. As a consequence, the thesis addresses multimedia services (or documents) where content adaptation is assumed to improve an existing presentation or enables progressive playback. Different adaptation parameters can be used such as display dimensions as illustrated in Figure 1, the user's level of interest, the presentation duration, the receiver's capability to cope with CPU-demanding animations, etc.

Figure 1: Example of scene scalability and adaptation to screen dimensions.

1.3 Summary of the contributions

Our contributions towards the modeling of multimedia scene scalability result in a consolidated approach that we called the *Scalable MSTI* model. *Scalable MSTI* stands for scalable *Media*, *Spatial*, *Temporal* and *Interactive*. In our approach, the term "scalable" and "scalability" should be understood as defined by H. Schwarz for the MPEG-4 SVC standard: Scalability "refers to the removal of […] bit-stream in order to adapt it to the various needs or preferences of end users as well as to varying terminal capabilities or network conditions" [104]. As a matter of fact, multimedia scene scalability refers to the removal of scene properties in order to adapt a multimedia service (or a document) to the usage environment, each scalability layer, that we call STI layers, treating the adaptation of the various media components of a scene as a group. The two major innovations that constitute a breakthrough when compared to state-of-the-art approaches are the adaptation graph and adaptation updates that are described in the following paragraphs respectively.

First, an adaptation decision-taking process based on a graph that structures the adaptation capabilities of the multimedia presentation into a 3D-space representation has been specified. In our adaptation graph illustrated in Figure 2, each scene alternative is positioned onto *Spatial*, *Temporal* and *Interactive* scalability axes, also called STI scalability axes, according to adaptation requirements selected during content creation. As a consequence, each scene version represented by a (S_i, T_j, I_k) triplet can be

[19] http://www.adobe.com/products/flash/

associated with adaptation parameter values (APV) that help driving the decision-taking algorithm. Since progressive requirements are imposed on the scene properties enclosed into our STI layers, the adaptation decision taking process can be represented as the selection of an adaptation path within all possible scene versions. Additionally, fallback scalability layers can be defined so that a multimedia player can identify its optimal configuration by progressively applying *STI* scalability layers until they exceed its capability. The advantages of such an adaptation graph for multimedia scenes are numerous. For instance, it provides guidelines for presentation authoring (empty adaptation placeholders to be filled in). In broadcast scenarios, it enables optimizations for the progressive transmission of the multimedia scene by defining priorities within presentation parts. Furthermore, the adaptation of a dynamic scene to fluctuating environments constraints can be handled through a deterministic and efficient decision-taking algorithm which does not have to be aware of the actual scene properties contained inside STI scalability layers. Finally, the classification of scene properties according to domain-specific adaptation parameters opens a large panel of application scenarios relying on the same adaptation engine.

Second, a new transformation (based on updates) that modifies the scene properties for adaptation purpose has been proposed. These adaptation updates are timed transformations similar to scene updates that are applied to modify a multimedia scene over time. However, these updates require a context-based validation before being applied to a scene. In other words, these adaptation updates might be filtered out if they do not fit the current usage environment. Adaptation updates differ from traditional scene adaptation transformations because they require minimal overheads in terms of processing-power, memory and bandwidth; allow for low-level and customized scene transformations and have progressive adaptation costs (that evolve along with the richness of the multimedia service). As a result, the multimedia presentation targeted for the heavier constraints (*Base* document) has very low adaptation overheads and can be carefully designed into an all-purpose and robust content while multimedia presentations targeting advanced environments will benefit from multimedia enhancements by taking in charge supplementary adaptation costs.

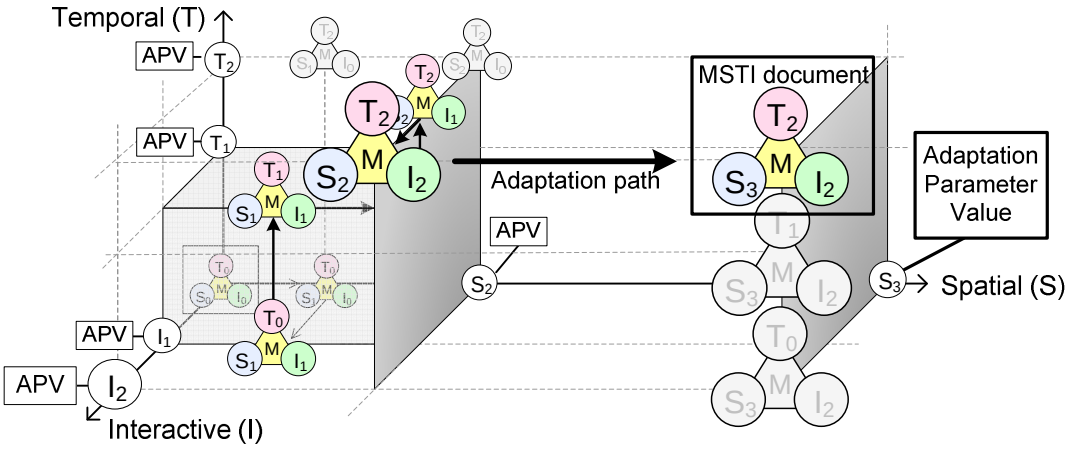


Figure 2: Adaptation graph of a Scalable MSTI scene.

1.4 Industrial context of our work and outputs

The work presented in this dissertation has been conducted within the industrial framework of RTL as a full-time employee between October 2006 and January 2010 (research position defined by a CIFRE contract - Conventions Industrielles de Formation par la REcherche). RTL, the leading radio station in France, belongs to RTL Group[20] which is the leading European entertainment network with 43 television channels, 31 radio stations in 10 countries and worldwide productions. In the scope of my activities, the four radio stations of RTL Group in France, namely RTL[21], RTL2[22], Fun Radio[23] and RTL L'Equipe[24]

[20] http://www.rtlgroup.com
[21] http://www.rtl.fr
[22] http://www.rtl2.fr
[23] http://www.funradio.fr
[24] http://www.rtl-lequipe.fr

have been my innovation playground but also put in light some strong requirements that guided my studies.

The timescale of my work was influenced by the legal time frame of the launch of Digital Radio in France which is defined by the 'freedom of communication' act (Loi n° 86-1067 du 30 septembre 1986 relative à la liberté de communication) since 2004[25], completed in 2007[26] and 2009[27]. In March 2008, the Conseil Supérieur de l'Audiovisuel (CSA) launched a first call for tender in the areas of Paris, Marseille and Nice-Cannes for the broadcast of Digital Radio using the T-DMB standard in band III[28]. I conducted standardization efforts within the MPEG Systems group and the WorldDMB technical committee, on behalf of RTL group, in order to improve T-DMB radio services efficiency and interoperability. These contributions, technically sustained by Telecom ParisTech, resulted into the update of the T-DMB standard published by ETSI in April 2009 [38] and the definition of new MPEG-4 BIFS profile published by ISO/IEC in July 2010. [60]

Our research activities on multimedia scene scalability have been published in [12] several scientific conferences [2][3][4][5][6][7] and journal [1] and shared with industrial partners of RTL [13]. Additionally, we demonstrated the broadcasting of scalable multimedia services in live conditions from the Eiffel tower and in several major broadcast showcases in France such as the Paris Motor Show[29] (2008) and the SATIS-Le RADIO[30] (2009) thanks to a close cooperation with the TDF[31] network operator as part of the RADIO+ research project[32]. From an industrial perspective, the adaptation of multimedia services to receiver's capabilities has first been tackled by defining minimum requirements[33] for digital radio receivers in cooperation with the SIMAVELEC[34] and DIGITALEUROPE[35] (former EICTA) at the European level. Additionally, provisions for the future adaptation of multimedia digital radio services through scene scalability have been standardized in T-DMB and MPEG-4 BIFS standards [8][9][10][11]. Finally, the progressive rendering features of scalable multimedia scenes helped the development of the multimedia production chain of RTL in coping with the narrow-bandwidth constraints of digital radio. In particular, the *Scalable MSTI* approach was partly introduced in an in-house authoring tool (*BIFSEdit*) that publishes scalable multimedia services to a generic interface. This interface is described as an XML-based specification called *DMB Markup Language* that I engineered for RTL. The *DMB-ML* can either be used to produce scalable multimedia scenes by using on-the-shelf T-DMB multiplexers such as the AllegroDVT[36] T-DMB encoders (AL1020) or validated using a multiplexer simulator developed by RTL (*MuxSimulator*).

1.5 Outline of the dissertation

Chapter 2 gives an overview of multimedia digital radio scenarios and explains why presentation is needed for the broadcasting of 'augmented' radio services.

Chapter 3 provides a survey of the models underlying the principal scene descriptions (or format) currently implemented in multimedia players. The classification of scene properties into spatio-visual, temporal and interactive components constitutes the basis of our *Scalable MSTI* model.

Chapter 4 describes state-of-the-art adaptation decision-taking techniques and identifies four main approaches. The *Scalable MSTI* approach is based on the combination of scene alternatives selection and

[25] Electronic communications and audiovisual communication Act, 2004, 9th of july.
[26] Audiovisual broadcasting modernisation & TV of future Act , 2007, 5th of march.
[27] Audiovisual communication and new TV public service Act, 2009, 5th of march.
[28] Signal characteristics related to the terrestrial and satellite broadcasting of digital radio Decree, 2008, 3rd of january
[29] http://www.mondial-automobile.com
[30] http://www.satis-expo.com
[31] http://www.tdf.fr
[32] ANR ANR 08-CORD-018 - Radio Augmentée et Diffusion de contenus Interactifs Optimisés
[33] EBU, Recommandation R126, http://tech.ebu.ch/docs/r/r126.pdf
[34] http://www.simavelec.fr (one of the French association for consumer electronics industry)
[35] http://www.digitaleurope.org
[36] http://www.allegrodvt.com

scene plasticity since the two others (media-based scene generation and custom scene publishing) failed to address our requirements. Additionally, scene transformations targeting the modification of the scene properties of multimedia services are also detailed in this chapter. These scene transformations constitute the standardized toolbox used to express the adaptation updates of the *Scalable MSTI* approach.

Chapter 5 details our adaptation requirements and describes two experiments we conducted in order to pave the last miles toward multimedia scene scalability. First, the synchronized adaptation of scalable and non-scalable media along with presentation enhancements is shown. Second, scene adaptation updates, as defined in the *Scalable MSTI* approach, have been experimented in broadcast scenarios through a comparative study with two other approaches applicable to the spatial adaptation of multimedia presentations according to progressive context requirements.

Chapter 6 describes our *Scalable MSTI* proposal by specifying the separation of the media structure (*Media*) and scene properties (*Spatial*, *Temporal* and *Interactive*). Furthermore, the flexibility offered by *Scalable MSTI* scenes is illustrated through the concept of adaptation graph. This adaptation graph defines a concise set of rules which guide adaptation decision-taking algorithms to a suitable presentation.

Chapter 7 concludes this dissertation and gives a perspective on future work.

1.6 Published work

1.6.1 Research papers

[1] B. Pellan and C. Concolato, Authoring of Scalable Multimedia Documents, In Multimedia Tools and Applications (MTAP), vol. 43, no. 3, July 2009, pp. 225-252.

[2] B. Pellan and C. Concolato, Summarization of Scalable Multimedia Documents, In proc. of the International Workshop On Image Analysis for Multimedia Interactive Services (WIAMIS), London, England, May 2009, pp. 304-307.

[3] B. Pellan and C. Concolato, Adaptation of Scalable Multimedia Documents, In proc. of ACM Symposium on Document Engineering, São Paulo, Brazil, September 2008, pp. 32-4.

[4] B. Pellan et C. Concolato, Scalable Multimedia Documents for Digital Radio, In proc. of ACM Symposium on Document Engineering, São Paulo, Brazil, September 2008, pp. 221-222.

[5] B. Pellan and C. Concolato, Spatial Scene Adaptation in Broadcast Environment, In proc. of the International Conference on Multimedia & Expo (ICME), Hannover, Germany, June 2008, pp. 389-392.

[6] B. Pellan and C. Concolato, Media-Driven Dynamic Scene Adaptation, In proc. of the International Workshop On Image Analysis for Multimedia Interactive Services, Santorini, Greece, June 2007, pp 67-70.

[7] M. Ransburg, R. Cazoulat, B. Pellan, C. Concolato, S. De Zutter, C. Poppe, A. Hutter, H. Hellwagner, R. Van de Walle, Dynamic and Distributed Adaptation of Scalable Multimedia Content in a Context-Aware Environment, Proc. of the European Symposium on Mobile Media Delivery (EuMob 2006), Alghero, Italy, September 2006.

1.6.2 Contributions to the MPEG standardization body

[8] J. C. Dufourd, C. Concolato, J. Le Feuvre and B. Pellan, Response to the CfP on additional BIFS technologies for Interactive Services for Digital Radio, M16637, London, June 2009.

[9] C. Concolato, J. Le Feuvre and B. Pellan, WD 1.0 of Amd7 of BIFS for Interactive Digital Radio Services, N10439, Lausanne, February 2009.

[10] C. Concolato, B. Pellan and M. Brelot, Requirements for a new BIFS profile to support interactive Digital Radio, N10228, Busan, Korea, October 2008.

[11] B. Pellan, Y.-K. Lim et C. Concolato, New BIFS Profile for Interactive Digital Radio, M15550, Hannover, Germany, July 2008.

1.6.3 White papers

[12] B. Pellan (RTL), J. Launay, M. Brelot, Vincent Simonacci (Radio France) and Michel Raichman, Multimedia Scenarios for Digital Radio, 2009, to be published on the RADIO+ website.
[13] B. Pellan (RTL), White paper on DMB Radio, 2010, to be published by the WorldDMB forum.

1.6.4 RADIO+ project specifications

[14] C. Concolato, B. Pellan, J. Launay, M. Brelot, R. Bouqueau, A. David, D. Jaillet and D. Vincent, Spécification de la chaîne de radio augmentée (architecture et interface), ANR 08-CORD-018 (RADIO+), L1.1b, January 2010.
[15] B. Pellan, D. Vincent, R. Bouqueau, A. David and C. Concolato, DMB-ML: Interface chaîne multimédia - codeur T-DMB, ANR 08-CORD-018 (RADIO+), L3.1a, January 2010.

Chapter 2 Multimedia digital radio scenarios

Digital radio is the evolution of the analog FM radio. It consists in delivering the sound of a radio station to its listeners using digital technologies and also include multimedia services that are synchronized with the audio stream, also called 'live' services (e.g. news programs, music programs, commercial breaks, display of auditor's messages via SMS). Additional multimedia services, not necessarily synchronized with the audio stream, such as weather forecasts, astrology, traffic or an electronic program guide services can also be transmitted to enrich the radio programs. We call them 'asynchronous' services. As further detailed in Appendix A, all of these digital radio scenarios include interactive usages for rich-media radio receivers by proposing extended information, navigation in a multimedia service, interactive quiz or services that require user inputs.

When considering mass services delivered to millions of listeners at the same time, digital radio broadcasting still remains a cost-effective technology. However, multimedia digital radio scenarios have to explore the new service opportunities on radio devices connected to 3G or Wi-Fi IP networks, either by considering a simple return channel or by using a two-way communication channel that allows the download or the streaming of auxiliary information as a complement to multimedia services being broadcast.

Figure 3: Examples of an astrology and an EPG service in addition to the live service.

The scenarios illustrating these multimedia functions led most of the developments that were conducted as part of the Radio+ project[37] and participated in defining the requirements that influenced the technical choices of our *Scalable MSTI* model. In particular, these digital radio scenarios were expressed as a unitary test suite for the multimedia production chain developed in the context of the Radio+ project. The same content illustrates all along this dissertation the adaptation experiments that we conducted in the scope of our research activities. All in all, more than 10 hours of live multimedia programs synchronized with the audio stream was produced for testing and demonstration purposes.

In the following, an overview of some essential multimedia functions of digital radio services, extracted from the technical scenarios detailed in Appendix A, are presented focusing on the synchronization of audio and multimedia data (Section 2.1), the feeding of live and asynchronous multimedia data (Section 2.2) and the interactivity of multimedia presentations (Section 2.3).

2.1 Audio and multimedia data synchronization

A digital radio service is composed of natural or synthetic images that are presented to the radio listener as part of a presentation in order to provide an advanced multimedia user experience. The technical constraints applied to the signaling and transport of these images highly depend on the level of synchronization that is required with the audio stream (ranging from 40ms to no synchronization at all). When switching on a digital radio station, the multimedia service is progressively displayed as images are retrieved and inserted into the presentation.

[37] Radio+ consortium gathers RTL, Telecom ParisTech, Allegro DVT, Cameon and TDF. Radio+ project (08-CORD-018) is partially financed by the French Research Agency (L'Agence Nationale de la Recherche) as part of the CONTINT program.

A digital radio service is based on a structure that organizes the different media of the presentation (audio, image, video) and also includes graphics or text paragraphs for instance. As a consequence, the presentation organizes efficiently-coded synthetic elements that can also be tightly synchronized with the audio stream.

The images, graphics and text elements that are presented in a digital radio service can be refreshed very frequently as illustrated in Figure 4, especially for synthetic elements that consume a very limited amount of bandwidth. The technical mechanisms used in such scenarios to perform presentation transformations (e.g. adaptation transformations) must ensure short latency from each steps of the whole digital radio transmission chain.

Figure 4: Examples of loose (album cover) and tight (song lyrics) synchronization.

2.2 Live and asynchronous multimedia data

Contrary to television channels, many radio stations produce live programs. As a consequence, digital radio services are composed of live media whose presentation can be prepared off-line or dynamically generated during the radio show. In that case, the live authoring of advanced multimedia presentations can only consists in filling in presentation templates with live information. As a consequence, highly-structured presentations as illustrated in Figure 5 can be used in order to ease live authoring or to enable automated content generation.

Figure 5: Examples of highly-structured presentations (tables and text area).

The update of images, graphics and text elements over time depends on synchronization rules that are strengthen with supplemental constraints in live broadcasting scenarios. Indeed, content pre-fetch is not possible during live productions which require a high reactivity. Instead, such dynamics can be handled by smartly designed interfaces and transitions effects which emphasize live aspects. For that purpose, smooth transitions are required when progressively loading a multimedia presentation.

Figure 6: Example of a dual-headed radio service.

During the content transportation to the radio receiver, the digital radio service can be partially (or completely) replaced by a local program or complemented with presentation coming from other sources than the terrestrial broadcast (e.g. IP-based connection on the receiver). The integration of such external multimedia data sources requires a harmonized management of the visual service as illustrated in Figure 7. Although dedicated visual areas can be safely reserved for commercial banners, the integration of interactive content in a dynamic presentation raises contractual obligations that must always be respected.

Figure 7: Examples of presentation designed to integrate external scene data.

2.3 Multimedia interactivity

Digital radio services can extensively exploit the functionality offered by interactivity standards now supported in digital technologies as illustrated in Figure 8. Of course, interactive interfaces should guarantee a satisfactory and similar behavior on all radio receivers in order to ease content accessibility.

Figure 8: Example of an interactive quiz synchronized with the audio stream.

2.4 Why and what presentation adaptation is needed ?

The production of digital radio services has to cope with the expectations of radio stations which have numerous and ambitious application scenarios in mind. However, the broadcasting of multimedia services also has to cope with the significant constraints of the digital radio environment. In practice, the quality of the multimedia services delivered to radio listeners can be limited by two main factors.

First, each receiver manufacturer is free to decide which function needs to be implemented in its multimedia devices. Indeed, the support of a multimedia standard by a digital receiver often relies on a technical profile which defines its minimal requirements (see Appendix C). As a consequence, on the one hand, multimedia radio services offer content that receivers might filter out depending on their level of support. For instance, the interactive properties of a presentation might be removed on receivers without interactive means. In the same way, animations might be ignored on car-radio receivers while driving for safety reasons. On the other hand, multimedia radio services offer content that might not be satisfactory on some high-end devices. For instance, the quality of a standard image resolution for common radio receivers might not be sufficient on a 10-inch screen.

Second, the broadcast of a multimedia service requires constant content repetition in order to provide a permanent access to the service. As a consequence, the configuration of digital radio broadcast is ruled by transmission delays. Some timed media such as audio or video cannot suffer any jitter due to their high frequency[38]. Multimedia services may have a lower dynamicity: they can be progressively transmitted with longer delays in order to increase their final quality.

Our *Scalable MSTI* model aims at tackling these adaptation needs taking into account the constraints that the continuous production of rich-media content induces.

[38] The duration of MPEG-4 HE-AAC v1 Access Unit is about 40ms at 48kHz when SBR is used (2048 samples per frame are used). The duration of MPEG-4 AVC Access Unit is about 2s at 0.5fps (slideshow) when only IDR NAL units are used.

Chapter 3 Scene description models

This chapter aims at giving an insight on the models underlying the principal scene description languages currently implemented in multimedia players (i.e. SVG [118], MPEG-4 BIFS [59], SMIL [119] or HTML [120]). These standards have already been documented in several surveys [25][21][29]. Our approach in this chapter consists in classifying and detailing the main principles of their underlying model. These typical properties, available in one or more languages, are illustrated on concrete examples.

First, some definitions related to the multimedia domain are introduced (Section 3.1). Then, this chapter follows an organization based on three aspects that lay the foundations of most, if not all, scene descriptions models: spatio-visual properties (Section 3.2), temporal aspects (Section 3.3) and interactions (Section 3.4).

3.1 Definitions

This section introduces some definitions for keywords that are used through this dissertation, like *multimedia scene*, *presentation*, *media component*, *scene description*, *document model*, *presentation model and scene description model*.

3.1.1 Multimedia scene and presentation

A *multimedia scene* specifies the organization of a presentation composed of several elementary media (audio, images, text…) as perceived by the user. As a consequence, a scene can be included in a multimedia document or in a multimedia service in order to define "the runtime behavior of the information units presented to the user" as L. Hardman defines the term *presentation* [48]. Multimedia documents that do not enclose or reference any scene usually rely on automated presentation rules that are sufficient to cope with specific scenarios. For instance, common Digital Television (DTV) services can be considered as multimedia services since they are usually composed of one video stream, one or several audio streams for multilingual support and one or several text streams for subtitling purpose. However, DTV scenarios assume a choice between the different available languages and the activation (or not) of a single subtitling option. Therefore, the presentation of such DTV services remains limited: it consists of rendering a single audio stream synchronized with a full-screen video component possibly overlaid by a couple of text lines in its bottom area as illustrated in Figure 9. In that case, it should be noticed that the presentation of the subtitles may differ from one TV set to another since no explicit scene is provided for managing the text layout and style.

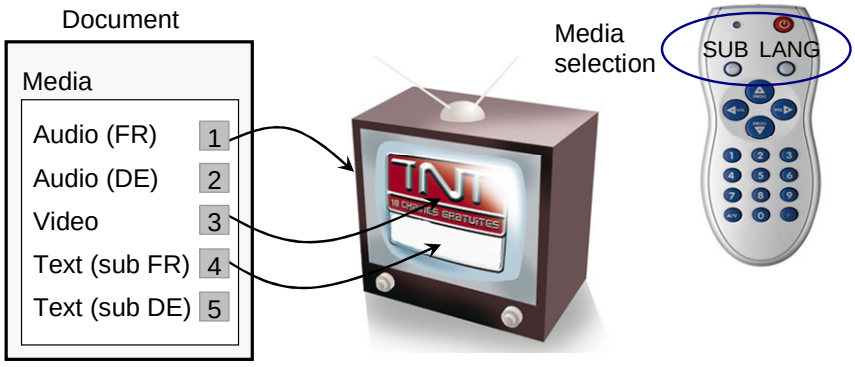


Figure 9: Example of a DTV presentation without any scene.

In our work, we have focused our studies on multimedia documents whose presentation is explicitly specified through a scene. In that case, a presentation similar to the DTV service described in Figure 9 can also be proposed to the user as illustrated in Figure 10. Additionally, the text properties can be defined either by the editor (by changing the text color over time depending on the background color) or by the user (by increasing the font size for a better text legibility). In practice, multimedia services targeting specific scenarios can also be achieved by defining a scene (possibly predefined and

normalized) as it is the case for mobile TV in Korea [38]. Indeed T-DMB standard can be used to broadcast simple audio/video scenes without any interactivity. As a consequence, assuming that the presentation of multimedia document is determined by a scene does not discard any application scenario but extends existing one with new functionalities. The MPEG-4 framework has been defined based on this principle: a scene gives provision for multimedia enhancements.

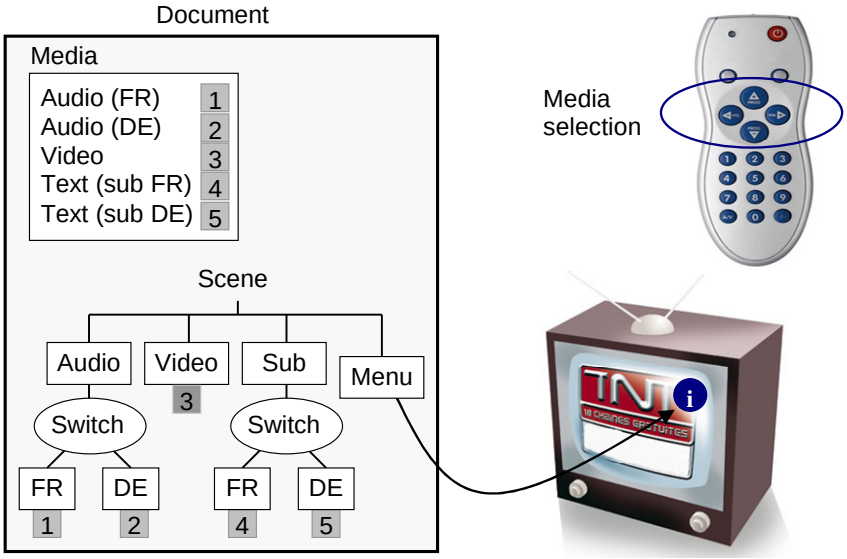


Figure 10: Example of a DTV presentation based on a scene.

3.1.2 Media components

A multimedia scene does not necessarily control all the details of the presentation. In particular, a presentation can be described by multiple and independent sub-scenes that are included into one master scene. In that case, the main scene can have full control over these sub-scenes (option 1), a limited control defined through an explicit control interface (option 2) or no control over them (option 3). In the same way, some elementary media that are part of a presentation may have an inherent organization or may enable a limited control. For instance, a video sequence being broadcast is composed of images whose presentation cannot be easily controlled from an external multimedia scene (option 3). However, a multimedia scene may have some control over a video sequence by controlling its playback speed or seeking into the video timeline (option 2). When considering graphic elements, such as rectangles, the multimedia scene usually has the full control over these elementary media to define their border styles or colors (option 1). In order to cope with such diversity, we do not make any difference between sub-scenes and elementary media in our terminology: we define as *media component* a document that offers a control interface to an external scene. As illustrated in Figure 11, this interface can be:

- minimal, if it only allows the inclusion of its presentation (as it is) into an external scene (e.g. a non-scalable video sequence in broadcast environments).
- limited, if it proposes a list of parameters that can be changed to modify some aspects of its presentation through scene characteristics or elementary media selection (e.g. the activated audio track in a multi-language environment).
- complete, if it allows an unlimited access to all presentation aspects of the media component (e.g. menu items displayed in overlay over the video).

3.1.3 Scene description

Multimedia scenes need to be described so as to be interpreted by a player and visualized by a user. A *scene description* can be defined as the concrete expression of a presentation that can be included in a multimedia bitstream (document or service). Its playback leads to a deterministic process that offers a multimedia experience to the user that should faithfully match the author's intention. Even though a

scene description is supposed to be a machine-to-machine description (edited thanks to the use of authoring tools), most of existing languages can be provided in a human-readable XML syntax.

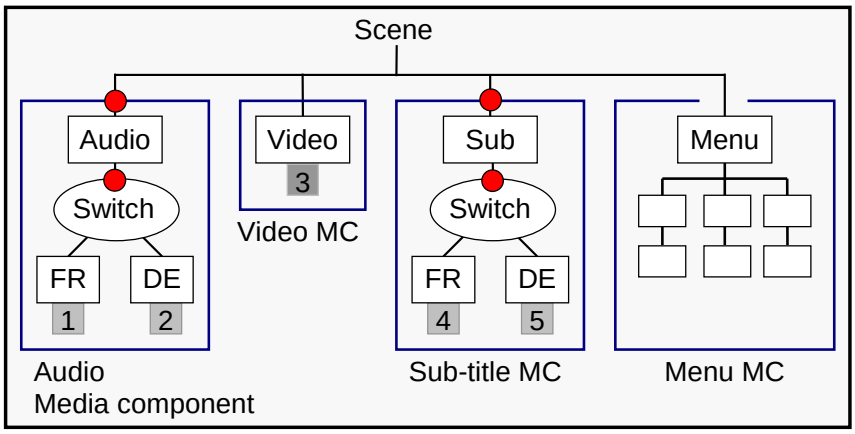


Figure 11: A multimedia scene and its media components.

An essential aspect of multimedia scenes is that there is no one to one mapping between a presentation and a scene description. Two scene descriptions can lead to exactly the same presentation as illustrated in Figure 12. For example, some content generation platforms such as the MM4U framework [101] manipulate a unique multimedia document encompassing a single scene that can be provided into multiple formats. During this transformation phase, called the 'last mile' by A. Scherp, the same multimedia content is transformed into a 'concrete multimedia presentation format' (we call it a scene description) that depends on the targeted playback environment.

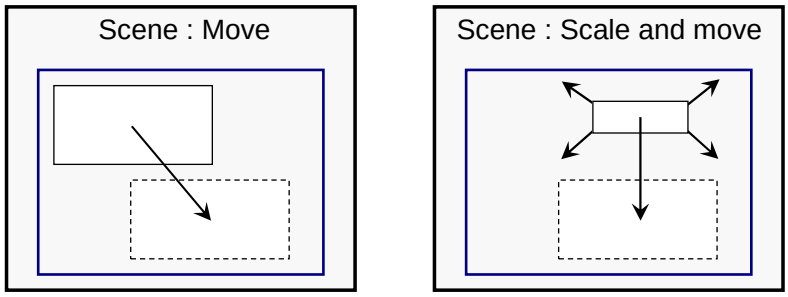


Figure 12: Two scene descriptions leading to the same presentation.

3.1.4 Presentation model, document model and scene description model

If we consider a multimedia presentation as a subject of study, an abstraction of this reality in a model is essential to perform experiments. In that sense, we do not strongly agree with the following statement from S. Boll: "A multimedia document is an instantiation of a multimedia document model that provides the primitives to capture all aspects of a multimedia document" [24]. Instead, we would rather define a multimedia document as the instantiation of a *document model* that captures a presentation (and not a document) as illustrated in Figure 13. Indeed, a document model is in the computer space (close to the machine execution), while the presentation model is in the user space (close to the human concepts). The main goal of such a theoretical *presentation model* consists in capturing the user's experience in front of a multimedia presentation, i.e. a representation of what she/he observes over time and depending on her/his interactions. To put it in a nutshell, a presentation model firstly aims at capturing all aspects of a multimedia presentation. Document models also try to capture these aspects but with additional objectives. For instance, the compactness, the exactness and/or the robustness of the description of the presentation can be additional quality criterions that lead to several multimedia documents models representing the same presentation. In fact, the representation process that defines a multimedia document model from an abstract presentation model is always the result of a trade-off that varies with different application scenarios.

Furthermore, each multimedia document designed according to a document model can be expressed (or serialized) in, at least, one document format with a dedicated syntax (or language) as illustrated in Figure 13. However, a single multimedia document model can be also serialized into several formats and therefore enable multi-format publishing platforms. Our objective is not to define a new document model encompassing all existing presentation models as the Amsterdam Hypermedia Model (AHM) [48] or the ZyX model [24] proposed in the past. Indeed, it is clear that several scene formats will coexist in the future since their underlying document models are relevant for specific applications. Instead, we propose a new model that abstracts and integrates existing scene formats and leverages the low-level functionalities of each scene format.

As a consequence, the presentation capabilities of our model are directly inherited from the document model relative to the scene description being used. For the sake of simplicity, we will name in the rest of this dissertation a "*scene description model*" the document model relative to a scene description. Such a scene description model is not designed for multi-publication. With that definition, the new model we propose (the *Scalable MSTI* model) is a scene description model. In the following, the scene description models relative to the main scene formats are reviewed focusing on presentation aspects [24][71][101]: spatio-visual properties (Section 3.2), temporal aspects (Section 3.3) and interactions (Section 3.4).

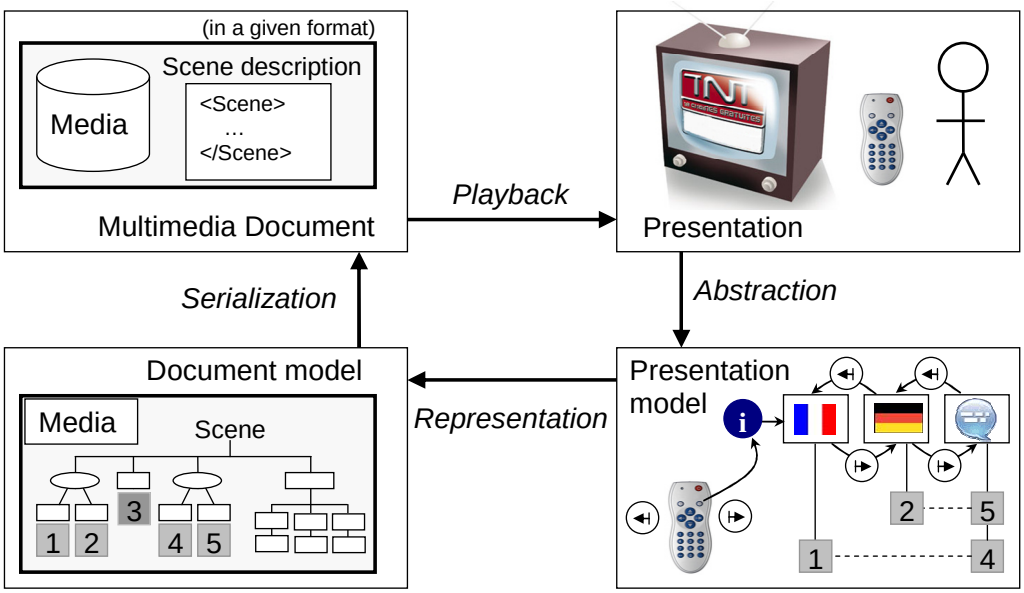


Figure 13: Modeling a multimedia presentation.

3.2 Spatio-visual models

The most obvious aspect that a scene description model must cover is the way the different media are perceived at a given instant from a user perspective. All these perceptual properties of media components at a particular moment are mainly visual (except some audio-related properties) and can therefore be expressed in a visual model that defines the positioning of media components in a 2D (or 3D) environment and their dimensions (Section 3.2.1) or their visibility (Section 3.2.2). Furthermore, their color, their border properties and all kinds of decoration features that aim at faithfully describing the multimedia presentation as created by the author are also visual properties (Section 3.2.3).

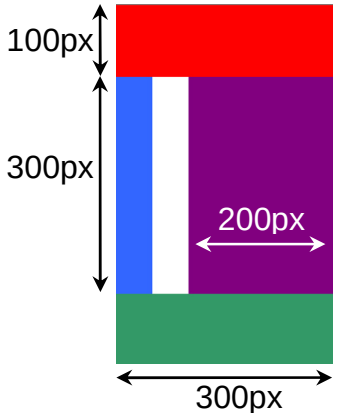
3.2.1 The positioning of media components

When considering the underlying visual model of a multimedia presentation, the layout of the content is a key property that is often the focus of state of the art studies [24][48][79]. In particular, as S. Boll describes it [24], 'the positioning of visual elements in the multimedia presentation can be expressed by the use of a spatial model'. Hence, three approaches can be distinguished: fixed (Section 3.2.1.1), topological (Section 3.2.1.2) and directional (Section 3.2.1.3) positioning.

3.2.1.1 Absolute and relative fixed positioning

In absolute/relative positioning, the scene is in charge of defining the exact placement of each media component by providing a coordinate pair (or triplet) in a 2D (or 3D) space. This positioning layout can be defined according to an absolute reference (absolute positioning) or can be relative to another reference (relative positioning). The definition of the origin of a coordinate system related to a media component is often determined by the scene structure which contributes to the specification of the spatial model in that case.

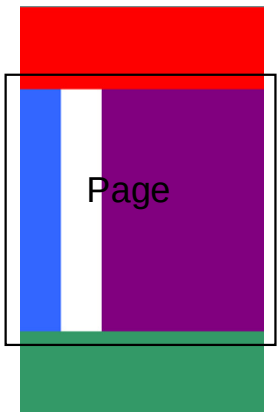
Absolute positioning in HTML can be defined using CSS(`position: absolute`) as illustrated in Code 1. In that case, all media components appear at the exact pixels that are specified using a (`left, top`) coordinate pair for instance.




```
<HTML>
  <HEAD>
    <STYLE type="text/css" media="screen">
      #sidebar {
        position: absolute;
        width: 50px;
        height: 300px;
        top: 100px;
        left: 0;
        background-color: rgb(51,102,255); // blue
      } ...
    </STYLE>
  </HEAD>
  <BODY>
    <DIV id="sidebar"/> ...
  </BODY>
</HTML>
```

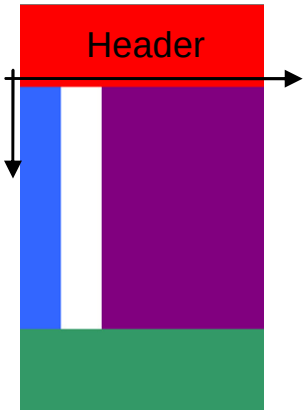
Code 1: Absolute positioning in HTML through CSS.

Relative positioning can also be defined in HTML based on CSS absolute positioning thanks to the XML structure of XTHML documents as illustrated in Code 2. In that case, the 2D space associated with each media component is explicitly defined. A last approach in HTML consists in relying on the vertical flow of `<DIV>` elements (see Section 3.2.1.2). In that case, the 2D space associated with each media component is implicit but can be updated using CSS code (`position: relative`) as illustrated in Code 3.




```
<HTML>
  <HEAD>
    <STYLE type="text/css" media="screen">
      #page {
        position: absolute;
        width: 300px;
        height: 300px;
        top: 100px; ...
      }
     #sidebar{
        position: absolute;
        width: 50px;
        height: 300px;
        top: 0; ...
      } ...
    </STYLE>
  </HEAD>
  <BODY>
    <DIV id="page">
      <DIV id="sidebar"/> ...
    </DIV> ...
  </BODY>
</HTML>
```

Code 2: Relative positioning in HTML through CSS and DIV structure.



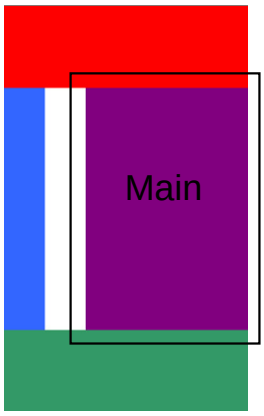

```
<HTML>
  <HEAD><STYLE type="text/css" media="screen">
     #header {
       position: relative;
       width: 300px;
       height: 100px;
       top: 0;
       left: 0; ...
      }
      #sidebar {
        position: relative;
        width: 50px;
        height: 300px;
        top: 0;
        left: 0; ...
      }
    </STYLE></HEAD>
  <BODY>
    <DIV id="header"/>
    <DIV id="sidebar"/> ...
  </BODY>
</HTML>
```

Code 3: Relative positioning in HTML through CSS.

3.2.1.2 Topological positioning

In topological positioning, the scene defines the placement of each media component by specifying the topological relationship between two contiguous media components. The Region Connection Calculus representation (RCC-8) can be used to define eight spatial topological relations [31]. Although these relations proved sufficient to describe all possible combinations of two media components in a one-dimension spatial environment according to authors, topological positioning is not widely used in existing scene formats.

In fact, most multimedia formats assume non-overlapping relations when a layout is dynamically generated. In the HTML format, the vertical positioning of media components relies on a similar *meet* relation (by default). Besides, the HTML flow of `<DIV>` elements can be modified using CSS `(flow:left)` in order to define horizontal *meet* relations as illustrated in Code 4. It should also be noted that *covers* and *covered-by* relations can be specified through the painter's algorithm (also known as priority fill) and becomes visible when relative positioning results in overlapping media components. However, these scene properties only partially cover the functionality that could provide a spatial topological model as implemented in multiple authoring tools [20][79].



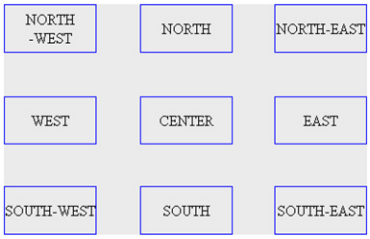

```
<HTML>
  <HEAD>
    <STYLE type="text/css" media="screen">
      #sidebar {
        float: left;
        width: 50px;
        height: 300px; ...
      }
      #main {
        position:relative;
        float: left;
        width: 200px;
        height: 300px;
        left: 50px; ...
        background-color: rgb(128,0,128); // violet
      } ...
    </STYLE>
  </HEAD>
  <BODY> ...
    <DIV id="sidebar"></DIV>
    <DIV id="main"></DIV> ...
  </BODY>
</HTML>
```

Code 4: A simplified 'meet' topological positioning in HTML using an horizontal flow.

3.2.1.3 Directional positioning

In directional positioning, the scene defines the placement of each media with respect to other media components by specifying a directional orientation. A simple modeling distinguishes height directional relations: *north-west*, *north*, *north-east*, *east*, *south-east*, *south*, *south-west* and *west*.

Directional positioning is largely used to define the position of a media component that is contained inside another media component. In HTML, *north-west*, *north-east*, *south-east* or *south-west* relations can be defined in CSS by providing `(top:0; left: 0)`, `(top:0; right: 0)`, `(bottom:0; left: 0)` or `(bottom:0; right: 0)` attributes as illustrated in Code 5. Partially-centered positions defined by *north*, *east*, *south* and *west* directions can also be specified in HTML but requires workarounds such as automated margins related to the actual size of the <`DIV`> element.



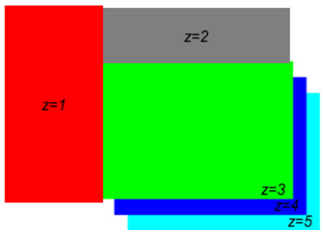

```
<HTML>
  <HEAD>
    <STYLE type="text/css" media="screen">
     .box {
       position: absolute;
       width: 100px;
       height: 50px; ...
      }
      .top-left {top:0; left:0;}
      .top-right {top:0; right:0;}
      .bottom-left {bottom:0; left:0;}
      .bottom-right {bottom:0; right:0;}
      .bottom-mid {bottom:0; left: 50%; margin-left: -50px;}
      .middle-left {left:0; top: 50%; margin-top: -25px;}...
    </STYLE>
  </HEAD>
  <BODY> ...
    <DIV class="box top-left">NORTH -WEST</DIV> ...
  </BODY>
</HTML>
```

Code 5: Directional positioning in HTML through CSS.

3.2.2 The visibility of media components

A multimedia presentation contains media components that are visually proposed to the user over time. However, all media components included in a document at a given time are not necessarily part of the current presentation. Indeed, their visibility may require some actions from the user who is interested in accessing them. Their visibility also may not last for the complete duration of the presentation. The scene is in charge of specifying visibility mechanisms. The scene can rely, for instance, on the topological relations of a spatial model to hide some media components behind some others. As previously explained in Section 3.2.1, media component hiding can be performed by taking advantage of the painter's algorithm. Also, relative depth positions can be described to define a 2D+1/2 space as illustrated in Code 6 by using an MPEG-4 BIFS *`OrderedGroup`*. In that case, the overlapping of media components is not implicitly defined by the structure of the scene but explicitly described by a dedicated structure. A last approach consists in defining an individual depth, named `z-index` property in the scope of CSS, which removes the burden of a dedicated scene structure.

Although these 'hiding' workarounds are very handy and efficient in some cases, specific visibility properties are also defined in spatial models in order to maintain these visibility semantics accessible from the scene. Three main types of visibility properties can be distinguished: visual activation (Section 3.2.2.1), alpha compositing (Section 3.2.2.2) and viewport cropping (Section 3.2.2.3).



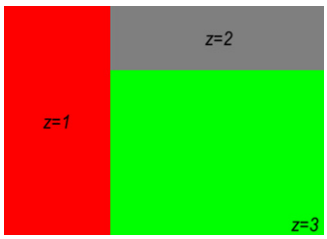

```
<OrderedGroup order="2 1 0"><children> ...
  <ProtoInstance name="PROTO_LIVE">
   <fieldValue name="backColor" colorValue="0 1 0"/>
   <fieldValue name="textField" stringArrayValue="'z=3'"/>
</ProtoInstance>
<Transform2D translation="15 -20"><children>
  <ProtoInstance name="PROTO_LIVE">
   <fieldValue name="backColor" colorValue="0 0 1"/>
   <fieldValue name="textField" stringArrayValue="'z=4'"/>
</ProtoInstance>
</children></Transform2D>
<Transform2D translation="30 -40"><children>
  <ProtoInstance name="PROTO_LIVE">
   <fieldValue name="backColor" colorValue="0 1 1"/>
   <fieldValue name="textField" stringArrayValue="'z=5'"/>
  </ProtoInstance>
</children></Transform2D> ...
</children></OrderedGroup>
```

Code 6: Reversing the painter's algorithm by defining relative depth positions in BIFS

3.2.2.1 Visual activation

A scene can define the visibility of each media component by individually specifying their visibility status or assigning a visibility status to the group of media components they belong to. In that case, either the media component is displayed in the presentation or it cannot be seen by the user at all.

The visual activation of a media component is usually defined using a dedicated element (e.g. the *`Switch`* SMIL element) or specific attributes (e.g. the *`display`* or *`visibility`* SVG attributes) that indicates whether the media component or the group of media component, it is attached to, shall be displayed or not. In the MPEG-4 BIFS format, the `Switch` element can be used to disable the visibility of its children element (`whichChoice='-1'`) but also to visually activate only one of its children using an index (`whichChoice='1'`) as illustrated in Code 7.




```
<Switch whichChoice="1"><choice> ...
<Transform2D translation="15 -20"><children>
  <ProtoInstance name="PROTO_LIVE">
   <fieldValue name="backColor" colorValue="0 0 1"/>
   <fieldValue name="textField" stringArrayValue="'z=4'"/>
</ProtoInstance>
</children></Transform2D>
<Transform2D translation="30 -40"><children>
  <ProtoInstance name="PROTO_LIVE">
   <fieldValue name="backColor" colorValue="0 1 1"/>
   <fieldValue name="textField" stringArrayValue="'z=3'"/>
  </ProtoInstance>
</children></Transform2D> ...
</choice></Switch>
```

Code 7: Visual activation in BIFS.

3.2.2.2 Alpha compositing

A scene can define the visibility of each media component by individually specifying their transparency or assigning a transparency to a group of media components. In that case, visual activation as defined in 3.2.2.1 can be achieved (visible or not visible) but it is also possible to emphasize the visibility of some media components by using partial transparency or by progressively modifying their visibility (e.g. visual fade-in or fade-out).

Although transparency can be used to define the visual activation of media components as illustrated in Code 8 in the MPEG-4 BIFS `DeclareProto`, it is not commonly used for that purpose because the visual activation of a media component goes beyond the fact it is visible or not. Indeed, a player might take advantage of activation mechanisms described in 3.2.2.1 to optimize its processing by caching inactive media components until their activation requires the allocation of an internal memory space for its decoded data. In practice, the total visibility or invisibility of a media component should only be defined using transparency if it is likely to become partially transparent during the presentation.

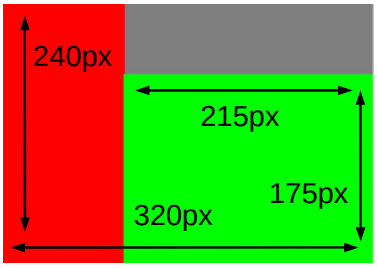

```
...<Shape>
  <appearance><Appearance/><material>
    <Material2D filled="true"
                emissiveColor="0 0 0" transparency="0.8">
    </Material2D>
  </material></appearance>
  <geometry>
    <IndexedFaceSet2D DEF="RTL_LOGO">
      <coord><Coordinate2D point="-102.375 -67.275 -73.575
          -67.275 -73.575 67.275 ..."/></coord>
    </IndexedFaceSet2D>
  </geometry>
</Shape> ...
```

Code 8: Transparency in BIFS.

3.2.2.3 Viewport cropping

A scene can define the visibility of each media component by specifying a visibility window also called viewport. All media components positioned inside this visibility window are displayed while media components laid outside this window are clipped and cannot be seen by the user. If a media component is not entirely inside (or outside) the viewport, the same rules apply and only a part of the media component is displayed. This approach is similar to the management of visibility by media component hiding but cropping operations can be optimized during playback in that case.

Most of the scene formats define a rectangular viewport for their media components. The dimensions of this viewport can be determined on-the-fly based on the computing of bounding boxes of media components. It can also be explicitly specified in the scene during authoring by the editor. Both approaches can even been combined to make a content fit its viewport. A visibility model based on viewport cropping relies on tools such as the `Layer2D` MPEG-4 BIFS element illustrated in Code 9 that do not perform any fitting. A global scene viewport can also be defined by specifying a scene size (`BIFSConfig` element in MPEG-4 BIFS or `viewport` attribute in SVG).




```
...<Shape DEF="GREEN_HEADER"/>
<Shape DEF="RED_SIDEBAR"/>
<Layer2D size="215 175">
  <children>
    <Shape>
      <appearance><Appearance USE="APP_GREEN"></appearance>
      <geometry>
        <Rectangle size="320 240"/>
      </geometry>
    </Shape>
  </children>
</Layer2D>...
```

Code 9: Viewport cropping in BIFS.

3.2.3 Style properties

A multimedia presentation often differs from the simple playback of several elementary media by the attractiveness of their composition and the coherence of their decoration. Style properties can be associated with elementary media and have a significant impact on the perception of the user. On the one hand, style properties are closely related to a specific visual layout, but on the other hand several styles can be associated with the same presentation (and identified as a ‘theme’ by the user). In short, style properties cannot be disconnected from the spatial organization of a presentation, i.e., positioning and visibility of media components, because those aspects are closely linked from a perception point of view as illustrated in Figure 14 on weather forecast examples. For this reason, we define style properties as presentation characteristics: they are part of the spatial model that is in charge of defining the visual properties of a presentation at a given time. Two style modeling approaches can be distinguished: one of them tackles the style of a presentation from media components (Section 3.2.3.1) while the other targets the style of the scene as a whole (Section 3.2.3.2).

Figure 14: Examples of weather forecast presentations with three different styles.

3.2.3.1 Media components style

The style properties of a presentation can be modeled using a bottom-up approach: style properties are attached to each media component in order to build the style of the complete presentation. As a consequence, style properties can be applied to media components without taking into account the scene they have to be included to. Of course, these style properties need to be coordinated during the authoring phase in order to end up with a harmonized design of the content, but this is not explicit in the style description.

Such styling model requires media components that offer some control over their style. These style parameters can be associated with an elementary media using a CSS style sheet targeting specific scene elements. As illustrated in Code 10, the font style of text elements is configured by CSS through the "menu" `<g>` element and a blurred JPEG image is assigned to "logo" `<rect>` element using some lines of JavaScript code.

Some state of the art approaches rely on media-based style model for their visual properties. They usually define dedicated tools for this purpose such as the *channel* concept of the Amsterdam Hypermedia Model (AHM) [48]. This kind of approach favors style reuse across different media components. However, the style properties of media components are always defined in an independent way even if they belong to the same presentation. In our menu bar example, some implicit layout rules need to be kept while authoring a presentation from styled media components. For instance, the number of menu items and their width implicitly drives the text font size. It needs to be maintained in that case. In the same way, the location of the "logo" image and its size clearly depend on the global layout of the scene. These parameters must remain inside the "logo" area if style decisions are only operated at the media-level. These restrictions are illustrated in Figure 15 where depicted menu bars look similar because of the limitations a media-level styling.


```
// inside the referenced CSS file
#menu {
  font-size:15pt;
  font-family:Arial;
  font-weight: bold;
  fill: white;
}...
// inside the referenced JS file
function apply_media_style(evt){ var imageLogo =
{url:'./menu_style/logo_RTL.jpg',
 width:80, height:56, xPos: 10, yPos: 8};
var svgdoc = evt.target.ownerDocument;
setImageStyle(svgdoc, 'logo', imageLogo);
}...
<svg onload="apply_media_style(evt)"> ...
  <rect id="logo" width="106" height="90"/> // blurred area
  <g id="menu" transform="translate(0,120)">
    <text>le direct</text>
  </g>
</svg>
```

Code 10: Styled media components in SVG using CSS and JavaScript.

Figure 15: Various menu bar examples using the same scene style.

3.2.3.2 Scene style

The style properties of a presentation can be modeled using a top-down approach: style properties are managed by a single choreographer that is also responsible for the organization of the presentation, i.e. the scene. In this approach, style properties are defined along with media components and therefore encompass all aspects of a multimedia scene.

Contrary to the styling approach described in 3.2.3.1, a scene-based approach creates a styled scene integrating media components that inherit some of these style properties. As far as CSS or JavaScript are concerned, it does not make much difference since they both have a global scope inside an SVG document. However, it does make a difference from a presentation modeling point of view. For instance, the width of the “menu” rectangle in our example can be defined according to the largest text width and its selected font style. As illustrated in Figure 14, the "logo" image can be positioned according to the whole scene layout. In practice, scene-based style modeling remains compatible with a styling based on media components as long as the scope of media-based style properties remains limited (the color of a font for instance).

In some cases, the difference between style properties and other visual properties might be difficult to distinguish because the style properties of the presentation have been customized to the presentation during the authoring phase. This ambiguity might raise issues in models allowing the authoring of several styles for a single content. However, this can be handled using alternative scene versions (instead of media-based style alternatives). In practice, such scene versions can be configured using `alternate` CSS stylesheet or created from scratch using JavaScript code.

3.3 Temporal models

A scene description model is in charge of representing the audiovisual experience of a user exposed to a set of elementary media during a period of time. A visual model captures a snapshot of a presentation that does not take into account the choreography of the course of the multimedia playback. All these changes in the perceptual properties of media components can be expressed in a temporal model that defines the presence of media components over time (Section 3.3.1), their synchronization (Section 3.3.2) and the timing of the presentation (Section 3.3.3).

3.3.1 The presence of media components

Multimedia models, such as ZYX [24], AHM [48] or Madeus [79], define their temporal model as a description of the 'temporal dependencies between the media elements of a multimedia document' [24]. These temporal dependencies refer to the presence of media components in the presentation which usually results in defining visibility properties of media components over time. These dependencies require a set of relations that can be defined based on intervals using the Allen algebra [16]. In that case, all possible dependencies between two different media components are defined using height temporal relations: *before*, *meets*, *overlaps*, *starts*, *during*, *finishes* and *equals* illustrated in Figure 16.

Although the Allen temporal model is flexible, these temporal relations between two media components are not accurate enough to define a deterministic presentation. For this reason, three temporal models can be derived from the Allen model and strengthen typical use cases by focusing on interval-based sequences (Section 3.3.1.1), interval-based timelines (Section 3.3.1.2) and point-based timelines (Section 3.3.1.3).

Relation	Illustration	Interpretation
X < Y Y > X	X Y	X takes place for Y
X m Y Y mi X	X Y	X meet Y (*i* stands for **i**nverse)
X o Y Y oi X	X Y	X overlaps with Y
X s Y Y si X	X Y	X starts Y
X d X Y di X	X Y	X during Y
X f Y Y fi X	X Y	X finishes Y
Y = X	X Y	X is equal to Y

Figure 16: Allen temporal relations between media components.

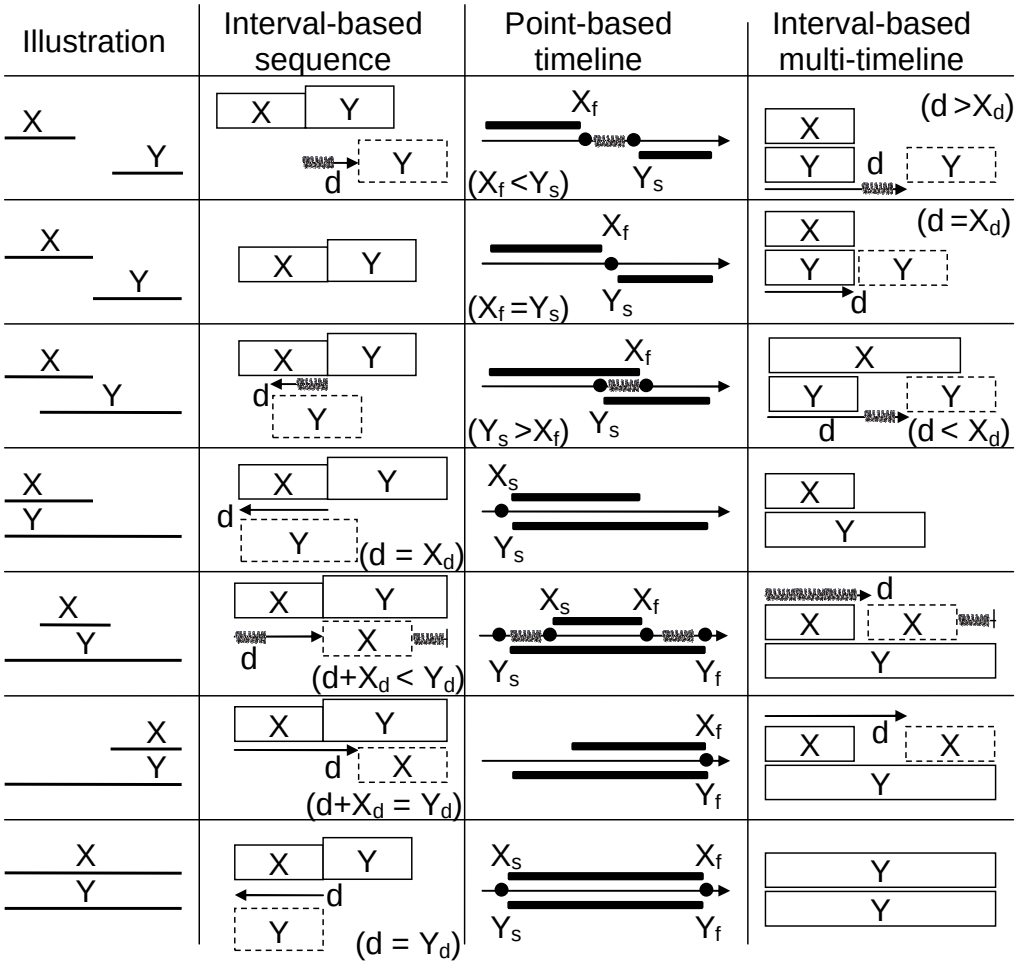


Figure 17: Three media presence models mapped on Allen temporal relations

3.3.1.1 Interval-based sequence

When considering off-line authoring, media components can be defined as intervals to be cascaded onto a time axis: their presence in the presentation is determined by a start time and a duration (or an end time). A temporal model can consist in defining the course of a presentation as a sequence of media components. In that case, the start time of an interval corresponding to a media component does not even have to be specified since it can be determined from the duration of the previous media component and from the time it started.

Since interval-based sequences define the presentation of media components over time, the structure of multimedia documents can be used to describe such temporal scene properties. For instance, the body of a SMIL document can be composed of a `<seq>` element which defines a sequence of elementary media with durations (`dur` attribute for `<img>`) as illustrated in Code 11. The sequence of media components can also be managed by the use of delays (`begin` attribute in SMIL timing) or customized by introducing empty intervals for live programs (`dur` attribute for `<seq>` elements) as illustrated in Code 12. All Allen relations can be explicitly defined using this interval-based sequence model (by using positive or negative `begin` attributes) as illustrated in Figure 17. However, it can be noticed that this model is more appropriate for *before*, *meets* and *overlaps* relations.

In practice, the interval-based sequence approach is flexible for off-line authoring since the removal (or insertion) of one media component from (or into) the sequence will be automatically managed by computing again all dependencies. It can also be used for live services by reserving time slots in the interval sequence. In that case, a dynamic computation of content planning needs to be implemented on the player's side.


```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
  <head>
    <layout> <root-layout width="320" height="240" />
    <region id="live" top="50" left="12" width="215" height="185"/>
    </layout>
  </head>
<body>
<par>
 <audio src="funRadio.mp3"/>
 <seq>
   <img src="fun1.jpg"  dur="20s" region="live" fill="freeze"/>
   <img src="fun2.jpg"  dur="16s" region="live" fill="freeze"/>
   <img src="fun3.jpg"  dur="7s"  region="live" fill="freeze"/>
   <img src="fun4.jpg"  dur="19s" region="live" fill="freeze"/>
   <img src="fun5.jpg"  dur="12s" region="live" fill="freeze"/>
   <img src="fun6.jpg"  dur="16s" region="live" fill="freeze"/>
   <img src="fun7.jpg"  dur="15s" region="live" fill="freeze"/>
   <img src="fun8.jpg"  dur="5s"  region="live" fill="freeze"/>
   <img src="fun9.jpg"  dur="4s"  region="live" fill="freeze"/>
   <img src="fun10.jpg" dur="8s" region="live" fill="freeze"/>
 </seq>
<par>
</body>
</smil>
```

Code 11: Interval-based sequence in SMIL.

```
<seq dur="122s">
  <img src="fun1.jpg" dur="20s" region="live" fill="remove"/>
  <img src="fun2.jpg" dur="16s" region="live" fill="remove"/>
  <img src="fun3.jpg" dur="7s" region="live" fill="remove"/>
  <img src="fun4.jpg" dur="19s" region="live" fill="remove"/>
  <img src="fun5.jpg" dur="12s" region="live" fill="remove"/>
  <seq dur="40s">
    <img src="fun6.jpg" begin="0s" region="live" fill="freeze"/>
    <img src="fun7.jpg" begin="16s" region="live" fill="freeze"/>
    <img src="fun8.jpg" begin="15s" region="live" fill="freeze"/>
    <img src="fun9.jpg" begin="5s" region="live" fill="freeze"/>
  </seq>
  <img src="./media/fun10.jpg" begin="0s" region="live"
fill="remove"/>
</seq>
```

Code 12: Interval-based sequence with delays in SMIL.

3.3.1.2 Interval-based multi-timeline

A presentation does not have to be restricted to a sequence of media components. A part of a presentation can be efficiently managed by an interval-based sequence model because it is composed of cascading media components but others parts may be designed to appear together on the display. As a consequence, a temporal model can consist in defining the course of a presentation as media components to be displayed in parallel. In that case, only the start time of an interval corresponding to a media component has to be specified since the visibility of the media component can expire when its duration time is reached.

An interval-based multi-timeline defines the visibility of media components by referring to a common timeline which includes an initial time reference and a clock. For instance, a SMIL document can declare a `<par>` element which describes a list of elementary media (`<img>`) with durations (`dur` attribute for `<img>`) to be displayed in parallel according to their starting time (`begin`) as illustrated in Code 13. All Allen relations can be explicitly defined using this interval-based timeline model as illustrated in Figure 17. However, it can be noticed this model is more appropriate for *starts*, *during* and *overlaps* relations.

In practice, the interval-based multi-timeline model is commonly combined with the interval-based sequence model to address a large set of use cases with the most suitable approach. Although multimedia documents relying on such models can be fed with live media components, the temporal properties of the presentation suffer the fact that the visibility of media components is managed by a temporal structure whose playback cannot be easily updated from a server which usually has no information about the

player's current status. Therefore, such interval-based models are suitable for off-line authoring but raise severe issues when used in live conditions.

```
<par>
  <audio src="funRadio.mp3"/>
  <img src="fun1.jpg" begin="0s" dur="20s" region="live" fill="freeze" />
  <img src="fun2.jpg" begin="20s" dur="16s" region="live" fill="freeze" />
  <img src="fun3.jpg" begin="36s" dur="7s" region="live" fill="freeze" />
  <img src="fun4.jpg" begin="43s" dur="19s" region="live" fill="freeze" />
  <img src="fun5.jpg" begin="62s" dur="12s" region="live" fill="freeze" />
  <img src="fun6.jpg" begin="74s" dur="16s" region="live" fill="freeze" />
  <img src="fun7.jpg" begin="90s" dur="15s" region="live" fill="freeze" />
  <img src="fun8.jpg" begin="105s" dur="5s" region="live" fill="freeze" />
  <img src="fun9.jpg" begin="110" dur="4s" region="live" fill="freeze" />
  <img src="fun10.jpg" begin="114s" dur="8s" region="live" fill="freeze" />
</par>
```

Code 13: Interval-based multi-timeline in SMIL.

3.3.1.3 Point-based timeline

Contrary to the interval-based timing models which require the duration of media components to generate the timeline of a presentation, a point-based timeline considers media components individually as instantaneous events to be positioned onto a time axis. In that case, each point (or timestamp) drives the activation of the visibility of one or several media components independently from the other media components of the scene. The duration of the visibility of a media component can either be defined from the activation of another media component (scene replacement) or by introducing a new point on the timeline which explicitly consists in deactivating its visibility (scene deletion).

Although the point-based timeline approach allows a multimedia player to generate a presentation without computing relations between media components, it relies on an implicit temporal model which has been used to define the temporal dependencies between media components (during content authoring). This point-based temporal model can be based on an Allen algebra limited to *before* and *starts* relations in case of replacement commands. For instance, a sequence of images can be generated using MPEG-4 BIFS by replacing images (`<Replace>` command) when their display timestamp is reached (*begin* attribute of the `<par>` elements) as illustrated in Code 14. The complete Allen algebra can also be expressed with independent insertion and deletion commands for media components as illustrated in Figure 17. Such advanced management of the presence of media components can be used to enable transitions between images by introducing, for instance, a new image (`<Insert>` command) before deleting (`<Delete>` command) the old image a few second later as illustrated in Code 15.

In practice, the point-based timeline approach is usually applied for services delivered by streaming technologies that are based on timestamps (RTP or MPEG-2 TS for instance). In that case, the playback of the scene is managed in the same way as the playback of an audio or video sequence: a buffer is used to temporarily record and then render chunks of scene at their required timestamps. It should be noticed that a point-based temporal model does not necessarily requires a multimedia format based on timestamps as in the previous MPEG-4 BIFS examples (or MPEG-4 LASeR [62]). Any command mechanism, such as XMLHttpRequest [131], can be used to modify an XML document upon reception. A typical example is the Remote Events for XML [124] that can be used to modify SVG documents and manage the presence of media components over time.

3.3.2 The synchronization of media components

In this section, the term synchronization refers to the mechanisms which ensure that two or more media components will be jointly displayed at the time it was specified during authoring. A simplistic synchronization would consist in specifying that the presence of two media components will start at the same time. However, such synchronization approach assumes a linear playback where media component data is always immediately accessible or cannot be partially paused, fast forwarded or sought by the user. All these additional constraints concerning the temporal properties of the presentation can be expressed in two models: timeline (Section 3.3.2.1) and event-based synchronization models (Section 3.3.2.2).

```
<Replace><Scene>
  <OrderedGroup><children>
    <Transform2D DEF="T2D"><children>
      <ProtoInstance name="LIVE_IMAGE">
        <fieldValue name="imageURL" stringArrayValue="fun1.jpg"/>
      </ProtoInstance>
    </children></Transform2D>
    <Sound2D>
      <source><AudioSource url="funRadio.mp3"/></source>
    </Sound2D>
  </children></OrderedGroup>
</Scene></Replace>

<!-- Note that the image url could also be updated but broadcast scenarios require few
dependencies between timed scene elements due to potential packet losses -->
<par begin="20">
  <Replace atNode="T2D" atField="children" position="BEGIN">
    <ProtoInstance name="LIVE_IMAGE">
      <fieldValue name="imageURL" stringArrayValue="fun2.jpg"/>
    </ProtoInstance>
  </Replace>
</par>

<par begin="36">
  <Replace atNode="T2D" atField="children" position="BEGIN">
    <ProtoInstance name="LIVE_IMAGE">
      <fieldValue name="imageURL" stringArrayValue="fun3.jpg"/>
    </ProtoInstance>
  </Replace>
</par>
```

Code 14: Point-based timeline in BIFS based on the replace command.

```
<par begin="20">
  <Insert atNode="T2D" atField="children" position="END">
    <ProtoInstance name="LIVE_IMAGE">
      <fieldValue name="imageURL" stringArrayValue="fun2.jpg"/>
    </ProtoInstance>
  </Insert>
</par>
<par begin="21">
  <Delete atNode="T2D" atField="children" position="BEGIN"/>
</par>

<par begin="36">
  <Insert atNode="T2D" atField="children" position="END">
    <ProtoInstance name="LIVE_IMAGE">
      <fieldValue name="imageURL" stringArrayValue="fun3.jpg"/>
    </ProtoInstance>
  </Insert>
</par>
<par begin="37">
  <Delete atNode="T2D" atField="children" position="BEGIN"/>
</par>
```

Code 15: Point-based timeline in BIFS based on the insert/delete commands.

3.3.2.1 Timeline-based synchronization

The synchronization of media components can be achieved by relying on a temporal model based on a unified timeline. As illustrated in Figure 17, the presence of media components can always be interpreted as a point-based timeline. Such a timeline constitutes a common reference during playback that can be used to synchronize media components. Hence, if the data corresponding to a media component cannot be displayed on time, all other media components synchronized on the same timeline will be postponed until all data attached to this time code can be displayed. The same synchronization principles can also

be used when seeking into a presentation: first, the new point in the timeline is determined and then media components are displayed according to that point. In order to ease synchronization of the content when seeking, Random Access Point (RAP) can be defined during authoring in order to improve user experience by immediately displaying an up-to-date content while maintaining a good coding efficiency. For instance, temporal prediction can be used on video sequence for non-RAP frames.

A synchronization model based on a timeline can be explicit. For instance, the point-based timeline model of MPEG-4 BIFS commands can be coupled to the timeline of an elementary media by pointing to the same Object Clock Reference (OCR) in the MPEG-4 framework. In that case, the user can seek temporally in the content by targeting a timestamp and visualize the presentation as if it was played from the beginning until that point.

A timeline-based synchronization model can also be implicit. For instance, the timing of an interval-based document based on `<seq>` and `<par>` in SMIL can be computed to build a playback timeline. In that case, the user can seek temporally in the presentation by targeting a percentage of its total duration. Additionally, the advanced synchronization of media components can be specified using dedicated tools which guide the implicit timeline synchronization: `syncBehavior`, `syncTolerance` and `syncMaster` attributes of SMIL timing.

The timeline-based synchronization of media components during playback consists in constantly checking the alignment of their timelines. Once configured by the editor, this process is automated but still depends on external characteristics such as the playback tempo of media components, temporal cropping or pre-fetching strategies that take into account buffering and decoding delays.

3.3.2.2 Event-based synchronization

The timeline-based synchronization of media components requires a deep control over the timing of all media components which is not always feasible. For instance, a multimedia production chain might be informed when an audio segment is started or stopped but does not necessarily have constantly access to the timeline of the audio stream. In that case, the synchronization of media components can still be achieved by relying on a temporal model based on events notifications. Contrary to timeline-based synchronization model which guarantees the synchronization of media components at any time, event-based model only ensure this synchronization at key moments of the presentation, i.e. at transitions between media components. Hence, an event-based synchronization model can specify the beginning of the presence of two (or more) media components by triggering a common event and can disable them by triggering individual events.

The synchronization of the media components can be achieved by focusing on key moments of a presentation by relying on event listeners. Such events listeners are configured to catch all notifications that correspond to media components becoming visible or disappearing from the current presentation. In some multimedia formats, event-based and timeline-based synchronization model can be advantageously combined in the same descriptors. For instance, the SMIL `begin` attribute, introduced in Section 3.3.1.1 as an explicit time reference, can also refer to the beginning or to the end of a media component as illustrated in Code 16. In this example, the cascading of media components is managed by triggering the beginning of a media component from the ending event of its predecessor. The same cascading scenario can also applied to point-based timeline. In that case, begin and end events can be triggered from timed commands as illustrated in the MPEG-4 BIFS example provided in Code 17.

The event-based synchronization is applicable to presentations whose timeline can be decomposed into segments where individual media components are visible. In particular, such synchronization modeling is not suitable for the synchronization of two (or more) elementary media streams such as audio and video because a lip-sync synchronization is required in that case. However, these two approaches successfully complement each other in many application scenarios.

```
<par>
  <audio src="funRadio.mp3"/>
  <img id="fun1" src="fun1.jpg" begin="0s" dur="20s" region="live" fill="freeze"/>
  <img id="fun2" src="fun2.jpg" begin="fun1.endEvent" dur="16s" region="live" fill="freeze"/>
  <img id="fun3" src="fun3.jpg" begin="fun2.endEvent" dur="7s" region="live" fill="freeze"/>
  <img id="fun4" src="fun4.jpg" begin="fun3.endEvent" dur="19s" region="live" fill="freeze"/>
  <img id="fun5" src="fun5.jpg" begin="fun4.endEvent" dur="12s" region="live" fill="freeze"/>
  <img id="fun6" src="fun6.jpg" begin="fun5.endEvent" dur="16s" region="live" fill="freeze"/>
  <img id="fun7" src="fun7.jpg" begin="fun6.endEvent" dur="15s" region="live" fill="freeze"/>
  <img id="fun8" src="fun8.jpg" begin="fun7.endEvent" dur="5s" region="live" fill="freeze"/>
  <img id="fun9" src="fun9.jpg" begin="fun8.endEvent" dur="4s" region="live" fill="freeze"/>
  <img id="fun10" src="fun10.jpg" begin="fun9.endEvent" dur="8s" region="live" fill="freeze"/>
</par>
```

Code 16: Event-based synchronization in SMIL.

```
<Replace><Scene><OrderedGroup><children>
  <Sound2D><source><AudioSource url="funRadio.mp3"/></source></Sound2D>
  <Transform2D DEF="T2D"><children>
    <ProtoInstance name="LIVE_IMAGE">
      <fieldValue name="imageURL" stringArrayValue="fun1.jpg"/>
    </ProtoInstance>
  </children></Transform2D>

  <Conditional DEF="ACTIVATE_FUN2"><buffer>
    <Insert atNode="T2D" atField="children" position="END">
      <ProtoInstance name="LIVE_IMAGE">
        <fieldValue name="imageURL" stringArrayValue="fun2.jpg"/>
      </ProtoInstance>
    </Insert>
    <Delete atNode="T2D" atField="children" position="BEGIN"/>
  </buffer></Conditional>

  <Conditional DEF="ACTIVATE_FUN3"><buffer>
    <Insert atNode="T2D" atField="children" position="END">
      <ProtoInstance name="LIVE_IMAGE">
        <fieldValue name="imageURL" stringArrayValue="fun3.jpg"/>
      </ProtoInstance>
    </Insert>
    <Delete atNode="T2D" atField="children" position="BEGIN"/>
  </buffer></Conditional>
</children></OrderedGroup></Scene></Replace>

<par begin="20">
  <Replace atNode="ACTIVATE_FUN2" atField="activate" value="true"/>
</par>

<par begin="36">
  <Replace atNode="ACTIVATE_FUN3" atField="activate" value="true"/>
</par>
```

Code 17: Event-based synchronization in BIFS.

3.3.3 The timing of the scene

The temporal properties of a multimedia presentation cannot be limited to the presence of media components and their synchronization. Indeed, all spatial properties introduced in Section 3.2 can be changed over time: positioning, visibility and style properties. As a consequence, a multimedia scene describes timing properties that aim at animating some aspects of the presentation: scrolling text, sliding pictures, blinking banners, transitions, fading and magnifying effects, etc. All these temporal properties can be modeled using two different approaches: timed properties (Section 3.3.3.1) and animations (Section 3.3.3.2).

3.3.3.1 Timed properties

The changes over time of the spatial aspects of the presentation (or interactive aspects) can be specified by assigning a duration to the value of some scene properties. Hence, the timing of all scene properties can be managed by an interval-based or point-based timing model in the same way as the presence of media components over time is handled. Such a model relying on timed properties extends temporal models focused on media components because it defines their visibility over time as any other presentation property that can be timed.

The definition of timed properties requires identifying targeted presentation aspects of the spatial (or interactive) model, defining a timing period for them (possibly infinite) and specifying a value that will be assigned to each targeted property during its activation period. As illustrated in Code 18, the activation period of a SMIL property can be defined by an interval-based sequence with `<set>` elements. In this example, the position (`attributeName`) of each image is defined over time (`to` attribute) so that they become visible. As illustrated in Code 19, the same example can be replicated in MPEG-4 BIFS with the same scene timing model by using a point-based timeline and visual activation mechanisms (`Switch`).

In fact, the presence models for media components presented in Section 3.3.1 and some aspects of synchronization are simplified timing models which only take into account a limited aspect of a multimedia presentation (i.e. the visual activation of media components). Timed properties can advantageously complement such models or take in charge the whole timing aspects of the presentation depending on the level of functionality supported by targeted multimedia players and application scenarios.

```
<par>
  <audio id="caraudio" src="./media/funRadio.mp3"/>
  <img id="fun1" src="fun1.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun2" src="fun2.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun3" src="fun3.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun4" src="fun4.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun5" src="fun5.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun6" src="fun6.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun7" src="fun7.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun8" src="fun8.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun9" src="fun9.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun10" src="fun10.jpg" left="320" region="live" fill="freeze"/>
  <seq>
    <set targetElement="fun1" attributeName="left" to="0" dur="20s"/>
    <set targetElement="fun2" attributeName="left" to="0" dur="16s"/>
    <set targetElement="fun3" attributeName="left" to="0" dur="7s"/>
    <set targetElement="fun4" attributeName="left" to="0" dur="19s"/>
    <set targetElement="fun5" attributeName="left" to="0" dur="12s"/>
    <set targetElement="fun6" attributeName="left" to="0" dur="16s"/>
    <set targetElement="fun7" attributeName="left" to="0" dur="15s"/>
    <set targetElement="fun8" attributeName="left" to="0" dur="5s"/>
    <set targetElement="fun9" attributeName="left" to="0" dur="4s"/>
    <set targetElement="fun10" attributeName="left" to="0" dur="8s"/>
  </seq>
</par>
```

Code 18: Timed-properties in SMIL.

```
<Replace><Scene><OrderedGroup><children>
  <Sound2D><source><AudioSource url="funRadio.mp3"/></source></Sound2D>
  <Switch DEF="SWITCH_IMAGE" whichChoice="0"><choice>
    <ProtoInstance name="LIVE_IMAGE">
      <fieldValue name="imageURL" stringArrayValue="fun1.jpg"/>
    </ProtoInstance>
    <ProtoInstance name="LIVE_IMAGE">
      <fieldValue name="imageURL" stringArrayValue="fun2.jpg"/>
    </ProtoInstance>
  </choice></Switch>
</children></OrderedGroup></Scene></Replace>

<par begin="20">
  <Replace atNode="SWITCH_IMAGE" atField="whichChoice" value="1"/>
</par>
<par begin="36">
  <Replace atNode="SWITCH_IMAGE" atField="whichChoice" value="2"/>
</par>
```

Code 19: Timed-properties in BIFS.

3.3.3.2 Animations

The quick changes over time of the spatial aspects of the presentation can be defined by assigning a value to targeted scene properties for a short period of time (like in cartoons for instance). However, there are some use cases where the evolution of some scene properties can be determined over time through a mathematical formula. Based on the description of such a formula, the presentation playback consists in computing scene properties every time the rendering requires it to create an animation effect. As a consequence, an animation model defines spatial aspects of the presentation (or interactive aspects) over a period of time by associating a function of time to scene properties.

The management of the temporal aspects of a presentation using an animation model requires the explicit identification of targeted scene properties (`animatable` attributes in SVG) and an animation description that determines the different values of each property over a period of time. For instance, a specific function of time can be created using JavaScript code (`setTimeout` function). Simpler animation models can rely on predefined transformations to define animations such as interpolations. Such an animation model usually defines the duration of the animation and (at least) the initial/final states of the targeted property. As illustrated in Code 20, a SMIL `<animate>` element can be used to define a linear translation of the `left` attribute from 320px to 0px in a period of 2s.

Although any timed properties model could be expressed by an animation model (by specifying the animation as a constant function in that case), the animation model is seldom used as the primary temporal model for a presentation. In fact, the conditions that trigger the animation of some scene properties are often managed using a presence or synchronization temporal model for media components. Indeed, the animation model introduces a mapping between a local timer (or event generator) and values of a scene property. Therefore, an animation model only complement a global temporal model which masters the whole presentation.

```
<par>
  <audio id="caraudio" src="funRadio.mp3"/>
  <img id="fun1" src="fun1.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun2" src="fun2.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun3" src="fun3.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun4" src="fun4.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun5" src="fun5.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun6" src="fun6.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun7" src="fun7.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun8" src="fun8.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun9" src="fun9.jpg" left="320" region="live" fill="freeze"/>
  <img id="fun10" src="fun10.jpg" left="320" region="live" fill="freeze"/>
  <seq>
    <animate targetElement="fun1" attributeName="left" to="0" begin="0s" dur="2s"/>
    <animate targetElement="fun2" attributeName="left" to="0" begin="18s" dur="2s"/>
    <animate targetElement="fun3" attributeName="left" to="0" begin="14s" dur="2s"/>
    <animate targetElement="fun4" attributeName="left" to="0" begin="5s" dur="2s"/>
    <animate targetElement="fun5" attributeName="left" to="0" begin="17s" dur="2s"/>
    <animate targetElement="fun6" attributeName="left" to="0" begin="10s" dur="2s"/>
    <animate targetElement="fun7" attributeName="left" to="0" begin="14s" dur="2s"/>
    <animate targetElement="fun8" attributeName="left" to="0" begin="13s" dur="2s"/>
    <animate targetElement="fun9" attributeName="left" to="0" begin="3s" dur="2s"/>
    <animate targetElement="fun10" attributeName="left" to="0" begin="2s" dur="2s"/>
  </seq>
</par>
```

Code 20: Animations in SMIL.

3.4 Interactive models

A scene description model aims at proposing an audiovisual experience to the user who can influence the course of the presentation by interacting with multimedia content. An interactive model (sometimes called hypermedia model [48]) defines the interface between a human exposed to a set of elementary media and their presentation that can be expressed using the visual and temporal models previously described in Section 3.2 and 3.3. The variety of human interfaces is large because interactive means are numerous (a keypad, a mouse, a touch screen stylus, a finger on a touch screen, voice orders, etc.) and the possible interactive behaviors associated with user's wishes can be very rich. Among all possible

behaviors, an interactive model may offer some control over media components (Section 3.4.1), define a content navigation scheme (Section 3.4.2) or specify a user agent (Section 3.4.3).

3.4.1 The control of media components

The user interactions with a multimedia content often aim at controlling the media components that are part of the presentation. This requires that control parameters over media components are accessible to the user. The targeted properties of these control parameters are the visual and temporal properties of the presentation and could be associated to the 'design interactions [that] influence the visual and audible layout of a presentation' and the 'movie interactions [that] affect the temporal course of the entire presentation' defined by S. Boll [23]. However, such a distinction between design and movie suffers limitations since the design of a presentation can be closely linked to its temporal course. Instead, two main control models for media components are defined in the following depending on the scope of presentation parameters, i.e. external (Section 3.4.1.1) or internal (Section 3.4.1.2) to media components.

3.4.1.1 External control parameters

The control over perceptual properties of media components can be introduced in a presentation by specifying new parameters that override default scene properties. For instance, the default size of a video clip is determined according to the screen definition and the resolution of pictures. However, the user may select a full-screen or thumbnail rendering for his presentation by applying a scaling factor to be applied to the default size. Such control from the user can be modeled with external parameters since this supplemental description is not part of the media component itself but attached to it, as proposed by the ZyX projectors [23].

The control over media components by external parameters covers any perceptual aspect that can be overridden. Hence, the size of media components can be extended or reduced using a scaling factor, their playback speed can be modulated (and possibly paused), their position in a relative or absolute positioning model can be updated, their visibility can be canceled… All these presentation modifiers may be ordered by the user from independent interactive tools that rely on a content analysis (or specific descriptors). For instance, the playback control over an audio sequence can be managed by the presentation using dedicated tools such as the `<MediaControl>` element of MPEG-4 BIFS. Additionally, the playback control over the whole presentation can be handled directly from the multimedia player by using the media control bar. In the same way, a scaling factor might be applied by a web browser based on user preferences or explicitly proposed to the user as part of the multimedia presentation.

The modeling of user interactions through external control parameters is flexible because it introduces an interactive layer that is independent from other presentation aspects. However, this modeling only covers some general properties of media components (size, transparency, playback speed or sensitivity to interaction) and sometimes assumes specific spatial and temporal models. For instance, a media component might be based on a set of colors that an interactive model based on external parameters will be unable to control.

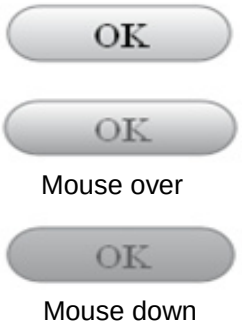
3.4.1.2 Internal control parameters

The control over the presentation of a media component can be offered to the user, by each media component, by exposing an interface that can be used to set the values of targeted parameters inside the component. The parameters of such an interface are specific to each media component but can be expressed using generic interface mechanisms.

A presentation whose interaction model is driven by internal control parameters allows the full control of the media components proposed to the user given that a behavior manager has been associated with the parameters of the exposed interfaces. Code 21 describes a button whose presentation changes when being pressed. This SVG button constitutes a media component whose interface is provided in an associated JavaScript description. Depending on the action of the user, the behavior manager configures the SVG

button through its interface with the required internal parameters. As illustrated in Code 22, such media component interface can also be defined using the `<ProtoDeclare>` MPEG-4 BIFS elements.

The modeling of user interactions through internal control parameters can be used to address specific needs to modify media components based on user action. However, the definition of such content-oriented interfaces requires the authoring of all interactive behaviors since a generic multimedia player might not be able to analyze targeted scene properties. For this reason, the simultaneous usage of internal and external control parameters models enables an advantageous trade-off between the flexibility of external parameters and the richness of interactive scenarios opened by internal parameters.




```
<svg xmlns=http://www.w3.org/2000/svg>
  <script id="script-root" type="text/ecmascript">...
    function setTextButtonColor(color){
      okText = document.getElementById("ok_text");
      okText.setAttribute("fill", color);
    }
    function mouseOver() {
      setTextButtonColor("#636466");
      setGradientColor("rgb(255,255,255)","rgb(157,159,161)";
    }
  </script>

 <g transform="translate(-500, -550)">
  <g onmouseover="mouseOver()" onmouseout="mouseOut()"
     onmousedown="mouseDown()" onmouseup="mouseUp()">
    <path fill="url(#gradient_effect)" d="M610.309,.../>
    <text id="ok_text">OK</text>...
  </g>
 </g>
</svg>
```

Code 21: JavaScript control interface for mouse inputs in SVG.

```
<ProtoDeclare name="BUTTON_PROTO">
  <field name="textColor" type="Color" vrml97Hint="exposedField" colorValue="0 0 0"/>
  <field name="gradientColor" type="Colors" colorArrayValue="0.73 0.74 0.74 1 1 1"/>
   <Transform2D><children>
     <Shape>
       <appearance><Appearance>
         <material>
           <Material2D USE="MAT"/>
          </material>
        <texture>
          <LinearGradient key="0 1" endPoint="0 1">
            <IS>
              <connect nodeField="keyValue" protoField="linearGradientColor"/>
            </IS>
          </LinearGradient>
        </texture>
       </Appearance></appearance>
       <geometry><Curve2D type="1 2 2 1 2 2"><point>
         <Coordinate2D point="-45 20..."/>
       </point></Curve2D></geometry>
     </Shape>
     <Shape>
       <appearance><Appearance><material>
         <Material2D filled="true">
           <IS>
             <connect nodeField="emissiveColor" protoField="textColor"/>
           </IS>
         </Material2D>
       </material></Appearance></appearance>
       <geometry><Text string="OK"></Text></geometry>
     </Shape>
   </children>
  </Transform2D>
</ProtoDeclare>
```

Code 22: Prototype control interface in BIFS.

3.4.2 Navigation schemes

An important aspect of content interactivity is the capacity of the user to select the media components he/she might be interested in. During authoring, a significant effort can be spent on the ergonomic aspects of a presentation to make sure that the user can easily identify all the content he/she can access. Most of the time, content possibilities cannot be proposed as a long catalogue: mainstreams need to be defined (e.g. welcome page), content can be categorized according to it level of details (e.g. headlines leading to detailed articles), content might be suggested according to selected topics (e.g. personalized advertising)… The presentation is in charge of defining the paths that guide the user accessing the proposed media components. As S. Boll says [24], 'this interaction type [navigation interactions] means that the document offers the users the possibility to select one out of many presentation paths'. Such a navigation scheme can be modeled using the two approaches described in the following: directions arcs (Section 3.4.2.1) and a finite state machine (Section 3.4.2.2).

3.4.2.1 Directional arcs

A navigation scheme defines the different paths a user is allowed to follow when interacting with a presentation. These interactive paths proposed to the user aim at providing a customized access to media components depending on topics interests, requested level of details, etc. Building these routes implies editorial choices. These routes can be modeled using a gradual approach where the selection of a media component opens access to one of several other media components. Such directional arcs build up a complete navigation tree that the user can run through to get access to media components.

In practice, each media component needs to be the destination of a directional arc to become accessible by the user. As a consequence, such a navigation scheme can be illustrated using a connected graph where the interactions of the user can be drawn by a path as illustrated in Figure 18 on a menu example. For instance, these directional arcs can be defined by simple HTML `<a>` anchors.

The modeling of presentation interactions using directional arcs focuses on the best possible ways to access a given media component. Once a media component has been selected, the experience of the user needs to be pursued by providing new interactive paths. This can be performed by inviting the user to go back to the main presentation which is then the starting point of any navigation inside the content. Secondary directional arcs can also be defined as the next and previous arrows of an astrology service example. In an HTML browser, directional arcs are usually stored and can be undone in order to step back in the navigation scheme. In practice, directional arcs are quite static structures since they imply one-to-one dependencies between media components. For instance, the dynamic insertion and deletion a media component require repairing broken links.

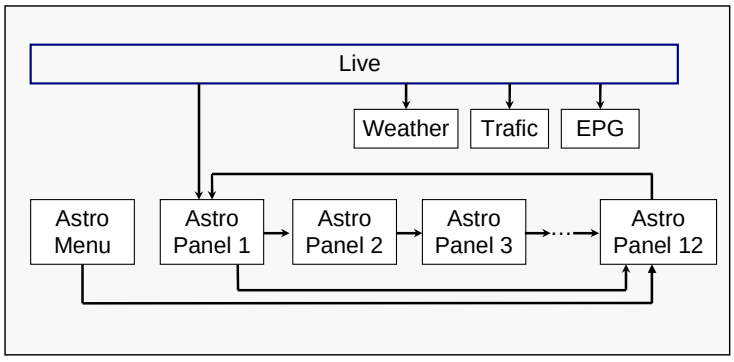


Figure 18: Interactive menu based on direction arcs.

3.4.2.2 Finite state machine

The design of a navigation scheme inside a presentation primary aims at proposing media components that meet the interest of the user. For example, keyword search engines are very efficient when looking for a precise piece of information. In that case, directional arcs can still be used to offer various links to

media components related to queries. However, it then requires numerous scene alternatives to cope with various presentation possibilities due the static nature of these links. Additionally, users are not always expecting a prioritized access to some specific media components while navigating into a presentation: they might be interested in having a multimedia tour to oversee a content dealing with a specific topic or a more general theme. Such navigation scenarios require key presentation states that can be accessed according to some pre-conditions and lead to other states as illustrated in Figure 19. The modeling of a finite state machine can be used to manage the various interactive paths offered to the user.

A navigation scheme modeled by a finite state machine should be seen as a storyboard. This approach has been largely explored in the human interface domain to generate Graphical User Interface (GUI) where the user should fulfil different steps to achieve his task [50]. Each presentation state can be associated with spatial or temporal modifications that materialize the dialog between the user and the presentation. For instance, sub-menu states can be created in order to host dynamic items that do not actually correspond to any specific media component (content placeholders). Based on interactivity managers several presentation states can be created to determine the media component to be displayed according to user actions.

The modeling of presentation interactions using a finite state machine enables a high-level management of user actions. Additionally, the current presentation state of interactivity can be managed at a global level to enable suitable transitions between scene configurations. Of course, transitions from one media component to another can be defined using directional arcs by introducing additional descriptors specifying some animation effects. However, such transition mechanisms heavily depend on the current status of the presentation when applied and not only depend from the media component that triggered the animation. As a consequence, both modeling can be used to efficiently cope with stateless and/or stateful links depending on the required level of control of user actions.

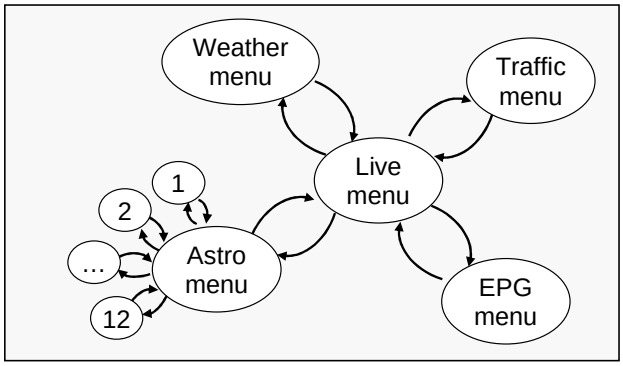


Figure 19: Interactive menu based on a finite state machine.

3.4.3 User inputs

The modeling of the interactive aspects of a presentation aims at defining the behavior of the content according to user actions. Therefore, the possible user's actions need to be specified in order to define the human interface of the presentation. In this domain, quite of a lot of user inputs describing the various actions of a human over a presentation have been standardized such as the MPEG-21 DIA `UserInteractionInput` support. As far as the presentation of media components is concerned only targeted user's actions matter. For this purpose, event-based notifications, as introduced for temporal models in Section 3.3.2.2, are the prevailing modeling approach that can be found in the current state of the art. The supported interactive means can therefore be modeled using two combined descriptions. First, interactive actions being supported are exposed, possibly through event listeners. Then, expected user's actions are translated into triggers that control the properties of some media component (Section 3.4.1) or step into the presentation navigation scheme (Section 3.4.2). In the following, the two main user interaction input models commonly available on multimedia terminals are described by underlying their functionalities that cannot be limited to hyperlink [92]: buttons (Section 3.4.3.1) and pointing devices (Section 3.4.3.2).

3.4.3.1 Button-based interactions

The first interactive means proposed to users to interact with a multimedia presentation were buttons that can be pressed to trigger some actions on the display. Such buttons can be specific to a device like the 'Home' button of an internet tablet (e.g. Nokia N800). A limited set of buttons can be proposed to the user on handheld receivers like the usual keypad quintet composed of the 'right', 'left', 'up', 'down', 'ok' buttons. Finally, a complete AZERTY keyboard can be accessible to the user on a PC or some mobile phones like the Samsung i780. All these keyboards need to be modeled to enable the identification of the buttons activated by the user and also to determine the action that was performed.

The modeling of keyboard buttons that need to be notified to the presentation engine when being pressed is partially standardized by existing character sets [51] that provide a mapping between a numerical representation and printable characters such as '@' or 'L' as illustrated with the `accessKey` of the SVG example described in Code 23 (decimal code is #&76 for 'L' character). Additionally, non-printing characters such as a 'down' action triggered by pressing on the down arrow button of a keyboard can also be caught using the ASCII code characters as illustrated with the `<InputSensor>` element of the MPEG-4 BIFS example described in Code 24 (decimal code is #&18 for the down arrow). Finally, these key identifiers can be complemented with a categorization of the user action. For instance, the `<InputSensor>` element of MPEG-4 BIFS distinguishes press and release commands and can notify the presentation of the combined used of 'alt', 'shift' or 'control' buttons and an additional key.

The modeling of user actions on keyboard buttons heavily relies on the standardized mapping of the device buttons with respect to the ASCII or any extended character set. The association of key events and presentation behaviors is mostly based on the same mechanisms that are used to modify the visual properties over time. In fact, both mechanisms can even be used to trigger the same behavior or combined into a single event. For instance, the main menu of a presentation could become immediately visible by pressing the 'Q' character on a keyboard (or a 'Home' button) while this menu can also be automatically displayed after a certain period of inactivity. In a similar way, keyboard events might be triggered a few seconds after pressing a button as illustrated in Code 23.

3.4.3.2 Focus-based interactions

A usual interactive means proposed to users to interact with a multimedia presentation is a pointing device locating an area of the visual presentation. For instance, such pointing devices can be virtualized on a screen by a mouse. It can also be the real finger of the user (or a stylus) in the case of touch screens. The pointing actions that need to be notified to the presentation engine are closely linked to the media component the user is supposed to interact with. As a consequence, the modeling of focus-based user interaction inputs is mostly based on sensors that can be attached to media components whose surface then becomes sensitive to user's selections. A focus-based interaction model aims at identifying the media components pointed by the user and specifying the triggered actions.


```
<svg xmlns=http://www.w3.org/2000/svg>
 <defs>
  <image id="myTabImg" width="106" height="29"
         xlink:href="tab.gif"/>
   <rect id="myTab" width="106" height="30" opacity="0"/>
 </defs>
 <use id="menu_tab1" display="none" xlink:href="#myTabImg"/>
 <text>le direct</text>
 <use id="menu_item1" xlink:href="#myTab"/>
 <set xlink:href="#menu_tab1" attributeName="display"
 to="inline" begin="menu_item1.click+1s; accessKey(&#76;)"/>
</svg>
```

Code 23: Button-based user's inputs in SVG.

```
<Conditional DEF="C_UP"><buffer> ... </buffer></Conditional>
<Conditional DEF="C_DOWN"><buffer> ... </buffer></Conditional>
<Valuator DEF="V_PRESS_UP" Offset1="-17" />
<Valuator DEF="V_PRESS_DO" Offset1="-18" />

<InputSensor url="KeySensor">
  <buffer>
    <Replace atField="inSFInt32" atNode="V_KEY_PRESSED" value="0" />
    <Replace atField="inSFInt32" atNode="V_KEY_RELEASED" value="0" />
    <Replace atField="inSFInt32" atNode="V_ACTION_KEY_PRESSED" value="0" />
    <Replace atField="inSFInt32" atNode="V_ACTION_KEY_RELEASED" value="0" />
    <Replace atField="inSFBool" atNode="V_SHIFT_KEY_CHANGED" value="TRUE" />
    <Replace atField="inSFBool" atNode="V_CONTROL_KEY_CHANGED" value="TRUE" />
    <Replace atField="inSFBool" atNode="V_ALT_KEY_CHANGED" value="TRUE" />
  </buffer>
</InputSensor>

<ROUTE fromField="outSFInt32" fromNode="V_ACTION_KEY_PRESSED"
       toField="inSFInt32" toNode="V_PRESS_UP" />
<ROUTE fromField="outSFInt32" fromNode="V_ACTION_KEY_PRESSED"
       toField="inSFInt32" toNode="V_PRESS_DO" />
<ROUTE fromField="outSFBool" fromNode="V_PRESS_UP" toField="reverseActivate" toNode="C_UP" />
<ROUTE fromField="outSFBool" fromNode="V_PRESS_DO" toField="reverseActivate" toNode="C_DOWN"/>
```

Code 24: Button-based user's input in BIFS.

The focus on a media component can be triggered using a pointing device. In that case, pointing actions need to be further categorized by specifying specific events such as `click`, `mousedown`, `mouseup`, `mouseover`, `mousemove`, `mouseout` as defined by SVG standard and linked to a media component to define a particular behavior. As illustrated in Code 25, a `mouseover` event can change the color of a blue rectangle to yellow for one-second period. Additionally, generic focus mechanisms that do not depend on a specific pointing device can also be specified. For instance, the `focusin`, `focusout` and `activate` interactive events as defined by the SVG standards can be used to handle some part of the mouse-like interactivity but also some focus-based actions triggered from keyboard buttons such as the 'tab' (or the 'shift'+'tab') shifting key and 'ENTER' or the "alt+tab" (or the 'shift'+'alt'+'tab') switching key.

The pointing devices used to interact with a presentation are various and evolve quickly over time. The scroll wheel of a mouse is an example. In practice, event handlers need to be extended to be able to recognize new actions applied to a media component that gets a focused action from the user. As an extension, some global interactive events that apply to the whole presentation, and not necessarily focused onto one or several media components, can be defined (presentation-based focus). For instance, the display dimensions set by the user when resizing its HTML browser window or the coordinates of a mouse are global events that can be registered by the presentation. In that case, the behavior associated with fired events can be defined by involved media components that listen to events through an extended scope or managed at the presentation level, possibly through JavaScript.


```
<svg xmlns=http://www.w3.org/2000/svg
  width="600" height="100" viewBox="0 0 100 600" >

  <rect x="10" y="10" width="80" height="55" rx="10" ry="10">
    <set begin="click" attributeName="fill" to="yellow" dur="1s"/>
  </rect>
  <text x="50" y="80" text-anchor="middle" >click</text>

  <rect x="10" y="110" width="80" height="55" rx="10" ry="10" >
    <set begin="mousedown" attributeName="fill" to="yellow" dur="1s"/>
  </rect>
   <text x="150" y="80" text-anchor="middle" >mousedown</text>

  <rect x="10" y="210" width="80" height="55" rx="10" ry="10">
    <set begin="focusin" attributeName="fill" to="yellow" dur="1s"/>
  </rect>
  <text x="50" y="80" text-anchor="middle" >focusin</text>
</svg>
```

Code 25: Focus-based user's inputs in SVG.

3.5 Conclusion

The organization of scene properties into three main aspects (spatio-visual, temporal and interactive properties) gives a structured view of the multimedia experience that can be proposed to users. Depending on application scenarios, a multimedia scene might focus on different scene properties:

- the spatio-visual organization of media components into a single document (e.g. a presentation slideshow or an interactive book)
- the temporal evolution of the presentation through content transitions and animation effects that attract the user's attention or improve the ergonomic aspects of interactivity (e.g. rich-media applications)
- the interactive flexibility of the presentation to access media components but also to play with them (e.g. a content gallery or a game)

The use of scene properties as common abstract concepts for all of these presentation aspects contribute in reducing authoring limitations while minimizing implementation efforts for generic multimedia players. For instance, based on common interactive models, multimedia events can be used to retrieve the following: the dimension of the display window (Section 3.2.2.3), the beginning of the playback of a video (Section 3.3.2.2) and action of the user pressing on a button (Section 3.4.3.2). In the same way, using a common set of tools, the scene structure can be used to define the position of a media component on a grid layout (Section 3.2.1.3), the cascading of media components over time (Section 3.3.1.1) and 'previous-next' navigation in a menu (Section 3.4.2.1). Another example is the stateful management of the current presentation, which is commonly used in HMI methodology. Scene states can encompass topological relations between media components (Section 3.2.1.2), a time reference used to synchronize the display of several media components (Section 3.3.2.1) and the presentation modifications required to navigate to another presentation state (Section 3.4.2.2).

Beside this convergence of presentation tools, the natural evolution of existing multimedia standards to address new requirements by extending existing models does not necessarily tend to a unique solution. Indeed, the requirements of targeted application scenarios are an important factor when selecting the models that will depict a multimedia presentation. For instance, authoring and playback scenarios do not have to deal with the same constraints. Content editing requires some flexibility while multimedia rendering implies an efficient content computation. In the same way, generation and delivery scenarios have different focuses. Data generation requires the maintenance of a high content quality while content delivery may be optimized by giving up some explicit semantics.

As a consequence, each scene description model described in this chapter may be advantageously deployed in some multimedia environments. Some of these models might be combined with limited implementation costs or can complement each other to extend some initial requirements. Multimedia presentation systems must be designed to leverage all of these possibilities. In particular, reducing multimedia richness to enable specific scene processing, such as the adaptation of a presentation, is not an option. The evolution of a multimedia platform has to be considered with a high priority when considering software development, even if presentations design is narrowed to dedicated scenarios at creation time.

As a last concluding remark for this chapter, it must be mentioned that we do not claim to have provided an exhaustive list of scene description models as far as spatial, temporal and interactive dimensions are concerned. Multimedia documents are only limited representations of a presentation and will therefore always evolve over time. We strongly suggest relying on extensible approaches when dealing with multimedia documents, and as you will see, this concluding remark will guide our designs in the following chapters.

Chapter 4 Multimedia scene adaptation

The underlying models of scene descriptions based on the three types of scene properties described in Chapter 2 (viz. visual, temporal and interactive properties) assume that scene instructions will be accurately applied during the rendering process of the document. However, the ideal playback conditions assumed during content conception might not be applicable in some environments, thus degrading the audiovisual experience of the user. Hence, multimedia scene usability cannot be simply guaranteed by a smartly designed and appealing presentation. It also requires a technical compatibility between the multimedia scene and different usage environments [134]. The common approach, which is largely deployed in nowadays authoring chains, consists in designing multimedia presentations that only feature very well supported functionalities in order to guarantee the expected document playback to the user in all environments. In fact, this "one-content-fits-all" strategy is very frustrating for service providers who must author highly constrained multimedia documents to fit the most restricted usage environments. It is also unsatisfactory for end users who cannot access an advanced multimedia experience even though they are appropriately equipped.

Scene adaptation consists in providing a multimedia scene tailored to meet the constraints of the rendering engine in the user's environment. These constraints are commonly described as part of the user's context (or user and network characteristics as defined in MPEG-21 DIA [66]). The scene adaptation processes described in this chapter can be modeled as a two-step process, as depicted in Figure 20 [91]. In the first step, an adaptation decision-taking engine (ADTE) transforms environment constraints into adaptation parameters expressed in the scene domain. In the second step, the bit-steam adaptation engine (BAE) performs scene transformations. Although this schematic representation of scene adaptation may look simplistic, it is not easy to take appropriate adaptation decisions when numerous context aspects need to be handled. Additionally, dynamic context aspects and live documents also require continuous scene transformations that have to be integrated in a real-time adaptation process.

In the following, general considerations about the user's context are discussed in Section 4.1. Section 4.2 describes state-of-the-art adaptation decision-taking approaches while scene transformation techniques commonly implemented in scene formats are described in Section 4.3. These decision-taking approaches and scene transformations mechanisms enable the modification of the multimedia presentation based on scene description models which were previously detailed in Chapter 2.

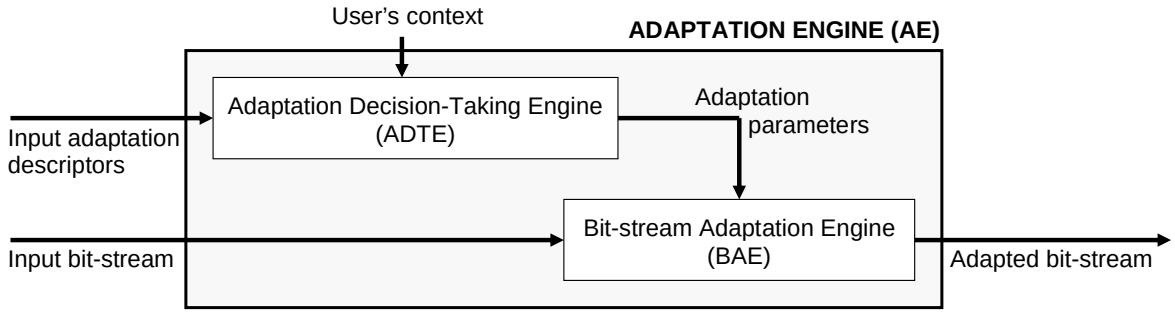


Figure 20: The adaptation engine architecture.

4.1 The user's context

The context parameters that can influence the playback of a presentation can be various and numerous [28][33]. As S. Boll suggests it, 'for the extent of adaptability, we distinguish between adaptation to user interest, which adapts the contents of a document to the user's knowledge, professional background, and the like, and adaptation to technical infrastructure, which adapts to the technical infrastructure available to a user' [24]. In the following, we introduce the scope of our environments constraints (Section 4.1.1) and define user's preferences as we handled it (Section 4.1.2).

4.1.1 Environment constraints

The concrete user context that we consider can be described using standardized approaches such as the *User Environment Descriptors* (MPEG-21 UED) [66] or *Composite Capability / Preference Profiles* (W3C CC/PP) [125]. This user context (or execution environment) is naturally heterogeneous and may fluctuate over time. For instance, the resources offered by wireless networks vary with the number and the position of connected users. Additionally, memory usage, processing power, decoding capabilities, screen size, screen resolution and available interactive means are examples of technical requirements for terminals, also called receivers profile. These technical requirements must be addressed to ensure the access of the multimedia presentation to the end user. Furthermore, other user-oriented context parameters such as the level of details, presentation legibility, content duration or lighting levels can also be exploited to adapt the presentation to user's expectations, also called user profile.

These environment parameters can be described in an MPEG-21 document as provided in Code 26 and constitute adaptation constraints to be respected by scene properties. Since the mapping between scene properties described in Chapter 2 and environment constraints might be a difficult task to perform, as illustrated in Figure 21, environment constraints are sometimes grouped into typical configurations. Hence, device-specific authoring consists in preparing a custom scene description for each targeted environment. In that case, a version of the same presentation is defined for each configuration (e.g. an iPhone). Such authoring approach is efficient in usage environments where receiver's types can be identified. However, it does not scale well as the number of possible device configurations increases. Even though such a high-level context description of the user's environment has limitations, it still enables some essential adaptation scenarios such as the content selection operated by an MPEG Digital Item Processing (DIP) engine [68] or an adaptation framework [101].

Our approach relies on intermediate descriptors in between environment or context parameters and scene properties, that we call adaptation parameters. In fact, adaptation parameters gather scene properties into groups that are targeting specific user contexts. These presentation groups are linked to adaptation parameters which describe content characteristics that will be matched against environment constraints. The `Switch` element of SMIL is an example of such a presentation configuration. For instance, its `systemBitrate` attribute can used to define bandwidth requirements for several audio tracks. In that case, this adaptation parameter refer to an available bandwidth that can be allocated for the playback of an audio track instead of requiring a minimal system bandwidth (as described in Code 26) for the presentation playback.

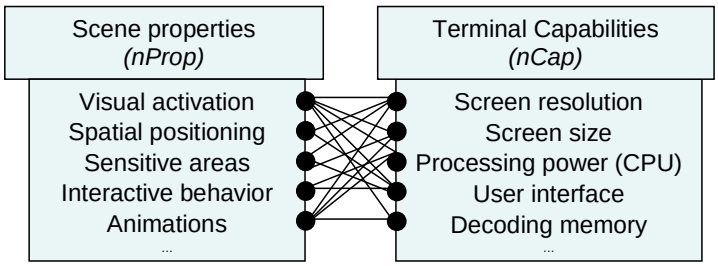


Figure 21: Mapping scene properties to terminal capabilities.

4.1.2 User preferences

As far as user's interest is concerned, we consider author-driven content adaptation according to high-level concepts of user profiles (level of interest, text legibility or available time to watch a timed presentation). Indeed, although user's context descriptors encompass user interests that can be used for the presentation adaptation according to user's preferences for some topics, we do not envisage these personalization use cases due to our broadcast application scenarios. In fact, adaptation to user interests as tackled by A. Scherp in the MM4U framework [101] is out of the scope of this dissertation. Additionally, we do not attempt to model user's behavior as proposed by some researches from the virtual document domain (Digital Virtual Processors) [111][132].

```
<DIA xmlns="urn:mpeg:mpeg21:2003:01-DIA-NS"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001">
  <Description xsi:type="dia:UsageEnvironmentType">
    <UsageEnvironmentProperty xsi:type="dia:NetworksType">
      <Network>
        <NetworkCharacteristic xsi:type="dia:NetworkCapabilityType"
          maxCapacity="384000" minGuaranteed="32000"/>
      </Network>
    </UsageEnvironmentProperty>
    <Description xsi:type="dia:UsageEnvironmentType">
      <UsageEnvironmentProperty xsi:type="dia:TerminalsType">
        <Terminal>
          <TerminalCapability xsi:type="dia:DeviceClassType">
            <DeviceClass href="urn:mpeg:mpeg21:2003:01-DIA-DeviceClassCS-NS:1">
              <mpeg7:Name xml:lang="en">PC</mpeg7:Name>
            </DeviceClass>
          </TerminalCapability>
        </Terminal>
      </UsageEnvironmentProperty>
    </Description>
  </Description>
</DIA>
```

Code 26: MPEG-21 usage environment description.

In our study, we assume that the environment limitations have been made explicit and can be retrieved without any ambiguity. For instance, we do not consider implicit feedback that can be used to adapt multimedia content based on user-centric interests [95]. In such an approach, a user agent tries to guess the context of the user from its interaction with the presentation using 'trial and error' learning or by reinforcing successful decisions by 'cumulating rewards and avoiding fruitless decisions' [107]. Although such a dynamic context might be an efficient solution to overcome the difficulty raised by the explicit statement of the user's context, we decided to first tackle scene adaptation with respect to identified environment constraints and reserve implicit-feedback as a perspective for future work (see Section 7.3.4). As a consequence, we developed our proposal keeping in mind that P. Plesca underlines that "the adaptation approaches proposed in the literature [often] suppose the contextual data is easy to perceive or at least that there is no possible ambiguity to identify the state of the current context." [96].

4.2 Scene adaptation approaches

One of the main challenges when adapting a multimedia scene is to preserve semantics while altering its presentation in order to address the specific constraints of the usage environment. State-of-the-art research work that has been conducted on the adaptation of multimedia presentations is reported in the following. This section is organized according to our classification of existing adaptation decision-taking approaches. This classification aims at underlining some essential functionality of these algorithms which determine the appropriate scene transformation to be applied. This study distinguishes scene adaptation policies in four categories: media-based transformations (Section 4.2.1), custom publishing (Section 4.2.2), alternatives selection (Section 4.2.3) and scene plasticity (Section 4.2.4).

4.2.1 Media-based scene generation

A scene organizes the composition of several elementary media into a coherent presentation. The adaptation of a presentation necessarily requires the adaptation of its elementary media according to the usage environment. The transformation of an elementary media into a content that is compatible with the playback environment of the user is not studied in this dissertation. Instead, we refer to the numerous transcoding or transmoding approaches from the state of the art [34][73][80][108]. The adaptation of these elementary media requires the modification of the multimedia scene in order to integrate these media-level changes. Two main adaptation policies are described in the following: media neutral (Section 4.2.1.1) and media-driven scene adaptation (Section 4.2.1.2).

4.2.1.1 Media-neutral scene adaptation

A multimedia scene can address a wide range of use cases from the simple decoration of a few elementary media (e.g. an audio/video sequence) to the dynamic and interactive applications integrating various elementary media types. In some usage scenarios, content adaptation can primarily focus on some key elementary media because they might require demanding capabilities. The straightforward inclusion of adapted elementary media in the multimedia scene might enable a large set of adaptation scenarios and only require to modify the scene to cope with these converted or transmoded content. Hence, numerous scene adaptation policies [48][83] have been defined to support media references that do not depend on scene properties. For instance, L. Hardman recalls 'the importance of including the resources as a separate layer is that each can be replaced by a similar resource while leaving the document structure itself unchanged.' In what we call the media-neutral approaches, the independence of the document structure from the elementary media it organizes ensures that an appropriate presentation will be generated according to adapted media.

The transformation process required when adapting a multimedia scene following media conversion (or transmoding) basically consists in updating some parts of the presentation. The authoring of media-neutral scene mainly aims at delimiting areas that can be affected by converted media and try to minimize them. For instance, the AHM model [48] defines a spatial layout that is based on independent *channels*. The counterpart of such an adaptation policy is that the main scene basically contains empty placeholders that are to be filled in according to media-level decisions independent from it. For instance, the conversion of a video clip into a sequence of images would benefit from a deep modification of the scene navigation scheme (e.g. to include next and previous buttons) which is not feasible in that case. In practice, these approaches mostly rely on automated layout rules (usually based on high-level structure as defined in SMIL [119], Madeus [79], HTML or highly-templated documents). We do not necessarily rely on such structured documents as defined in [94] because we believe that in carefully designed content the structure of the document should even be updated.

4.2.1.2 Media-driven scene adaptation

The media-neutral adaptation approaches described in 4.2.1.1 can be advantageously extended by taking into account the other elementary media that belong to the presentation. In that case, the adaptation decision-taking is still based on media-level adaptation decisions but these decisions can be influenced by presentation aspects or by the relationships between elementary media. For instance, the adaptability modeling defined in Tiempo [115] relies on media selection groups and quality of service ranges. These groups and ranges define priorities between the media elements of the presentation. They help in computing an optimal decision at a given time. In a similar way, the synchronized template for adaptable multimedia presentation (STAMP) model [22] defines *data*, *composition* and *decorative* dimensions in addition to traditional *spatial*, *temporal* and *interactive* properties in order to query and select related media elements based on a nucleus-satellite metaphor. A last example is the multimedia adaptation framework based on MPEG-21 standard [98] that can perform generic and dynamic media transformations based on perceived quality of service metrics. All these approaches do not directly tackle the adaptation of multimedia presentations but focus on the combined adaptation of several media elements that belong to the same presentation. These media-driven scene adaptation approaches mostly assume, as media-based approaches, that an underlying presentation model will smartly take into account these changes operated at the media level.

Such a management of media-level adaptation decisions can be enhanced by steering scene-level adaptation decisions. In particular, media-level adaptation decisions can be combined with a guided adaptation of the multimedia scene. For instance, the MPEG-21 Multimedia Scene Semantic Adaptation (MSSA) framework [75] provides semantic adaptation descriptors along with the content and helps a scene adapter in computing an optimized presentation. For instance, layout constraints such as a maximum resolution reduction factor of elementary media (`maxRRF`) can be defined. Such a hint from the editor can guide the computation of layout rules thus allowing the fragmentation of the presentation into several content pages. Additionally, dependencies between media components are key information that cannot be inferred easily. For instance, the MSSA descriptors introduce spatial relations between

media components that help in computing their acceptable distance in the adapted presentation (`SpatialSemanticDependencies`). A last approach defined by S. Laborie [78] consists in minimizing the modification of the presentation during the adaptation process by defining spatial, temporal and interactive distance between media components. In that case, the presentation can either be transformed through refinement (by maintaining the presentation constraints of the document) or through transgression [45].

Although media-driven approaches significantly extend adaptation capabilities compared to media-neutral approaches, they also require adaptation hints (e.g. additional scene descriptors or structure patterns) to meet author's expectations. The adaptation decision-taking algorithm remains fuzzy. As a consequence, the quality of the presentation resulting from such automated media-driven approaches can be questioned since it may not be easy, or even feasible, for the author to control the output of the scene adaptation process.

4.2.2 Custom scene publishing

The production phase of a scene is probably the best period to define adaptation procedures for a multimedia scene since it guarantees an excellent knowledge of the presentation semantics which is the most important ingredient for a successful adaptation recipe. Content production can be divided into two main steps as far as the scene is concerned: document generation and document formatting. During these two steps, scene transformations can leverage a deep knowledge of the presentation semantics in order to look for the best possible match between the user's environment and scene properties. Multimedia publication platforms based on custom scene publishing output scenes that are individually generated to address a specific adaptation scenario.

Custom publishing cannot always be considered as an adaptation process since it may not use any ADTE, which is the core of the adaptation process in our view (see Figure 20). Custom publishing sometimes consists in only doing scene format conversions, and even if they include an ADTE, they are not always designed with a clear separation between the ADTE and the publication platform. For instance, abstract graphical toolboxes, such as the GTK[39] or Java AWT/Swing or multi-environment toolboxes [32], can be used to guarantee the portability of a GUI on different operating systems. However, it does not mean that the interface will be adapted to the screen size as in multimodal approaches [30][93]. In the scope of this dissertation, scene format conversions are not covered since they constitute a complementary transformation to the adaptation process which is the main subject of our study.

In the following, two types of publication platforms generating adapted multimedia documents are described depending on the abstraction level of the document: model-based (Section 4.2.2.1) and format-based document publications (Section 4.2.2.2)

4.2.2.1 Meta-model scene adaptation

A popular state of the art approach [19] for scene adaptation consists in focusing on the content generation phase (as opposed to the formatting phase). Such an approach defines adaptation policies on a unique presentation model, which integrates all possible presentation models. Such an 'integrative multimedia document model', as S. Boll calls it [25], is often coupled with authoring systems such as ZYX [24], AHM [48], Madeus [71] or the Multimedia Presentation Generator System (MPGS) [20]. However, the complete modeling of an 'abstract multimedia content representation', as A. Scherp defines it, is suitable for editing a document but often fails to meet the requirements of a multimedia player [29]. An extended review of multimedia document models can be found in [24]. It describes a large panel of underlying presentation models of some major standardized formats or research projects by focusing on a list of basic and advanced requirements for multimedia documents as illustrated in Table 1.

[39] http://www.gtk.org

When considering multi-publication platforms, an optimal approach would consists in applying the adaptation process on a meta-model which 'gathers different models under one roof' and that can then be transcribed into any multimedia format by dedicated translators. Unfortunately, such a meta-model remains the Holy Grail of the multimedia community since the number of multimedia models constantly grows over time and because document requirements significantly evolved according to users expectations in few years. For instance, HTML+CSS, HTML+JS, HTML+AJAX, HTML 5.0, SVG, MPEG-4 BIFS, VRML, NCL, FLASH would significantly complicate the model classification proposed by S. Boll illustrated in Table 1. Moreover, scene properties described in Chapter 2 are richer than those proposed as a classification for the *spatial*, *temporal* and *interaction* entries in Table 1. As a consequence, a presentation meta-model cannot claim to be universal.

In the field of User Interface, which can be considered as a special type of multimedia document, the *concrete* user interface (such as events, callbacks and rendering) can be factored out in order to define an *abstract* user interface [103]. In that case, the adaptation facilities of the scene model will necessary lead to some limitations during the concrete user interface specification since all multimedia functionality will not be accessible to the author. However, such abstraction ensures that consistent interaction semantics are maintained independent of changes in the concrete user interface (e.g. adaptation to display resolution). Depending on targeted application scenarios, such high-level multimedia authoring might be acceptable since it offers a unified authoring (edit once and publish in all formats) and can also integrate scene adaptation functionalities.

Another example is the adaptation process proposed by S. Laborie in [78]. Indeed, this semantic approach cannot be directly applied to SMIL documents since SMIL is not based on connected graphs or qualitative relationships between the elementary media of the presentation. Instead, a SMIL scene is abstracted into a theoretical meta-model where a semantic adaptation process can be performed. Once adapted, the multimedia presentation is instantiated back in an adapted SMIL document. In that case, the functionalities of the adapted SMIL document are limited by the spatial, temporal and interactive scene properties taken into account by this meta-model. Besides, S. Boll underlines this loss saying that the ZyX model 'can [then] be seen as an internal format that can be converted to other model rather than a proprietary format […]. This allows us a (lossy) export or conversion of our multimedia document into standard data models […]. The model aims at supporting a rich multimedia functionality while still keeping a high semantic level' [24].

Table 1: Quote from [24] - Document models analysis

	HTML	HTML DHTML	HTML +TIME	SMIL 1.0	SMIL 2.0	…	Madeus
absolute positioning	+	+	+	+	+	…	+
spatial relationships	-	-	-	-	-	…	O
point-based	-	-	-	-	-	…	-
interval-based	-	-	+	+	+	…	+
event-based	-	+	-	-	+	…	-
script-based				-	-	…	-
general navigation support	+	+	+	+	+	…	+
linking from media subparts	spatial	spatial	spatial/ temporal	spatial/ temporal	spatial/ temporal	…	o
linking to media subparts	-	-	temporal	temporal	temporal	…	+
design	-	o	-	-	o	…	+
Multimedia functionality	very low	medium	low	medium	high	…	medium
Semantic level	medium	Very low	medium	medium	very low	…	very high

(+support, o partial support, - no support)

4.2.2.2 Meta-format scene adaptation

The production chain of a multimedia content can be divided into several steps that possibly involve various human actions such as conception, making and validation. Additionally, content production remains an iterative development where various competences are needed. For instance, a journalist and a designer might select elementary media according to editorial rules and graphical charter (conception) while the document created by a specialized technician (making) will be supervised by a chief editor before publication (validation). Moreover, several mock-ups might be created and further modified to refine the content concept or validate editorial choices to progressively reach the final version of the document. In addition to these multimedia scenes that are not generated only once, multimedia presentations are quite often built up with recycled content which is reused or updated. As a consequence, the adaptation of a multimedia scene in the production chain is not limited to conception phase as described in 4.2.2.1. It also has to take into account pre-authored inputs which have not necessarily been expressed using a meta-model.

The meta-format approach might look similar to the meta-model approach described in Section 4.2.2.1. Both rely on a document model that is transformed to meet environments constraints. However, the design of meta-model comes from the desire of having multimedia functionalities inspired from existing document formats, but which model does not necessarily cover all of their functionalities. This results into a new document model for each design. Instead, the meta-format approach is designed to cover a set of identified scene formats and must therefore be able to import any kind of multimedia scene as long as their format is supported. In this approach, the meta-format is not associated with any new document model. The meta-format defines a glue around the existing formats. In that case, the document generation and formatting phase rely on the underlying model of a single format (possibly harmonized) that can be converted to several export formats. For instance, the AHM approach was implemented in the CMIFed authoring system [49] based on the CMIF format and L. Hardman proposed to encode the CMIF format in HyTime since “a hypermedia document expressed in HyTime can be converted to other document types, or to particular output formats, using standard tools”.

Scene format conversion is a well-known subject of the state of the art. It raises the issue of the one-to-one map of such a scene transformation and the loss of data it may cause. S. Boll concludes that “if the target document model does not offer an equivalent multimedia functionality as offered by the source model, the conversion will be lossy”. S. Laborie also demonstrated with the SMIL format that the simple import/export in an abstract model without any adaptation might end up with a document different from the initial one [78]. The workaround proposed in that case consists in storing and transforming the hierarchical structure of the SMIL document along with the specification of the scene. However, there is no guarantee that such a structure-based approach will be compatible with other formats such as Flash for instance. The same issue is tackled by D. Thevenin in the Man-Machine Interface domain. D. Thevenin defines an ‘inverse zipping process’ which consists in minimizing the scope of conversions in the scene tree [109]. As S. Boll indicates, the question is about “how much power the integration layer should have? It could be the smallest common denominator of the primitives of the different integrated models. The integration model could also define more abstract primitives with specific semantics for integration”. The development of the second option rapidly becomes cumbersome because it implies a main model featuring exceptions for each targeted format. In practice, dedicated scene converters (SVG to SMIL for instance) are feasible but it is clear that the complexity of the operation that consists in inversing the transformation of an SVG polygon to a PNG image referenced in a SMIL scene shows the limits of a meta-format for multimedia scenes.

4.2.3 Scene alternatives selection

Although the content production phase benefits from a direct access to the author’s intention, it might not be the best moment to apply scene transformations because an appropriate decision also requires an accurate and up-to-date access to the user’s context. For this reason, a well-known adaptation approach of multimedia presentation consists in capturing author intentions by preparing a set of typical content configurations and then selecting, at the presentation time, the most appropriate version that addresses

the user's context. Two types of scene adaptation decision-taking approaches can be distinguished: explicit (4.2.3.1) or implicit (4.2.3.2).

4.2.3.1 Explicit alternative-based scene adaptation

The adaptation of a presentation basically consists in transforming the multimedia scene from its initial form to another version that fits the user's context. An adaptation policy consists in selecting the form of the multimedia scene from a set of pre-defined alternatives according to some context parameters [114]. For instance, the SMIL language defines a `<Switch>` element that allows the activation of the scene properties according to the context conditions. In that case, the mapping between the user's context and scene properties is explicitly defined by the author. In the same way, HTML and CSS format [123] allows authors to associate any number of mutually exclusive style sheets with a document (`alternate` CSS attribute) and to select the appropriate stylesheet depending on the context (e.g. using CSS Media Queries) [121]. For instance, an author may specify one style sheet designed for small displays, for printing or for partially sighted persons (e.g. with large fonts).

The selection of scene alternatives prepared during content authoring can be based on explicit environment constraints as introduced in Section 4.1. In case several presentation versions fulfil these requirements, quality metrics can be defined in order to optimize the decision-taking algorithm [77][108]. However, presentation alternatives usually remain limited to a small set since their edition can require significant authoring efforts.

4.2.3.2 Guided alternative-based scene adaptation

Several versions of the same presentation can be generated during the content production phase by using multimedia document publishing platforms described in Section 4.2.2. In that case, content creativity is not the focus of these presentation alternatives since they constitute pragmatic solutions automatically generated from an adaptation algorithm based on general heuristics. Therefore, an incremental transformation processor, such as incXSLT [112], can be used to control the scope of presentation changes. Based on a set of selected presentation versions, the author may modify and also annotate them with semantic descriptors possibly attached to their transformation rules (XSLT in that case). This semantics information constitutes implicit guidelines for an appropriate selection of presentation alternatives according to the user's context.

For instance, the role of the MMCM model [89] is mostly to provide knowledge on the presentation's semantics. Hence, semantics QoS parameters are used to determine the prioritization and categorization of the different presentation alternatives. In that case, parallel views of the presentation can be provided (or generated) to guide the adaptation of all physical layers (including the presentation layer) according to environment constraints thus defining an implicit selection algorithm.

4.2.4 Scene plasticity

Modern scene formats have overcome most content production limits by offering advanced multimedia functionalities. For instance, most of the creative inspirations of a web designer can be expressed by editing a Flash presentation. However, there is still an important need for highly-templated media components which only need to be filled in to become a multimedia document. This can be explained by two factors. First, whereas a significant effort can be spent on the custom design of the main presentations of a service, the large-scale production of daily content requires automated design. Second, since the content quality of multimedia production has become more democratic, there is a need for user-friendly configurable presentation toolkits[40]. Such a compromise between high-quality and authoring productivity is a trade-off that is also applicable to scene adaptation.

A popular adaptation approach, also called liquid or fluid design [90], consists in adapting a presentation to the available space, the same way water takes the shape of the glass it is in. In that sense, D. Thevenin uses the term 'plasticity' for the adaptation of the user interface [109]. We define scene plasticity as the

[40] http://www.dojotoolkit.org/

self-adaptation of the scene properties of a document to various environment constraints, such as the spatial layout according to the display resolution. However, scene plasticity does not have to be limited to visual properties and can also cover temporal and interactive properties. For instance, a menu bar can be created to support a variable number of menu items and configured with transition effects if an animation support is detected. In this content-oriented approach, the adaptation decision-taking algorithm can be defined as a function that directly maps user's context onto scene properties.

It should be noted here that scene plasticity can address very specific use cases, such as the layout of a justified text. Such automated layout approaches are not described in this dissertation. However, two generic principles related to the adaptation of visual properties through scene plasticity are described in the following: interpolation-based (Section 4.2.4.1) and constraint-based scene adaptation (Section 4.2.4.2).

4.2.4.1 Interpolation-based scene adaptation

The authoring of an interpolation-based adaptable presentation consists in creating several key scenes configured for typical usage environments. For adaptation purpose, key scenes can be interpolated to address usage environments that have not been initially covered during authoring. An application of such interpolation-based techniques is the adaptation of presentations to receivers screen size. The interpolation-based approach extends the scalability features of vector graphics languages such as SVG by providing intermediate versions that are specifically designed for targeted scene sizes. It also leverages authoring efforts of content creators who are used to provide their multimedia documents in several versions (large and small logo of a company, top or left banner for a Flash commercial on a web site…).

Artistic resizing is an interpolation technique introduced in [36] targeting the spatial adaptation of user interfaces. The artistic resizing approach offers non-linear resizing functions by transforming several copies of an object at key size using inference and interpolation algorithms in order to virtually support any resolution. In terms of processing, it requires mathematical computations for each media component to be adapted. The adapted scene properties of the artistic resizing approach are restricted to the spatial layout and orthogonal interpolations and still have some awkward limitations in terms of flexibility (e.g. rotations, cross-axis constraints, non-linear constraints). At the current state of the art, the main drawback of this approach is the flexibility of linear interpolations since subparts of the presentation cannot be suppressed or added.

4.2.4.2 Constrained-based scene adaptation

In this approach, the adaptation process takes into account usage environment variables by submitting them to a constraint-solving engine generated during the authoring that computes, at playback time, adapted scene properties [86]. The authoring of constraint-based adaptable multimedia documents consists in creating several versions of media components and scene configurations (style, spatial layout…) along with a set of one-way or multi-way constraints that must be maintained when transforming the document for adaptation purposes. The main difficulty raised by the authoring of constraint-based adaptable documents is that the specification of document constraints requires a high level of expertise. For instance, the explicit and global constraints that guide the adaptation process in [19] can be provided by an application developer but not a content creator. Based on this conclusion, a customized constraint-based authoring tool has been developed for diagrams [84] and allows the specification of default adaptation behaviors for multimedia presentations based on several diagram models. However, "a key question is [still] how the author can specify diagram specific adaptive layout behavior" without having "to write textual attribute expressions when, say, specifying one-way constraints in CSVG [17]".

4.3 Scene transformations

Within the adaptation domain, a scene transformation consists in modifying an original presentation into another one that is compatible with the playback environment of the user. However, scene

transformations can also be applied to a multimedia presentation in order to modify the perception of the user over time. For instance, scene description formats include scene transformations in order to cope with live scenarios (where transformations are needed after the scene has been processed and initialized) or to specify presentation behaviors upon user interactions. A scene adaptation engine can take advantage of these available functionalities in order to optimize its performance and to provide adaptation features at the lowest implementation cost.

In this section, state-of-the-art scene transformations currently implemented in multimedia players (i.e SVG [116], MPEG-4 BIFS [59], SMIL [119], HTML [120] or Flash) are described. These scene transformations can be used to adapt multimedia scenes according to context-based decision-taking algorithms. Our approach in this section consists in providing examples of standardized transformations such as format-specific facilities or external tools such as CSS [123], XSL [122] or JavaScript [64] descriptions. In the following, the identification of updatable scene properties and the conditions that trigger their transformation are discussed. This study follows an organization based on three types of transformations that can be applied to the XML-view of a multimedia scene: attribute replacements (Section 4.3.1), attribute spreading (Section 4.3.2) and element updates, including replacement (Section 4.3.3).

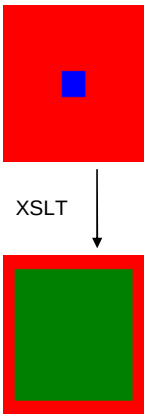
4.3.1 Scene attributes replacement

A multimedia scene is the concrete expression of a presentation according to models that have been introduced in Chapter 2. The transformation of a scene can be performed by addressing each property of its underlying presentation model (e.g. the vertical position of an image) and by updating its value (e.g. overwriting its actual value with a new vertical position). Several approaches can be used to achieve these attribute replacements depending on the conditions that trigger the scene transformation: the initialization of the process (Section 4.3.1.1), a timed event (Section 4.3.1.2) or external events (Section 4.3.1.3).

4.3.1.1 Init-based scene update

The update of scene attributes can be a one-shot process applied to the whole scene, typically upon presentation loading. In that case, all attribute updates are described as a list of replacement commands which associate identified scene properties and their replacement values. A typical example is the eXtensible Stylesheet Language [122] which defines transformation rules for XML documents that can be applied by an XSLT engine. Once started, the XSLT process performs scene transformations successively until the last command is executed. In Code 27, the `style` attribute of a `<DIV>` element (size, position and colour) of the HTML scene description is updated as soon as the web browser loads the content by applying the attached stylesheet. Similar scene transformation processes can be triggered during the loading phase of the presentation by catching events, such as the `load` event, which fires at the end of the content initialization.

Since a scene transformation targeting the whole multimedia document with a list of attributes to be updated does not depend on the presentation state, it does not necessarily have to be performed during the initialization phase: it can also be applied offline. However, most of the scene transformations meant for adaptation require a configuration which often depends on usage environments. A common trade-off consists in applying an XSLT transformation upon user's request in order to deliver a multimedia presentation which matches the initial rendering conditions [71][101]. Such an approach can also be used to convert a scene description into a new format or to publish a scene into a given format.




```
// extract from paramTransform.xsl
<xsl:template match="*[@id='inner']">
  <xsl:copy>
    <xsl:attribute name="style">
      position: absolute;top: 25px;left: 25px;
      width: 250px;height: 250px; background-color: Green;
    </xsl:attribute>
    <xsl:apply-templates/>
  </xsl:copy>
</xsl:template>

<?xml-stylesheet type="text/xsl" href="paramTransform.xsl"?>
<HTML xmlns="http://www.w3.org/1999/xhtml">
  <BODY>
    <DIV id="border" style="position: absolute;
      width: 300px; height: 300px;background-color: Red;"></DIV>
    <DIV id="inner" style="position: absolute; top: 125px;left: 125px
      width: 50px;height: 50px;;background-color: Blue;"></DIV>
  </BODY>
</HTML>
```

Code 27: Init-based attribute replacements in HTML using XSLT.

4.3.1.2 Time-based scene update

The update of scene attributes can be managed as a scheduled process where transformations are applied according to a timeline. In that case, attribute updates are replacement commands that are grouped according to the time the new values are supposed to replace current scene properties. For instance, a SMIL player can apply a list of `<set>` commands at a specified time according to an interval-based timing as illustrated in Code 28. Such a transformation could also be described by applying an external SMIL timesheet [127] attached to an HTML document.

In a different way, the MPEG-4 Synchronization Layer (SL) allows the association of an MPEG-4 BIFS update command, possibly referencing scene properties, with a timestamp. Once the timestamp deadline is reached according to the player's internal clock, the BIFS decoder performs buffered scene transformations. Such an approach can be used to transform multimedia services where the active scene varies over time. Additionally, scene transformation processes can be run without any clock reference by applying replacement commands upon reception (such as DOM mutation events). In that case, the scheduling is performed by transmitting groups of replacement commands progressively using streaming or broadcasting transport mechanisms. This method is applicable with replacement commands, expressed for instance in the MPEG-4 BIFS format, or through Remote Event for XML [124].

In the adaptation domain, time-based scene updates can be used to progressively transmit a document over time. In narrow-bandwidth scenarios, a multimedia player can progressively apply scene attribute replacements as if they were timed properties of the multimedia service, without knowing they are adaptation transformations.

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
 <head><layout>
   <root-layout id="border" width="300" height="300" background-color="red"/>
   <region id="in" top="125" left="125" width="50" height="50" background-color="blue"/>
  </layout></head>
 <body>

 <par dur="10s">
 <set targetElement="in" attributeName="width" begin="5s" dur="10s" fill="freeze" to="250"/>
 <set targetElement="in" attributeName="height" begin="5s" dur="10s" fill="freeze" to="250"/>
   <set targetElement="in" attributeName="top" begin="5s" dur="10s" fill="freeze" to="25"/>
   <set targetElement="in" attributeName="left" begin="5s" dur="10s" fill="freeze" to="25"/>
   <set targetElement="in" attributeName="background-color" begin="5s" dur="10s"
        fill="freeze" to="green" />
 </par>
 </body>
</smil>
```

Code 28: Time-based attribute replacements in SMIL using timed properties.

4.3.1.3 Event-based scene update

The update of scene attributes can be handled by an event manager that triggers transformation sub-processes according to fired events. In that case, attribute updates are replacement commands that are grouped to be applied together by successively updating the values of targeted scene properties. For instance, JavaScript code can describe transformation rules for an XML document using the DOM API and can define activation triggers for each group of scene transformations as illustrated in Code 29. Once started, a JavaScript engine will listen to event notifications and will perform scene transformations whenever a targeted event is caught.

In terms of scene adaptation, event-based scene updates can be used to transform multimedia documents requiring scene transformations that need to be repeated. In particular, the event-based approach can be suitable for dynamic context where the adaptation decision can quickly evolve and may imply to reprocess some parts of the scene, without relying on new data coming from a server.

```
<svg  xmlns="http://www.w3.org/2000/svg" width="300" height="300" viewBox="0 0 300 300">
  <script id="script-root" type="text/ecmascript">
    function mouseDown() {
      innerRectangle = document.getElementById("inner");
      innerRectangle.setAttribute("width", 250);
      innerRectangle.setAttribute("height", 250);
      innerRectangle.setAttribute("x", 25);
      innerRectangle.setAttribute("y", 25);
      innerRectangle.setAttribute("fill", "green"); }
  </script>
  <g onmousedown="mouseDown()">
    <rect id="border" width="300" height="300" fill="red"/>
    <rect id="inner" width="50" height="50" x="125" y="125" fill="blue"/>
  </g>
</svg>
```

Code 29: Event-based attibute replacements in SVG using JavaScript.

4.3.2 Scene attributes spreading

A scene aims at organizing several media components into a coherent presentation. Hence, there may be a strong correlation between some scene properties. For this reason, the replacement of some scene attributes may lead to the replacement of other attributes either with the same value or with a new value that can be determined from it. Instead of duplicating scene replacement commands, as it could be done by only relying on the attribute replacement approach described in Section 4.3.1, another technique consists in applying a single attribute replacement and then spreading this value to other related attributes. In the following, four different scene attribute spreading approaches are presented: replication (4.3.2.1), inheritance (4.3.2.2), bubbling (4.3.2.3) and routing (4.3.2.4).

4.3.2.1 Replication

The simpler spreading approach for scene attributes is probably the description of scene properties that reference values of other scene properties for their initialization as illustrated in Figure 22. The presentation of a drawing composed of several graphics elements might take advantage of a colour palette. In that case, the update of one colour attribute will automatically update the colour of all rectangles that refer to it. Such a replication process is compatible with dictionary-based authoring paradigms where the actual multimedia scene mainly consists in organizing the different words of a dictionary composed of media components.

The reuse of existing content is one of the essential requirements underlined by S. Boll [24] because it significantly speeds up content production and takes into account the common practice of content recycling. Additionally, a proper authoring would explicitly reuse scene properties in order to avoid attribute duplication by rather relying on attribute replication. Such automated replication process can successfully be exploited to spread the adapted properties of a multimedia scene.

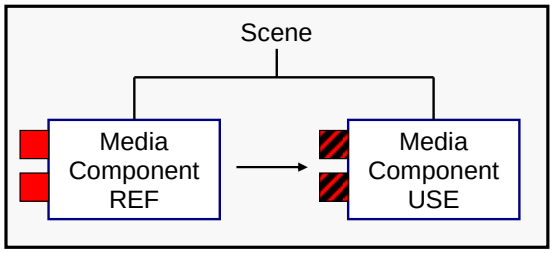


Figure 22: Replication of scene attributes.

4.3.2.2 Inheritance

A multimedia scene might be structured as a hierarchical tree where branches are all related to a trunk (or root node). The connection between two branches is called a node (or container element) and a leaf ends each branch mostly with elementary media such as video, audio, image, text or graphics elements. The hierarchical structure of a multimedia document favours the efficient management of scene properties. For instance, cumulative coordinates can be described to define the position of a media component or relative start time can be used in the case of interval-based timing models. Additionally, a hierarchical structure allows grouping some shared scene properties such as media components visibility or their sensibility to user actions. The propagation of attributes in the scene tree can follow a rising of sap approach, also called inheritance approach, when considering the multimedia scene as an upside down tree as illustrated in Figure 23.

The inheritance process targets scene attributes of leaf elements that are not entirely described at the leaf level. These attributes can be set to their default value (e.g. default black colour) or specified as modifiable (e.g. coordinates when relative positioning is used). In that case, all the ancestors of a leaf can specify a value that overrides its default value. As a consequence, the update of a scene attribute might automatically imply the update of the scene attributes of all its children. In practice, inheritance is largely used during authoring since it avoids duplicating related values and enables a compact description of the scene which can be efficiently transmitted to the user. In that case, the playback process is responsible for applying scene transformations during the rendering phase.

In practice, many existing standards provide such inheritance feature. Hence, the style properties of Cascading Style Sheets [123] can be inherited from the document structure when attributes are set as `inherit`. As a consequence, the modification of the scene structure during adaptation may impact some inherited scene attributes such as style properties.

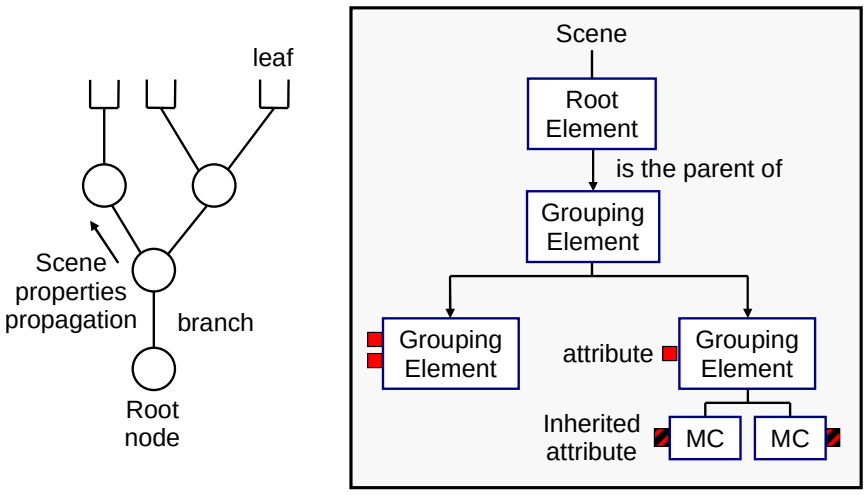


Figure 23: Inheritance of scene attributes.

4.3.2.3 Bubbling

Scene transformations based on inheritance spread scene properties from the root of a scene tree down to its leafs elements. It naturally follows authoring approaches where a master scene manages media components in order to create a presentation. However, the elementary media of a presentation might also trigger scene transformations. For instance, the user action on an image, a video sequence that ends or any event fired by a media component might trigger attribute updates that do not only target children nodes or leafs but also parents or sibling elements. As a consequence, the propagation of an attribute value in the scene tree can follow the parent chain upward and scatter in their children looking for targeted scene attributes to be updated as illustrated in Figure 24. Upon modification, these scene attributes behave as bubbles that rise along the branches of the scene tree (bubbling process). For instance, the properties of an SVG document can be modified through some attribute changes using a `DOMAttrModified` event listener.

Among all scene attributes, the output of event generators, such as interactive sensors, typically rely on bubbling processes to be dispatched to all the event listeners of the document. The DOM 2.0 recommendation [130] specifies a bubbling procedure and allows event handlers to stop the propagation of the bubbling event flow (`stopPropagation`).

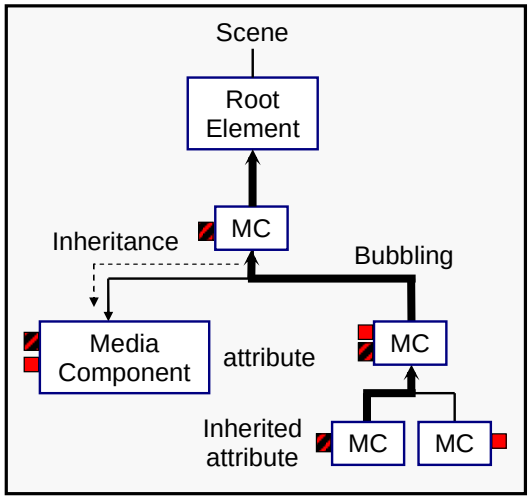


Figure 24: Bubbling of scene attributes.

4.3.2.4 Routing

Inheritance and bubbling processes heavily rely on the hierarchical structure of the multimedia document to automatically spread transformed scene properties. However, this document structure might be useful for multiple scenarios: spatial organization, temporal scheduling, interactive sensitivity, prioritized visibility… Hence, the document structure has to be designed taking into account all of these needs and may suffer limitations due to incompatible usages. A possible workaround consists in defining an explicit path between two scene attributes to propagate a value from one attribute to another. This routing description creates new connections between scene attributes (routes) and this additional structure can be independent from the hierarchical structure of the multimedia scene as illustrated in Figure 25. Such routing description can be found in MPEG-4 BIFS (`<Route>` element) [59] or in NCL (`<link>` element). [105]

In the scope of content adaptation, routes can be seen as one-way constraints which replace the value of a scene attribute according to a computed value from another scene attribute. Constraint-based decision-taking consists in adjusting free variables of the system while the scene transformation process updates attributes following validated routes that fulfil usage constraints.

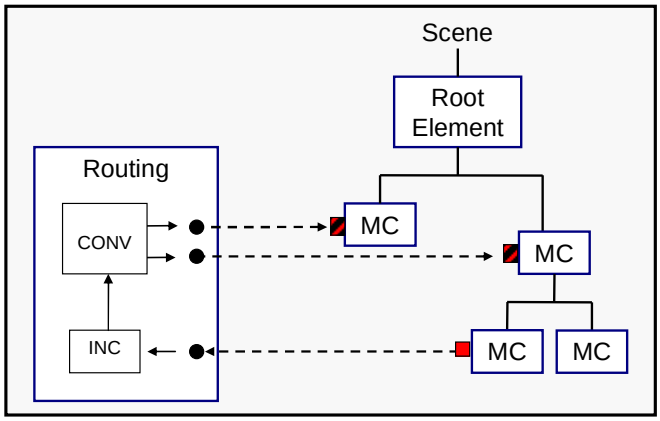


Figure 25: Routing of scene parameters.

4.3.3 Scene elements update

The elements of a scene can be updated through scene transformations that can deeply modify the document or its format-dependent description. The update of elements relies on the same identification and triggering mechanisms as the replacement of scene parameters (Section 4.3.1). However, scene element updates differ from attribute updates for two main reasons:

- identifiers do not only point to a list of scene attributes but may point to a group of elements. It includes the identified scene element to be updated but also all its children elements. The grouping strategy of scene elements in a document might be driven by authoring guidelines in the case of semantically-related elements. Scene elements might also be grouped in order to facilitate the progressive rendering of the content by gathering them according to their significance. A last example is the grouping of elements to minimize playback requirements when animating the position of several media components. As a consequence, transformations targeting scene elements might disturb the whole document and therefore require a more advanced understanding of the content than scene attribute replacements.
- the nature of the transformation can be the replacement of an existing scene element by a new one as it is the case for scene parameters replacement. However, scene element updates can also manipulate the multimedia scene and modify its structure.

A multimedia service requires the modification of the media components referenced by the scene over time. Such modifications imply significant scene transformations including attribute updates but also element updates that possibly modify the architecture of the scene. Additionally, the transformation of scene elements is essential when designing large interactive presentations since the footprint of a scene has to be controlled on handheld devices due to memory requirements. Scene element updates can be described as a set of four scene transformations: insertion (Section 4.3.3.1), deletion (Section 4.3.3.2), replacement (Section 4.3.3.3) and move (Section 4.3.3.4) as considered in the XML Delta model [100].

4.3.3.1 Insertions

The insertion of a media component requires a scene structure that is dynamic since the document possibly evolves during a scene transformation. However, such dynamicity of the multimedia scene requires specific authoring efforts (see scene plasticy approach in Section 4.2.4, for instance). Additionally, the insertion of a new media component in the scene may trigger a brand new computation of the presentation which requires rendering processing capabilities on player side that might be too important in some scenarios. For these reasons, scene elements might be introduced in the presentation along with all the visual, temporal and interactive transformations it implies. As illustrated in Figure 26, a new media component might be added as a new slide of a slideshow along with all the JavaScript description required to update the interactive selection mechanisms.

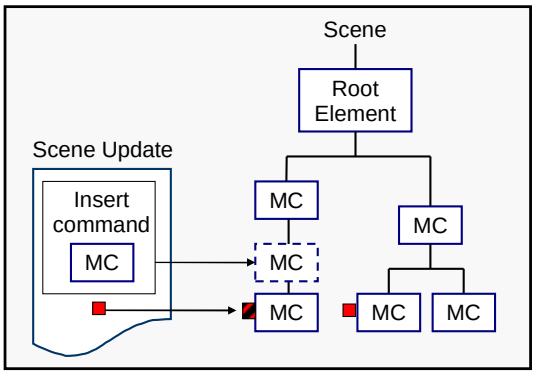


Figure 26: Insertion scene elements.

4.3.3.2 Deletions

The deletion of media components consists in removing them from the scene either because their activation is permanently disabled or because the scene needs to be cleaned from these unused parts. The scene manipulation process involved when deleting media components are the same as for element insertion: related scene elements and attributes need to be cleaned off either by relying on flexible structure or by explicitly targeting them. As illustrated in Figure 27, an item from a slideshow can be removed by setting back selection mechanisms.

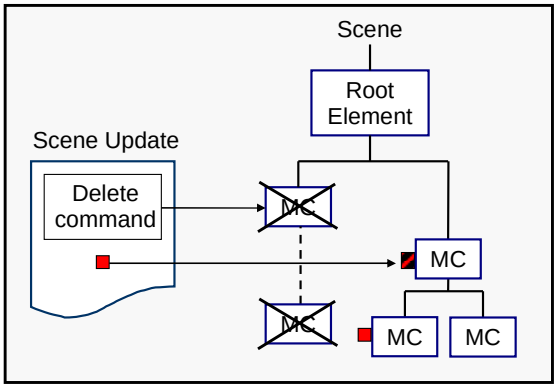
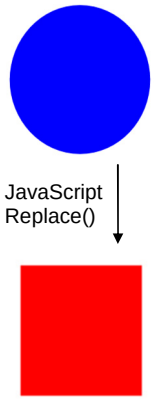


Figure 27: Deleting scene elements.

4.3.3.3 Replacements

The replacement of scene elements might be performed by a combined process that delete the identified elements (Section 4.3.3.2) and insert brand new ones (Section 4.3.3.1). For instance, the MPEG-4 BIFS `<Replace>` command can be executed by an XML parser that is not aware of the details of the XMT-A syntax except this replacement command as illustrated in Code 32. In the same way, a JavaScript function can perform the replacement of XML elements (Code 30). As opposed to these memoryless element replacements, the replacement process might analyze the elements to be replaced so as to configure the new elements to be inserted to the current scene. For instance, a XSLT process can parse the XML syntax of the initial SVG document description to extract some scene attributes, such as a rectangle size, that are then integrated in a new HTML document as illustrated in Code 31 and Code 33.

Although replacement commands primarily target specific scene elements, it can also be used to replace the root node of a scene. As a consequence, the replacement of scene elements can be considered as the minimal requirement for adapting a multimedia presentation since it can reach all scene properties.




```
// extract from paramTransform.xsl
<svg  xmlns="http://www.w3.org/2000/svg" width="100%" height="100%"
      viewBox="0 0 100 100" onload="replace(evt)">
  <script id="script-root" type="text/ecmascript">
    function replace(evt){
      var svgNS = "http://www.w3.org/2000/svg";
      var circle = document.getElementById("circle");
      var rect = document.createElementNS(svgNS, "rect");
                  circle.setAttributeNS(null,"width", 50);
                  circle.setAttributeNS(null,"height", 50);
                  circle.setAttributeNS(null,"fill", "red");
                  rect.parentNode.replaceChild(rect, circle); }
  </script>
  <g>
    <circle id="circle" cx="25" cy="25" r="50" fill="blue"/>
  </g>
</svg>
```

Code 30: Replacement of scene elements in SVG using JavaScript.

```
// extract from paramTransform.xsl
<xsl:template match="@*|node()">
  <xsl:copy>
    <xsl:apply-templates select="@*|node()"/>
  </xsl:copy>
</xsl:template>
<xsl:template match="*[@id='circle']">
  <svg:circle id="rect" width="50" height="50" fill="red"/>
  <xsl:apply-templates/>
</xsl:template>

<?xml-stylesheet type="text/xsl" href="elemTransform.xsl"?>
<svg  xmlns="http://www.w3.org/2000/svg" width="100%" height="100%" viewBox="0 0 100 100">
  <g">
    <circle id="circle" cx="25" cy="25" r="50" fill="blue"/>
  </g>
</svg>
```

Code 31: Replacement of scene elements in SVG using XSLT.

```
<Replace><Scene>
  <OrderedGroup><children>
    <Transform2D DEF="T2D"><children>
      <Shape><appearance><Appearance USE="APP_BLUE"/></appearance>
        <geometry>
          <Circle radius="35"/>
        </geometry>
      </Shape>
    </children></Transform2D>
  </children></OrderedGroup>
</Scene></Replace>

<par begin="0">
  <Replace atNode="T2D">
    <Transform2D><children>
      <Shape><appearance><Appearance USE="APP_RED"/></appearance>
        <geometry>
          <Rectangle size="50 50"/>
        </geometry>
      </Shape>
    </children></Transform2D>
  </Replace>
</par>
```

Code 32: Replacement of scene elements in BIFS.

```
// extract from svgToHTML.xsl
<xsl:template match="/">
  <html:HTML xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <html:BODY>
   <xsl:for-each select="svg:svg/svg:circle">
    <html:DIV style="position: absolute;
       top: {@cy}px; left: {@cx}px; width: {@r}px;height: {@r}px;background-color: {@fill};"/>
   </xsl:for-each>
  </html:BODY>
 </html:HTML>
</xsl:template>

<?xml-stylesheet type="text/xsl" href="svgToHTML.xsl"?>
<svg  xmlns="http://www.w3.org/2000/svg" width="100" height="100" viewBox="0 0 100 100">
  <circle id="circle" cx="25" cy="25" r="50" fill="blue"/>
</svg>
```

Code 33: Replacement of scene elements in SVG into HTML using XSLT.

4.3.3.4 Moves

The scope of the manipulation of a scene does not have to be narrowed to the transformation of a single scene element. In particular, a piece of the scene can be cut and paste into another part of the scene. The move manipulation of the scene can be performed combining insertion and deletion commands. However, move commands constitute a significant optimization since scene elements do not have to be repeated in that case. Moreover, moving scene elements allows to dynamically modifying the scene structure while keeping the current presentation state as illustrated in Figure 28. For instance, deletion and insertion transformations would imply the loss of scene properties modified by the user through interactivity during document playback. Only siblings, cousins or distant cousins elements can be moved (no child relation is possible in that case)

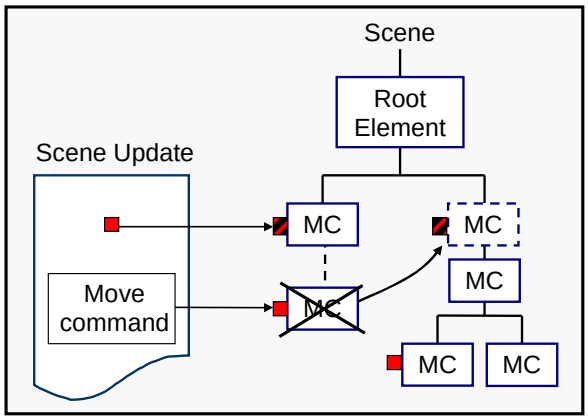


Figure 28: Moving scene elements.

4.4 Conclusion

The adaptation of a multimedia scene requires an advanced control over the scene description models which has been previously described in Chapter 2. Hence, scene properties can be complemented with adaptation semantics (Section 4.2.1), abstracted into high-level models (Sections 4.2.2), prepared for a set of typical environments (Section 4.2.3) or designed to cope with uncertain contexts (Section 4.2.4). All of these approaches endeavor to deal with a large panel of content adaptation scenarios while maintaining a good content quality. They also have inherent limitations that define the scope of their application scenarios that are detailed below:

- Media-based scene adaptation efficiently tackles the adaptation of elementary media but also restricts the adaptation of scene properties to simple scenarios (possibly guided by author preferences).
- Custom scene publishing lightens the burden of editors by providing authoring facilities through the abstraction of scene properties that can be automatically adapted. However, such an abstraction reduces the capabilities of multimedia presentations since only a part of the scene

properties can be transformed through adaptation algorithms. Furthermore, these transformations follow generic aesthetic rules that do not necessary fit the author's needs.

- Scene alternatives selection enables the full control over adapted scene properties but also requires significant authoring efforts that limit the targeted adaptation scenarios to a set of typical use cases.
- Scene plasticity allows the definition of adaptation behaviors as a part of the document for some scene properties. These transformation functions can be configured according to the user's environment. However, these specific functions, closely related the content, cannot be applied to all properties and may require complex authoring tools.

Naturally, it would be idealistic to look for an optimal solution targeting all adaptation scenarios based on a unified proposal; this is because there is always a trade-off between the need for low-level adaptation control and simple authoring paradigms depending on application scenarios. However, all of these approaches could be combined into a single production chain. Media-level adaptation decisions taking into account presentation considerations could be combined with custom scene generation tools to propose presentation alternatives to the author for editorial modifications and validation. Based on adaptation scenarios, scene alternatives and flexible media components (plastic media) would be combined in order to efficiently modify the properties of created scene descriptions according to the actual user's context.

The scene transformation techniques described in Section 4.3 are currently available in multimedia format (MPEG-4 BIFS update commands) or applicable using external tools such as JavaScript [64] or XSL [122]. These scene transformations allow an advanced and optimized control over scene transformations by replacing scene attributes (Section 4.3.1), controlling their spreading in the scene tree (Section 4.3.2) and updating scene elements (Section 4.3.3). Additionally, numerous communication protocols suitable to convey multimedia scene offer placeholders for descriptors that could ease the deployment of adaptive solutions. As a consequence, a scene adaptation decision-taking engine should generate adaptation parameters that can be easily formatted into such descriptors. Thus, adaptation scenarios naturally cover the dynamic transformation of multimedia presentations during the authoring phase but also while being transmitted or even during the course of the playback on the end-user terminal.

Finally, the scene decision-taking process and transformation process described in this chapter incur overheads (e.g. implementation costs, processing power, memory requirements, battery consumption, bandwidth…). These overheads have to be carefully weighed depending on application scenarios. For instance, some approaches might be oversized compared to expected adaptation features. In some cases, environment constraints might prevent the implementation of some approaches because they would worsen these constraints and directly impact content quality. Thus, our study on multimedia digital radio broadcast has led to technical choices and enhancements that will be described in Chapter 5. This study paves the last miles toward multimedia scene scalability.

Chapter 5 Towards multimedia scene scalability

The background of our scalable scene model that will be described in Chapter 6 (*Scalable MSTI*) can be easily identified from the classification of scene properties as described in Chapter 2 (viz. spatial, temporal and interactive properties). However, in terms of scene adaptation, the underlying principles of our scalable model significantly differ from the existing approaches described in Chapter 4. The main factors that led us to this new proposal are detailed in this chapter. In particular, we show in Section 5.1 how a large number of existing solutions fail to address our adaptation requirements in broadcast environments. Additionally, we describe in Section 5.2 and Section 5.3 two contributions that we developed starting from the existing solutions and that partially succeeded in addressing our needs. The results of these two preliminary experiments suggest beforehand the essence of our proposal: multimedia scene scalability.

5.1 Adaptation requirements

All the adaptation requirements that guided technical choices during our research activities come from a single use case which is the delivery of adaptable multimedia services over a broadcast channel. In practice, the adaptation of the presentation of a broadcast service to terminal capabilities and to network capacities is a challenging task. It can be derived into multiple scenarios such as file casting, progressive content downloading or real-time streaming. Indeed, although document sharing or on-demand client-server streaming did not directly drive our choices, it appeared that our 'broadcast' requirements also fulfil these 'unicast' or 'local' use cases thus enabling a hybrid broadcast-broadband adaptive solution [88].

In the following, the main adaptation requirements that define the sub-header of this dissertation are detailed: *"A generic, autonomous and low-overhead transformation process for the controlled adaptation of advanced multimedia services"*

5.1.1 Generic adaptation process

Content adaptation needs are always specific since they are determined by the particular wish of an editor and the unique context of the user. However, the adaptation process applied to a presentation must remain generic so as to be implemented on multimedia devices that were not configured for these dedicated application scenarios. In particular, broadcast scenarios cannot assume any dedicated pre-installed software on playback platforms for each service. For instance, an iPhone with a 3G or Wi-Fi connection may download a custom application for each service. However, many other handheld terminals used to watch television or listen to the radio only rely on broadcast signals to retrieve data. As a consequence, all adaptation processes have to be implemented in a framework, possibly standardized, that is generic enough to be industrialized. Hence, media-driven (Section 4.2.1.2), meta-model (Section 4.2.2.1), meta-format (Section 4.2.2.2) and guided alternative selection (Section 4.2.3.2) decision-taking algorithms sometimes rely on sophisticated implementations dealing with content semantics which can be developed on a publication platform but are not generic enough to be largely deployed on receivers.

5.1.2 Autonomous adaptation process

Broadcast scenarios do not allow any external support for the adaptation of a multimedia presentation other than broadcast channel information. Although connected TV could provide an adaptive feedback channel for broadcasted services, the current developments focus on unicast adaptive applications such as the Samsung's TV Apps store[41]. Hence, adaptation decisions must be exclusively taken from the input service's description provided to the receiver. As a consequence, all adaptation proposals based on scene alternatives (Section 4.2.3) selected through the analysis of users requests are not suitable for broadcast scenarios. Instead, all possible alternatives need to be transmitted to receivers so that they can perform the presentation adaptation themselves. Such scene alternatives or additional transformation instructions

[41] http://tv.samsungapps.com/

are data that must remain very limited compared to the main media transmitted through the broadcast channel (Section 4.2.1.2). It can also be noticed here that meta-model (Section 4.2.2.1) and meta-format approaches (Section 4.2.2.2) are not applicable in broadcast environments since only one standardized format should be used for document description in a mass market such as free-to-air TV and Radio services.

5.1.3 Low-overhead adaptation process

An essential requirement for the adaptation of multimedia services is that the actual transformation process should not impact the capabilities of the end user terminal. Indeed, an adaptation process that would significantly modify user's context adds uncertainty on its own input parameters. For instance, a scene transformation that aims at reducing content complexity to match the limited processing power capabilities of terminals must not require a complex processing. In practice, the well-known option which consists in operating the adaptation process on a distant server in order to lighten the burden of receivers cannot be applied in broadcast environments. Meta-model (Section 4.2.2.1) and meta-format approaches (Section 4.2.2.2) are not applicable in that case. Although a transparent adaptation process would be difficult to achieve, all additional costs on bandwidth, memory consumption and processing power have to be minimized. In fact, a single rule summarizes the acceptable costs of adaptation functionalities: the bigger the overheads are, the higher the benefits of adaptation must be.

Among our classification of adaptation approaches, media-driven scene adaptation (Section 4.2.1.2) is a possible option that requires efficient scene transformation techniques. Whereas implicit adaptation transformations (Section 4.2.3.2) could be directly performed on receivers, they do not fit well with limited platforms where the intensive processing of a semantic adaptation decision-taking engine cannot be easily implemented. Finally, constrained-based scene adaptation approach (Section 4.2.4.2) requires noteworthy processing capabilities for limited devices and do not scale well with multimedia content. Indeed, even if the complexity of one-way constraint-solving techniques [110] is not high, it might not be acceptable for very limited terminals especially when the number of media objects remains reasonable [86]. Furthermore, a description of adaptation constraints, provided as JavaScript code in [84], requires a significant bandwidth when frequently repeated in a broadcast carousel. Indeed, such adaptation descriptors cannot be resident in the terminal since they differ for each content.

5.1.4 Dynamic adaptation process

In broadcast environments, multimedia radio services are either transmitted as continuous streams (e.g. MPEG-4 Elementary Streams in T-DMB [38]) or provided as individual chunks (e.g. using Multimedia Object Transfer of T-DAB [43]). In both transmission scenarios, a multimedia service can be considered as a live document. However, services differ from the simple succession of independent documents because a significant value of the content production comes from the editorial choices made to dynamically compose these documents over time by introducing seamless transitions. An adaptation process must be able to follow such dynamicity. In particular, all adaptation proposals which define the adaptation decision-taking as an initial step prior to the complete presentation delivery cannot be applied. Only a persistent decision-taking engine can handle a never-ending document and cope with the variability of user's context over such a long period of time.

Even though an optimal presentation can be generated for any new context, the coherence of the user's experience might require some hysteresis effects. For instance, it might be more beneficial for the user to keep watching an image slideshow than switching to a video sequence for a short period of time during occasional bandwidth increases. Additionally, the seamless transition between two adapted multimedia documents requires an advanced management of scene properties. In that case, the abstraction of a meta-model (Section 4.2.2.1) or a meta-format (Section 4.2.2.2) approach raises severe issues since such a service continuity often relies on low-level scene properties provided to the user over time.

Finally, constraint-based adaptive documents (Section 4.2.4.2) are not designed to be updated over time. Indeed, whenever a new object, which was not considered during authoring, is to be dynamically inserted to the multimedia document, the whole constraint-based algorithm description has to be updated and the

complete adaptation has to be reprocessed. However, incremental constraint solvers, such as DeltaBlue [46], can be considered for that purpose.

5.1.5 Enhanced adaptation process

Content quality is an essential aspect for service providers who propose a multimedia offer through a dedicated channel (e.g. a DMB sub-channel) or a web portal (e.g. a website) because the loyalty of users is a key success factor. In fact, a significant part of authoring effort is spent on selecting the set of media that will catch user attention and in designing the presentation in such a way that the content looks attractive. The adaptation of multimedia presentations is another way to make content more comfortable for users by avoiding never-ending loading time, unpredictable degraded services or even playback failures that would ruin the service reputation. However, adapting a presentation should not interfere with the editorial choices of the author or dictate authoring constraints that would ease the adaptation decision-taking process. Instead, the content adaptation process must be flexible enough to handle advanced multimedia scenes (such as SVG [118], MPEG-4 BIFS [59], SMIL [119], HTML [120], NCL [105] or Flash) where the spatial, temporal and interactive properties of a presentation are essential and allow content creators to carefully design their presentations. For this reason, all media-neutral scene adaptation approaches (Section 4.2.1.1) that assume simplistic (e.g. audio/video only) or structure-driven presentations (e.g. SMIL-based layout) where the analysis of the multimedia content is straightforward only cover a limited range of content possibilities. In the same way, meta-model (Section 4.2.2.1) and meta-format approaches (Section 4.2.2.2) only offer a limited set of functionalities compared to the multimedia formats they target. Finally, interpolation-based adaptation (Section 4.2.4.1) is not sufficient to deal with the adaptation of all scene properties.

5.1.6 Controlled adaptation process

An editor is responsible for the multimedia service it produces. For this reason, the content production chain is usually composed of several steps where the content is controlled by humans in order to check its quality but also that editorial rules have been respected. For instance, it is not acceptable to broadcast a tour operator advertisement next to the headline about a plane crash. The scene transformations applied for the adaptation of a presentation must follow the same validation process to always remain under control. For this reason, all implicit scene transformations (Section 4.2.3.2) that let software systems automatically operate the adaptation of the presentation are not acceptable. Of course, these approaches lighten the burden of content creators by taking care of usage environment constraints but output documents that are completely out of the control of the editor. However, all of these automated adaptation decision-taking algorithms remain excellent tools for assisting authors during the content generation phase.

The complete control of an adaptable presentation quickly requires increasing efforts as the number of adaptation parameters grows. Two options can be envisaged: lessening editorial checkpoints and/or limiting adaptation scenarios. For instance, media-driven scene adaptation techniques (Section 4.2.1.2) can offer some control to the author over adaptation decisions by providing optional semantic information. However, these approaches [75][78] are usually not satisfactory because they still require quite a lot of authoring efforts without providing a clear control on their impact in terms of content usability. In the same way, the specification of presentations resulting from the resolution of a constraint-based algorithm (Section 4.2.4.2) requires advanced validation tools to cope with all possible configurations. As a consequence, a trade-off between content control and adaptation flexibility needs to be found.

5.1.7 State of the art analysis

The result of the combined analysis of our adaptation requirements and state-of-the-art scene adaptation approaches is summarized in Table 2. It can be seen that four main approaches can be envisaged for the adaptation of multimedia scenes in broadcast environments:

- The media-driven approach can be a good candidate if it can offer a clear control over scene modifications to the author, make use of efficient scene transformation techniques and optimize the amount of data required for scene adaptation.

- The alternative-based approach is an excellent candidate as long as the weight of scene alternatives can be reduced to acceptable values (especially for large presentations and memory-limited devices). Additionally, the generation of presentation alternatives is a tedious task that should be supported by automated content generation [27][70].
- The constrained-based approach is an attractive option but raises important issues when the adaptation of dynamic services in constrained environments is considered. Additionally, it remains a complex task for authors to define content constraints (developer-oriented approach) and for receivers to resolve them with low-processing requirements.
- The interpolation-based approach can be seen as a light constrained-based approach focused on some specific scene properties (visual properties mainly). This approach combines the advantages of a compact representation and rendering optimizations for scene transformations when used for dedicated purpose (e.g. text layout for instance). It should be used for adaptation purpose whenever the content makes it possible.

Table 2: Matching GALDEC scene adapation requirements with state-of-the-art approaches.

	Generic	**A**utonomous	**L**ow-Overhead	**D**ynamic	**E**nhanced	**C**ontrol
Media-neutral	+	+	+	+	-	+
Media-driven	o	o	o	+	+	o
Meta-model	o	-	-	o	-	+
Meta-format (publishing)	o	-	-	o	-	+
Alternative-based (Explicit)	+	o	+	+	+	+
Alternative-based (Guided)	o	o	-	+	+	-
Constrained-based	+	+	o	o	+	o
Interpolation-based	+	+	+	+	o	+

+ supported, o might not be supported or require specific optimization, - unsupported

In following, we describe two contributions we developed to fulfil our adaptation requirements starting from state-of-the-art approaches. First, we have combined media-neutral and explicit alternative-based scene adaptation to make a trade-off between the need for media adaptation and scene adaptation control with a coarse granularity (Section 5.2). This contribution defines a new media-driven approach for scene adaptation. Second, we have defined a new progressive scene transformation technique for the alternative-based approach in order to improve its autonomy (Section 5.3). Our experiments compared our approach with classical alternative-based approaches and an interpolation-based approach based on objective measures.

5.2 Media-driven presentation adaptation

The media-driven scene adaptation presented in this section is a method to complement existing audio, video, and image adaptation techniques in order to improve the perceived quality of interactive rich-media applications. Some scalable video codecs [104] and image codecs [85] organize the media bit-stream in abstraction layers, thus simplifying the adaptation process to cut and paste operations [66]. Additionally, transcoding and transmoding offer a wide range of adaptation scenarios such as: coding format transformation, video key-frames extraction, text-to-speech synthesis, etc. Both approaches, scalable and non-scalable, can also be coupled to guarantee the highest media quality in a large variety of usage environments. However, media adaptation, in its own, is not sufficient to guarantee a satisfying quality of the end user's experience in interactive rich-media applications. In fact, the adaptation of a media may significantly impact the user's display and may require the accommodation of the whole presentation, as illustrated in Figure 29, where a large portion of the display remains empty.

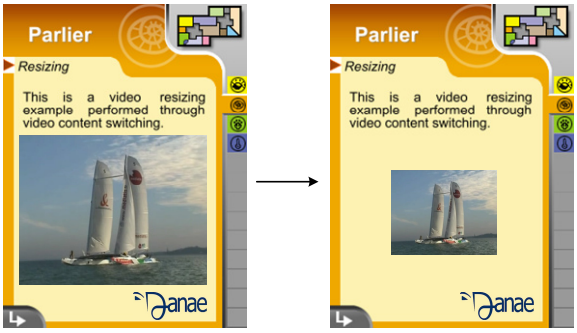


Figure 29: Example of a non-optimal media-driven scene adaptation.

Our approach published in the proceedings of the International Workshop on Image Analysis for Multimedia Interactive Services [6] consists in dynamically adapting multimedia services at the presentation level according to media-level adaptation decisions. The principle of this approach is further explained in Section 5.2.1. Section 5.2.2 depicts the application test-bed using MPEG-4 and MPEG-21 that we implemented with the proposed adaptation approach and concludes our study with an evaluation against our adaptation requirements listed in Section 5.1.

5.2.1 Principles

The adaptation of elementary media (audio, video, images, text paragraph or graphic elements) is a prerequisite for scene adaptation. Indeed, it would not make sense to adapt the presentation of an elementary media that can not be displayed to the user. Instead, a multimedia presentation should adapt to its transformed media and/or propose an equivalent version of these media in their best possible form. In the scope our study, we did not directly tackle the transformation of elementary media in broadcast environments but we rather relied on two currently deployed approaches which are mostly compatible with our adaptation requirements: the simulcast of several media alternatives and the broadcast of scalable media. The large deployment of MPEG codecs in the broadcast industry (typically MPEG-4 AAC and MPEG-4 AVC) and the recent development of the MPEG-21 adaptation framework are an opportunity to adapt elementary media along with their presentation based on a single adaptation decision-taking engine. Such a harmonized approach aims at facilitating the implementation of general-purpose adaptation engine by receiver manufacturers.

The underlying principle of our proposal is to combine the strength of a generic decision-taking engine for media-driven scene adaptation with the full control over presentation options which is guaranteed by explicit scene alternatives. The architecture of such a media adaptation engine enhanced by the addition of scene adaptation capabilities is illustrated in Figure 30. Thus, a typical media adaptation module (grey box) composed of an adaptation decision-taking engine and a media resource adapter [134] constitutes the core of our adaptation architecture. The media decision-taking engine selects an optimized media

adaptation process (e.g. scalable adaptation, bitstream-switching, transcoding) for a given environment context while the resource adapter is in charge of performing the adaptation operation on the media bitstream.

In order to guarantee the overall presentation quality to the end-user, a dynamic scene adapter module has been introduced in addition to the media adaptation module. This scene adapter module detailed in Figure 31 provides an additional presentation transmission channel and takes into account media-level adaptation decisions in the presentation layout. First, an enhanced decision-taking engine decides on the best presentation version based on author constraints. Then, presentation updates dynamically modify the user display according to fluctuating environment constraints.

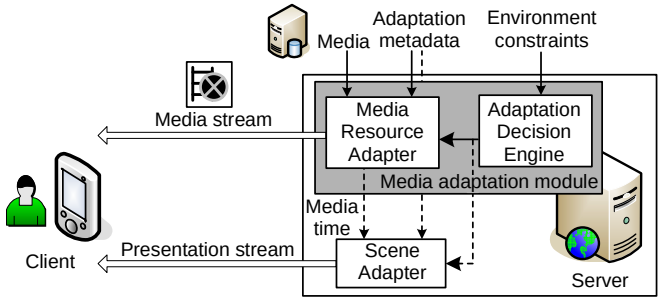


Figure 30: The enhanced adaptation architecture

The proposed decision-taking algorithm relies on a media decision-taking engine described in 5.2.1.1 including the `ConversionLink` tool, which we contributed to the MPEG standardization body and is now part of the MPEG-21 DIA standard. The inferred scene decision-taking engine that complements media-level decisions is explained in Section 5.2.1.2. Finally, our scene transformations through presentation updates are described in Section 5.2.1.3. A fine-grain control of the synchronization of presentation updates with adapted media streams is used to enable seamless switching of media bitstreams with different coding (transcoding) or switching to media streams with different modalities (transmoding).

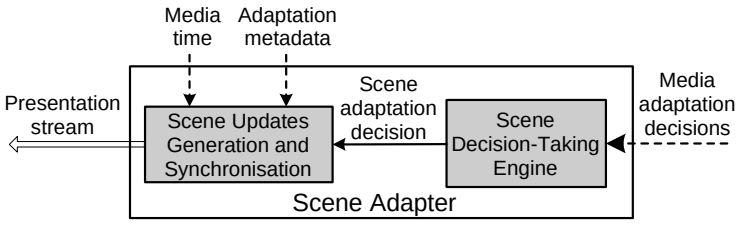


Figure 31: The scene adapter module.

5.2.1.1 The media decision-taking engine

Given the limited processing capabilities of some handheld devices used in broadcast environments, two main adaptation approaches have been envisaged for the adaptation of elementary media: the filtering of appropriate enhancement layers of scalable content and the selection of a suitable version among several content alternatives. In particular, all dynamic conversion processes have been discarded because of the heavy processing they require. Since all media alternatives are known beforehand, they have been explicitly described in the MPEG-21 Digital Item used for their presentation. An adaptation decision-taking engine, such as a compliant MPEG-21 DIA engine, is responsible for selecting the best option among all media versions and relies on an explicit mapping between environment descriptors (e.g. MPEG-21 UED) and these alternatives. For instance, the MPEG-21 DIA standard describes this mapping

as adaptation modules (MPEG-21 `AdaptationQoS`) which can be expressed as a lookup table. In particular, key layers of the video bitsteams are mapped onto explicit parameters so that a context-aware adaptation decision-taking engine can select relevant layers according to the usage environment. Such a straightforward adaptation decision-taking engine can be enriched with adaptation constraints (MPEG-21 UCD) which can be used to introduce some computable relations between adaptation parameters and adapted media versions. For instance, the maximum bitrate allocated to a video sequence might be limited to 80% of the available bandwidth.

Once an adaptation decision has been taken, output parameters (e.g. MPEG-21 `IOPin`) need to be fed into a resource adapter which will perform the actual media adaptation. This adaptation process can be simply described by referencing the original content to be adapted and the adaptation descriptors that will guide the adaptation process. For instance, two descriptors can be used in the scope of MPEG-21 DIA as illustrated in Figure 32: `BSDLink` and `ConversionLink`.

- The MPEG-21 `BSDLink` creates a connexion between a scalable bitstream, a bistream description (e.g. gBSD) which signals the characteristics of its adaptation layers and a transformation description (BSD Transformation) compatible with the bitstream description that features input parameters for its configuration.
- The MPEG-21 `ConversionLink` creates a connexion between a bitstream, an optional MPEG-7 resource description and a conversion act that can transform the bitstream according to input parameters.

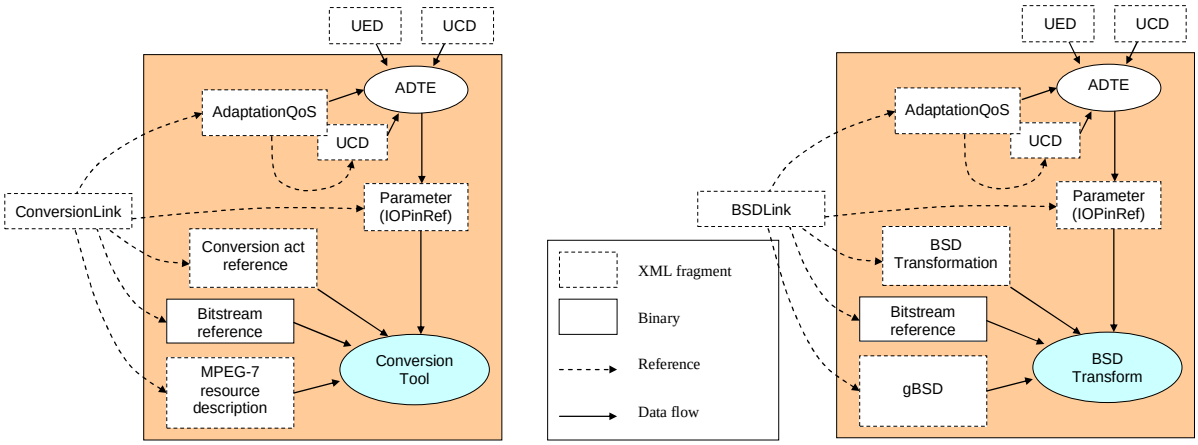


Figure 32: MPEG-21 BSDLink and ConversionLink description tools

The `ConversionLink` descriptor results from a contribution we conducted to the MPEG standardization body to harmonize media conversion tools with existing MPEG-21 DIA tools targeting the adaptation of scalable content. This tool has been standardized [67] and is now integrated in the latest release of the standard [66]. In practice, conversion acts are defined as registered media transformation processes defined outside the scope of MPEG. As a consequence, a media selector featuring a Boolean input parameter that enables filtering out a media item can be defined using a `ConversionLink` descriptor. Hence, a document (e.g. MPEG-21 DID) can describe a large panel of adaptation scenarios for an elementary media stream (scalable or not) that can be dynamically selected according to the decisions of a unique adaptation platform.

5.2.1.2 The inferred scene adaptation decision

A scene decision-taking processing consists in deciding the best overall presentation for the current set of media. Since broadcast scenarios require efficient adaptation techniques, we decided to transfer most of the complexity of scene transformations to the content creation phase in order to limit scene transformations to the replacement of a few scene attributes (Section 4.3.1). Hence, our approach relies

on scene format capabilities to define presentation alternatives, referred as branches in the scene tree. In practice, the scene has to integrate as much branches as possible dynamic adaptation scenarios.
Figure 33 shows a multimedia scene specifically designed to enable the dynamic branch switching from a video to a slideshow. For instance, these two scene alternatives might offer a full screen video sequence or a slideshow featuring smooth transparency effects between images and an interactive access to the last displayed images.

In our approach, the content creator is in charge of defining the mapping between the media state (media-level adaptation decisions) and the activated presentation alternatives (scene branches). Therefore, the scene decision-taking algorithm is straightforward and simply consists in selecting predefined presentation options from media-level adaptation descriptions. This content referencing can be performed through media descriptions identifiers (e.g. MPEG-21 DII [65]) that point to branch identifiers in the corresponding multimedia scene (e.g. MPEG-4 Object Descriptors ID). Our approach can also be used for the dynamic generation of multimedia presentations based on a format- and layout-agnostic structure (MPEG-21 DID). In that case, the scene adapter can be coupled with an automated scene generator which follows specific editorial rules.

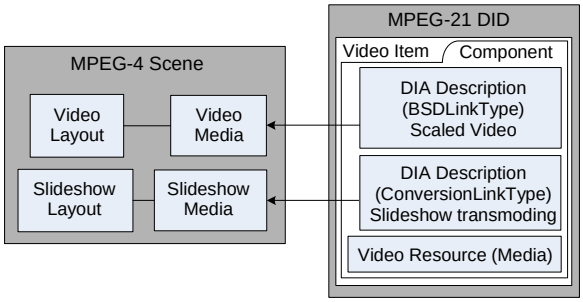


Figure 33: Example of a scene and its related MPEG-21 content description.

5.2.1.3 The scene transformation engine

Once the decision for a specific scene adaptation has been taken, the scene adapter module is in charge of modifying the presentation. This basically implies replacing the scene properties that control the visibility of the required scene option and synchronizing this scene transformation with adapted media streams. More precisely, the modification of the scene basically aims at two different goals: transition effects and branch switching.

- Transitions updates modify the display to prepare the coming media adaptation and the possible corresponding branch switching.
- Branch updates actually perform the media switching at the scene level. In any case, precise synchronization is required to achieve seamless and attractive transitions.

In order to perform these scene transformations, we decided to select time-based scene updates as defined in 4.3.1.2 due to our needs for a dynamic scene adaptation to a fluctuating context and in order to guarantee the finest synchronization with media-level adaptations. The synchronization model chosen in our approach takes advantage of the fact that a media adaptation buffer is used for content delivery or playback. For instance, a server necessarily needs to fetch and adapt the media bitstream in advance in order to deliver the content on time. This adaptation buffering time can be used to start transition effects as soon as the media adaptation decision is taken. The branch switching update is only applied when the adaptation actually becomes effective on the terminal.

This scheduling of presentation updates is depicted in Figure 34 in a video resizing scenario. A video bitstream-switching decision is taken at the server side at time T_1 but will effect at the output of the server at time T_2. Snapshots of the user display show scene enhancements along with video adaptation.

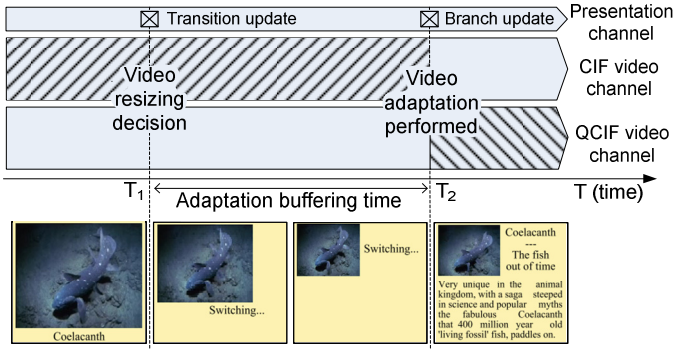


Figure 34: Example of a scene layout adaptation to video resolution.

5.2.2 Experiments and results

Our experiments conducted on the adaptation of scene properties in concordance with media-level adaptation decision have been performed in the scope of the DANAE IST European project[42]. This project targeted a complete framework able to provide end-to-end quality of (multimedia) service at a minimal cost to the end-user. Although the test bed was developed and integrated for unicast streaming scenarios, some of the results described in the following are also applicable for multicast scenarios.

5.2.2.1 The testing environment

The architecture illustrated in Figure 30 has been implemented and integrated in the DANAE platform. This platform is composed of a streaming server and an MPEG-4 client, compliant to MPEG-21, communicating with the RTSP/RTP streaming protocol. Several experiments have been conducted on the adaptation of multimedia services on both PC and PDA mainly relying on video adaptation techniques.

5.2.2.2 Adaptation efficiency

During our experiments based on the DANAE test bed, we have been able to demonstrate the dynamic adaptation of multimedia scenes with latency lower than 1 second in streaming scenarios (see our dynamic requirement in Section 5.1.4). Such reactivity results from the performance of the entire chain that is capable of detecting context changes rapidly (e.g. through RTCP reports), processes media data with short pre-fetching delays and plays presentations with a short buffering time.

In unicast streaming scenarios, the proposed approach has low processing overheads on the server side thanks to its straightforward scene decision-taking algorithm. Moreover, it is also transparent to mobile handsets in streaming scenarios since scene updates are interpreted as regular timed scene commands (replace, insert or delete) and do not require any specific context handling nor any specific scene processing engine. However, the difficulty raised by the implementation of an MPEG-21 DIA engine on handheld terminals for multicast scenarios has not been tackled. Even though we did not measure the performance of an MPEG-21 adaptation decision-taking engine on portable platforms, the complexity of the adaptation solving clearly increases as the number of adaptable media grows. As a consequence, acceptable overheads for our adaptation approach combining media selection and presentation adaptation directly depends on the ability of the MPEG-21 DIA implementation to efficiently handle multiple instances of the adaptation module processor. As a consequence, our low-overhead adaptation requirement (Section 5.1.3) is likely to fail because of our need for advanced multimedia services (Section 5.1.5)

Concerning our autonomy requirement (Section 5.1.2), all metadata referenced by the MPEG-21 DID could be transmitted along with media streams. Besides, distributed adaptation scenarios were demonstrated in DANAE by continuously transmitting gBSD data as metadata streams to operate the adaptation of scalable media in network adaptation nodes [35]. So, this would seem feasible to transmit gBSD data directly to receivers for a client-side adaptation. However, since our approach depends on

[42] http://danae.rd.francetelecom.com/

media-level adaptation decisions, the number of scene alternatives quickly grows in the case of presentations featuring numerous and highly-adaptive media. In that case, all adaptation descriptors and the initial scene featuring many alternatives would consume a significant bandwidth. However, this bandwidth remains limited compared to the usual bitrate of a single video stream.

5.2.2.3 Adaptation flexibility

Our scene adaptation framework allows the on-the-fly generation of a static presentation from an MPEG-21 DID. In that case, the scene generator is in charge of producing a presentation from media-level decisions according to simple editorials rules which have been automated. Additionally, multimedia scenes can also be prepared off-line and fed into the adaptation system as long as they offer presentation alternatives compatible with the nature of the adapted media. For instance, the media switching from a video to a slideshow requires the scene switching from a video node to a timed image node in the scene. Such adaptation scenarios have been demonstrated on DELL AXIM PDA by performing QCIF MPEG-4 SP video-to-video seamless bitstream switching or video-to-slideshow synchronized transitions. Additionally, the content creator can freely enhance the spatial, temporal and interactive properties of the rich-media presentation according to media-level decisions. These scene properties can be defined in each scene branch to enable a complete control over adaptation options. The experiment described in Figure 35 illustrates the scene adaptation scenario of a scalable video sequence composed of two PSNR layers (L_0 and L_1) to a slideshow featuring 3D effects. Snapshots of the user display illustrate the benefits of presentation updates during and after the bitstream-switching key points.

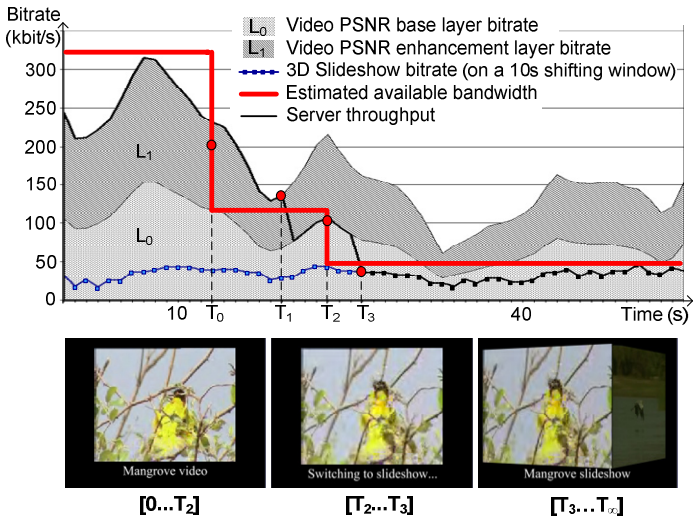


Figure 35: Example of a video summarization to slideshow.

5.2.3 Conclusion

The approach presented in Section 5.2 showed how existing adaptation architectures targeted for scalable media can be enhanced by including a scene adapter. The scene adapter module performs the mapping of the adapted media set onto optimized presentations designed by the content creator. This approach allows an improvement in the perceived quality of the adapted multimedia service and grants content providers more adaptation scenarios at a minimal cost.

The important concept developed in our work through our experiments is that of presentation transitions between the different adapted versions of media as illustrated in Figure 36. Media degradation can often be balanced with presentation enrichments (presentation switching updates). For instance, switching from a video to an image slideshow in order to save bandwidth can be counterbalanced with an interactive control over these slides. Scene transformations can also be introduced to create seamless transitions between the different adaptation states of elementary media (presentation transition updates). For instance, a VGA video can be progressively scaled down before switching to a QVGA configuration.

Both aspects, presentation switching and transition updates, have been successfully combined with the timed transformations of the multimedia service by relying on the same scene updates. This approach enables dynamic adaptation decisions but also ensure the synchronization of media-level and scene transformations which significantly contributes to the quality perceived by the user. These adaptation updates between presentation states have been integrated in our *Scalable MSTI* model by defining adaptation paths in our adaptation decision-taking engine. These configurable adaptation paths can be leveraged to trigger suitable transitions between presentation states.

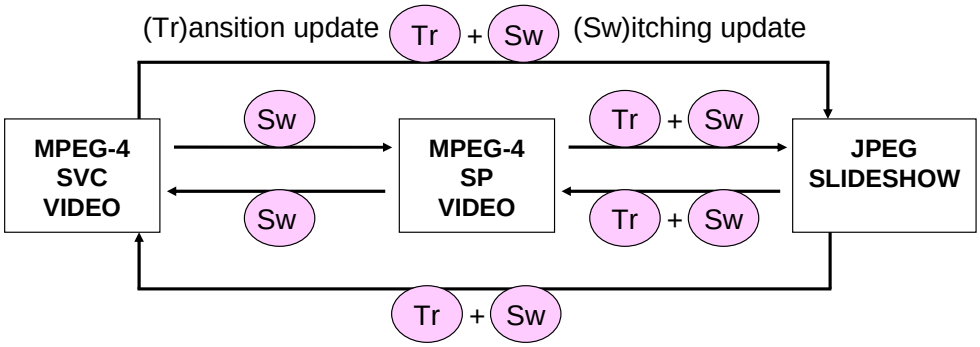


Figure 36: Linking presentation alternatives through adaptation updates.

Although the demonstrations presented in the Museon[43] museum were successful, several limitations have been identified.

First, media-driven approaches assume an audio and/or video content which is the main adaptation concern. In fact, these elementary media can require a significant bandwidth and their decoding (and rendering) process might consume a large part of the terminal's resources. However, it might also be useful to adapt a presentation to the user's context without adapting these key media. In that case, a pure media-driven approach should be combined with other scene adaptation approaches, such as scene plasticity (Section 4.2.4), to broaden the range of supported context parameters (e.g. targeting presentation size other than typical video resolutions).

Second, the adaptation complexity of our proposal increases as the number of media configurations grows. Advanced multimedia content featuring a large number of adaptive media would require significant overheads for the continuous processing of all the adaptation engine instances it would imply. Of course, it is still possible to narrow scene adaptation scenarios to the main media configurations. However, this limitation also shows that the shared management of the adaptation of several elementary media (MPEG-21 descriptors driving MPEG-4 scene) does not have the same objective as the single adaptation of a presentation including several elementary media (MPEG-4 scene driving MPEG-21 descriptors). On the one hand, media-driven scene adaptation mainly aims at proposing each identified elementary media in their best possible quality to the user. On the other hand, scene-driven approaches focus on the global audiovisual experience proposed to the user.

Third, the MPEG-21 adaptation framework is flexible enough to cope with the adaptation of advanced presentation. However, media-driven scene adaptation clearly targets context parameters that are related to audio, video or image decoding (e.g. bandwidth, codec capabilities, decoding resolution, and display resolution). Other parameters from the user's context should also be taken into account when adapting the properties of a presentation such as the available interactive means or the presentation legibility which are directly related to context-oriented constraints as described in Section 5.3.

In fact, these three limitations are overcome in our *Scalable MSTI* approach described in Chapter 6 by defining a scene-driven decision-taking algorithm that extends media-driven adaptation decisions to all presentation characteristics.

[43] http://www.museon.nl

5.3 Context-driven presentation selection

Among all the research done in the scene adaptation field, the adaptation of the spatial layout of elementary media is an interesting topic because it raises severe problems. In practice, spatial scene adaptation allows the presentation of a multimedia scene on different devices with various screen sizes, resolutions and aspect ratios. Existing scene description languages such as (i.e. SVG [118], MPEG-4 BIFS [59], SMIL [119], HTML [120] provide some basic spatial adaptation mechanisms such as automatic flow/grid layouts for text and/or graphical objects, scalable representation of vector graphics, etc. Additionally, these formats can be extended [17] or semantically annotated [76][135] to automatically perform enhanced adaptation. However, more author-driven adaptation techniques are required to strictly preserve the legibility of the text content or the graphic standard as illustrated in Figure 37, Figure 38 and Figure 39.

Figure 37: Two examples of unsuitable presentation adaptation (original content on the left).

Figure 38: Examples of straightforward but questionable presentation adaptation.

Figure 39: Examples of semantic presentation adaptation requiring an editorial validation.

Our broadcast-friendly scene adaptation approach, published in the proceedings of the International Conference on Multimedia & Expo [5], consists in using incremental scene description updates for the context-driven adaptation of multimedia presentations. In the following, the principle of this approach is further explained in Section 5.3.1. Section 5.3.2 describes a concrete example illustrating the use of adaptation descriptors that target progressive screen resolutions. In this example, a scene adapted to increasing resolutions provide visual enhancements to large screen receivers. Our experiments described in Section 5.3.3 show the benefits of our approach in terms of bandwidth, memory and processing time on a set of typical multimedia contents.

5.3.1 Principles

The authoring of a presentation designed to natively integrate adaptation options is the best possible way to ensure content quality in multiple usage conditions. As proposed in our media-driven scene adaptation

introduced in Section 5.2, these scene versions can be prepared according to technical capabilities of the media to be adapted. Another option consists in enumerating specific target devices or network conditions and generating a specific scene alternative for each of them. We call this approach context driven, as opposed to the previous media-driven approach. This context-driven adaptation approach has the advantage to leverage the full capabilities of scene descriptions. Indeed, a multimedia scene can be interpreted and deeply transformed contrary to a video sequence which requires analysis and metadata extraction. Additionally, even if it is obviously impossible to capture all possible user contexts, the definition of a set of optimal presentations, according to the editor, for specific usages allows accurately addressing specific (but frequent) configurastions while other configurations simply match the nearest version that does not exceed any of their constraint parameters. The adaptation of presentations to the screen display of multimedia devices is a typical use case that our proposal tackles and that fits our broadcast adaptation requirements.

The underlying principle of our proposal is to rely on the intrinsic plasticity of standardized scene formats (e.g. automated layout, lossless vector scaling described in Section 4.2.4) and to introduce optimized scene alternatives. This new type of scene transformation aims at factoring the redundancy between scene alternatives in order to reduce the weight of these descriptions which is a critical cost in broadcast environments. First, a context-driven decision-taking selects an optimal scene version based on an explicit mapping between constraints and scene properties (Section 5.3.1.1). Then, presentation updates are progressively applied until the optimal adapted version is reached (Section 5.3.1.2).

5.3.1.1 Scene and context matching

The experiments we conducted on media-driven scene adaptation (Section 5.2) showed that even if the controlled adaptation of a presentation driven by adaptation decisions on media can rely on a standardized adaptation framework (e.g. the MPEG-21 framework), it still suffers limitations when the adaptation of rich-media content is considered. In particular, the number and the diversity of media options quickly lead to a significant number of scene possibilities. Furthermore, the organisation of several adaptive media in a common presentation implies defining independent scene branches. This leads to impenetrable barriers between some scene properties which narrow authoring possibilities since only the sub-scene related to the activated media can be adapted. For this reason, we decided to rely on a scene-based adaptation decision-taking engine that does not explicitly depend on media alternatives: the appropriate scene alternative is directly selected by matching context parameters and its scene adaptation parameters.

The authoring of an adaptive presentation with several scene alternatives requires defining the adaptation parameters that need to be addressed. An efficient option consists in selecting all targeted devices and specifically addressing their characteristics (e.g. an iPhone for instance). A more flexible option consists in identifying only typical adaptation parameters that are applicable in some targeted usage scenarios (e.g. display resolution, memory requirements, interactive means, information details…). In that case, the adaptation decision-taking still remains cumbersome since multiple alternatives for the same adaptation parameters need to be compared. For instance, how to decide between two scenes alternatives matching the user context: is it more important to minimize the content memory footprint or give access to more information? Our proposal consists in defining *steering* adaptation parameters that are to be maximized during the adaptation process. Each of these steering parameters has a priority so that multiple steering adaptation parameters can be handled by addressing the highest prioritized constraints before the others. Additionally, *threshold* adaptation parameters can be defined. These threshold parameters do not drive the adaptation decision-taking engine but disable adaptation options when they are not satisfied. Hence, if the quantity of information is defined as a steering adaptation parameter and memory requirements are threshold parameters, the decision algorithm will select the scene alternative that maximize the quantity of information and that still matches memory constraints.

5.3.1.2 Progressive scene updates

Scene update is a mechanism available in several languages (Flash, MPEG-4 BIFS [59] and MPEG-4 LASeR [62]) that allows the streaming of timed modifications of a scene as described in Section 4.3.1.2.

In our approach, we propose to use these updates to achieve efficient scene transformations for adaptation purpose in broadcast scenarios. Indeed, the transmission of multiple scene alternatives in broadcast environments requires repeating all data continuously. Since all of these alternatives constitute different versions of the same presentation, some redundancy exists between them and could be reduced in order to lighten transmission costs. A first step consists in reducing this redundancy using scene attribute spreading techniques (Section 4.3.2). Our proposal goes beyonds by expressing a set of scene alternatives into a sequence of scene updates. Scene alternatives are ordered in such a way that only the difference between a scene alternative and its predecessor is specified. The efficiency of the ordering of scene alternatives directly depends on the redundancy of scene properties. As a consequence, a bandwidth-optimal ordering needs to be computed during the content publication phase by comparing XML documents based on state-of-the-art approaches [100]. Then, a predictive coding for updated XML elements and attributes has to be applied.

In this approach, the adaptation decision-taking process and the scene transformation are, by default, independent. First, the adaptation decision-taking engine considers every possible alternative tagged with context descriptors and selects the one that maximizes its *steering* adaptation parameters one after the other according to their priority. During this process, or prior to this process, all alternatives that fail to address *threshold* adaptation parameter are discarded. Then, the selected scene alternative is generated by successively applying all scene updates until the selected scene alternative is reached.

Even though our approach does not mandate progressive scene updates to be ordered according to steering adaptation parameters, such a direct link might be meaningful when few steering adaptation parameters are considered and naturally imply a progressive evolution of the presentation. The screen dimension of a receiver is an interesting case since a large presentation can be created from the materials of a smaller version. Hence, the multimedia content can be built by concatenating the scene, designed for the smallest resolution, with adaptation scene updates which either adjust the layout of existing media or insert new media elements. In practice, upon reception of these updates, the terminal applies scene commands in the given order until it detects that the next update resolution will exceed its capabilities. The detection and filtering mechanism can be achieved by inserting specific updates between adaptation scene updates to signal the spatial characteristics required to process the related data. Such signaling is illustrated in Figure 40.

Figure 40: Spatial scene updates and bitstream organization.

5.3.2 An MPEG-4 BIFS example

The multimedia content provided in this section is described using the MPEG-4 BIFS language [59]. It shows a 3-day weather forecast (today, tomorrow and the day after tomorrow) that gives an interactive access to detailed information in the morning and in the evening (temperature, illustration icons, textual description, confidence rating) as illustrated in Figure 41. Text elements, interactive arrows, rectangular top captions are all described using scalable graphics elements. Only the sun and cloud icons are PNG images. The scene corresponding to the presentation depicted in this section features a new adaptation tool that we contributed to the MPEG-4 BIFS standard that is described in Appendix C. This new tool, called `EnvironmentTest`, can be used to signal key state updates (Section 5.3.2.1) but also intermediate state updates (Section 5.3.2.2) that allows the adaptation of our example to several screen configurations.

Figure 41: Example of a weather forecast service.

5.3.2.1 Key-state scene updates

Media instances are likely to be introduced progressively as the scene dimensions increase. Depending on designer wishes, a multimedia scene may have a limited number of *key* scene states corresponding to typical display resolutions. For each resolution, new media might be introduced while the presentation of existing media is modified as illustrated in Figure 42.

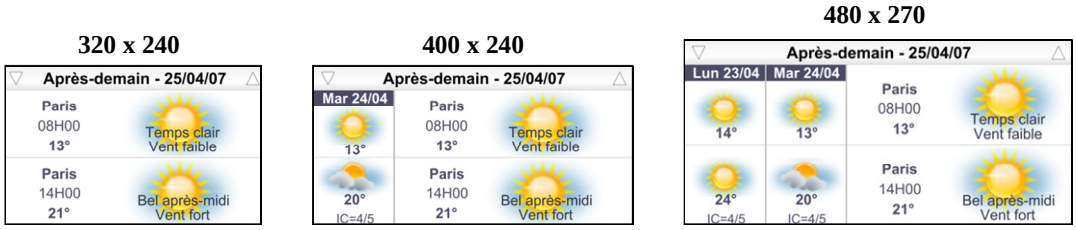


Figure 42: Example of weather forecast service prepared for three typical resolutions.

The media insertion updates and layout updates which constitute key state scene updates rely on identifiers, signaled by a `DEF` attribute in MPEG-4 BIFS, which are defined either in the initial scene or in previous adaptation scene updates. Each adaptable media introduced in the current scene must define such identifiers in order to allow the dynamic modification of its main spatial scene properties as defined in Section 3.2. In the MPEG-4 BIFS example described in Code 34, all media referenced in the multimedia scene are introduced inside a `<TransformMatrix2D>` element which enables classical geometric 2D transformations.

```
OrderedGroup { children [
  DEF SWITCH_DEF_MEDIA Switch {
    whichChoice -1
    choice [
      DEF SHAPE_MEDIA_COMPONENT_1 Shape {
        appearance Appearance {
          material Material2D { filled TRUE emissiveColor 1 1 1 }
        }
        geometry Rectangle{ size 320 240 }
      } DEF SHAPE_MEDIA_COMPONENT_2 ...
   ]}
  # Insert point for media instance
  DEF T2D_INSERT_POINT Transform2D { children [ ] }
 ]}

AT 0{
  # 320 x 240 layout
  APPEND TO T2D_INSERT_POINT.children
  Transform2D{ children [
    DEF TM2D_MEDIA_COMPONENT_1 TransformMatrix2D{ children[
      USE SHAPE_MEDIA_COMPONENT_1
    ]} ]}

  # 400 x 240 layout
  APPEND TO T2D_INSERT_POINT.children
  DEF ADAPT_400pxw_240pxh Conditional { buffer {
      APPEND TO T2D_INSERT_POINT.children
      DEF TM2D_MEDIA_COMPONENT_2 TransformMatrix2D{ children[
          DEF T2D_MEDIA_COMPONENT_2 Transform2D {
            translation 148 70
           children [ USE SHAPE_MEDIA_COMPONENT_2 ] } } } ] } }
```

Code 34: Media insertion for key states scene updates in MPEG-4 BIFS.

5.3.2.2 Intermediate-state scene updates

Key state scene updates define typical scene configurations designed to address a set of targeted usage environments (screen resolution of 320x240, 400x240 and 480x270 in our example). However, other intermediate resolutions need to be addressed to improve the adaptation granularity. As far as spatial scene properties are concerned, the layout of media can be interpolated by relying on two versions of the same presentation at different resolutions as illustrated in Figure 43. Based on our approach, *intermediate* state scene updates can be defined to create a finite number of additional states between two key states. These intermediate scene updates are signalled as any other adaptation updates as illustrated in Code 35.

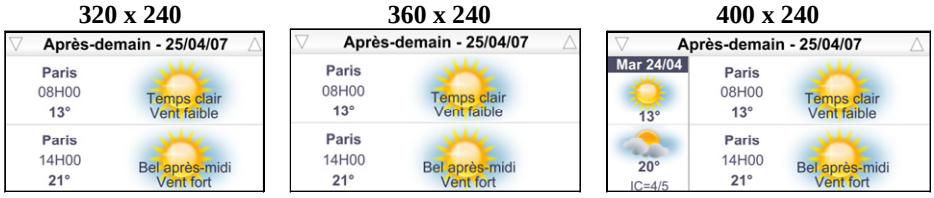


Figure 43: Example of weather forecast service generated with one intermediate resolution.

```
AT 0{

  // Adaptation update
  APPEND TO T2D_INSERT_POINT.children
  DEF ADAPT_K1_P1 Conditional { buffer {
    REPLACE TM2D_MEDIA_COMPONENT_1.mxx BY 1.12
    ...
  } }

  // Adaptation update trigger
  APPEND TO T2D_INSERT_POINT.children
  DEF C_360pxw_240pxh Conditional { buffer {
    REPLACE ADAPT_K1_P1.activate BY TRUE
  }}

  // Checks width against 400 pixels
  APPEND TO T2D_INSERT_POINT.children
  DEF ENV_360pxw EnvironmentTest {
    enabled TRUE parameter 2 compareValue 360
  }

  // Checks height against 240 pixels
  APPEND TO T2D_INSERT_POINT.children
  DEF ENV_240pxh EnvironmentTest {
    enabled TRUE parameter 3 compareValue 240
  }

  // check resolution against 360x240 and trigger BIFS adaptation update
  APPEND TO T2D_INSERT_POINT.children
  DEF C_360pxw Conditional { buffer {
    REPLACE V_360px_240pxh.Factor1 BY 0.5
    REPLACE V_360px_240pxh.inSFInt32 BY 1
  } }
  APPEND TO T2D_INSERT_POINT.children
  DEF C_240pxh Conditional { buffer {
    REPLACE V_360pxw_240pxh.Factor2 BY 0.5
    REPLACE V_360pxw_240pxh.inSFInt32 BY 1
  } }
  APPEND TO T2D_INSERT_POINT.children
  DEF V_360pxw_240pxh Valuator {
    Factor1 0 Offset1 -0.5
    Factor2 0 Offset2 -0.5
    Factor3 0 Factor4 0 Sum TRUE
  }
  INSERT ROUTE ENV_360pxw.valueSmaller TO C_360pxw.activate
  INSERT ROUTE ENV_240pxh.valueSmaller TO C_240pxh.activate
  INSERT ROUTE V_360pxw_240pxh.outSFBool TO C_360pxw_240pxh.reverseActivate
}
```

Code 35: Adaptation update for intermediate states in MPEG-4 BIFS.

5.3.3 Experiments and results

In this section, we present experiments that have been conducted on adaptable T-DMB multimedia radio services[44]. These experiments have been conducted on a set of typical scenes described in Section 5.3.3.1. Quantitative results are depicted in the following by focusing on processing requirements (Section 5.3.3.2), memory requirements (Section 5.3.3.3) and bandwidth requirements (Section 5.3.3.5). All these results compare our approach with adaptation approaches based on alternative selection (Section 4.2.3.1) and dynamic interpolation techniques (Section 4.2.4.1). Finally, the flexibility of our adaptation approach is described in Section 5.3.3.5.

5.3.3.1 The testing environment

Test scenes are composed of M media. These media may be used as several instances in the scene, of which N are to be adapted. A multimedia scene may have K *key* scene states and P additional *intermediate* states may be created between two key states leading to K+P*(K-1) states. Our quantitative criteria (processing, memory, and bandwidth) are overheads due to the adaptation functionality. They may depend on the number of adaptable media instances (N), the number of adaptation states (K, P) but also from the ratio R between the weights (in Kbytes) of media data (images, text, and graphics) compared to scene structures. A set of interactive scenes used during our experiments can be visualized in Figure 44. Their characteristics are summarized in Table 3.

Table 3: Characteristics of digial radio test scenes.

Sequence	(a)	(b)	(c)	(d)	(e)
M	8	13	16	31	39
N	8	19	9	19	58
K	2	3	3	3	3
R	2%	21%	2%	2%	25%

Figure 44: Digital radio test scenes with various characteristics

The multimedia scenes have been described using the MPEG-4 BIFS standard and the same visual behavior has been created for each adaptation approach: adaptation scene updates (*Update*), alternative-based selection (*Switch*) and dynamic interpolation-based adaptation (*Interpolations*). All adaptation algorithms have all been implemented in JavaScript in order to have a fair comparison of processing requirements, even though some approaches could easily be implemented natively (e.g. adaptation scene updates). As a consequence, all elementary media insertions for key states have been defined using MPEG-4 BIFS `<Switch>` element which is configured according to the adaptation decision computed by the JavaScript engine. Additionally, intermediate states have been defined according to the specificities of each studied approach as illustrated in Figure 45:

- For *Switch* approach, supplementary `<Switch>` children have been introduced according the desired adaptation granularity (P). In that case, the JavaScript engine is in charge of selecting the best `<Switch>` choice (key state of intermediate state) based on the targeted resolution.
- For the *Interpolation* approach, one `<Switch>` child has been introduced between each key state pairs. In that case, the JavaScript engine is in charge of selecting the best key state based on the targeted resolution. If no exact matching can be found, the spatial properties of the closest intermediate state are dynamically configured by the JavaScript engine by resolving the interpolation function proposed in [36] from the layout of its neighbouring key states.

[44] Test content and a video demonstration can be found here: http://icme2008.tnt.uni-hannover.de/1835/

– For the *Update* approach, no supplementary `<Switch>` child has been introduced. All key states have been kept for media insertion and adaptation updates have been described according to the desired adaptation granularity. In that case, the JavaScript engine is in charge of selecting the best key state based on the targeted resolution. Then, it applies the spatial properties of adaptation updates until the best configuration is reached.

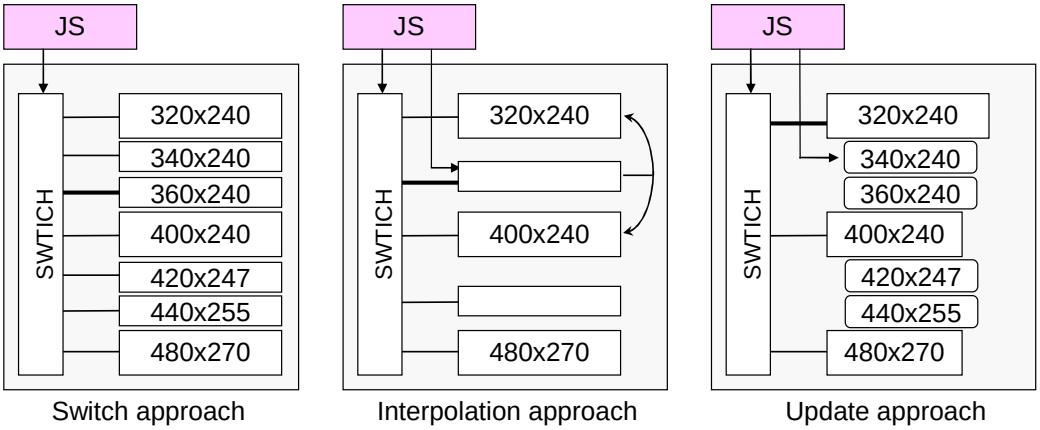


Figure 45: Test scenes (K=3 and P=2) for the switch, interpolation and update approaches

Measurements have been performed on a smartphone (SPV C500) using GPAC [81]. The initialization cost for the JavaScript engine is omitted from these results and the adaptation processing time corresponds to a complete presentation cycle (including decoding, compositing and rendering). The processing overhead represents the extra computing time (ms) dedicated to handle the adaptation process. The memory overhead takes into account the extra data (kbytes) that need to be loaded on the terminal because of the adaptable feature of a multimedia scene. The bandwidth overhead shows the bitrate (kbits/s) induced by the delivery of adaptation data.

5.3.3.2 Processing efficiency

In terms of adaptation processing overhead, Figure 46 depicts a comparison of interpolations, alternatives and our proposed scene updates with P intermediate states on scene (e), as introduced in Table 3, which has three key states (K=3). It can be noted that the processing of the interpolation approach is quite CPU demanding compared to others. It can also be noted that the handling of scene structures needed for the switching approach has a significant impact when P increases.

Figure 46 also shows that the scene updates approach has progressive CPU requirements contrary to the other approaches with a similar P value. Therefore, adaptation using scene updates is a very competitive approach for low-end digital radio terminals (with a small resolution) while it is equivalent to other approaches on high resolution terminals. As explained in Section 5.3.3.5, results below are used for a fair comparison and could be optimized.

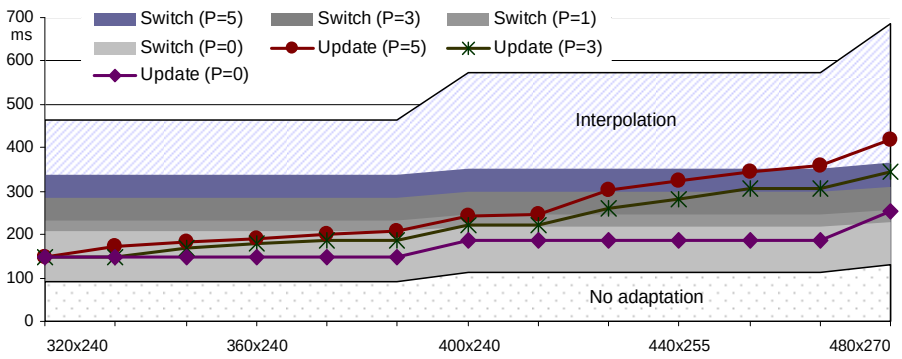


Figure 46: Adaptation processing overheads on (e).

5.3.3.3 Memory efficiency

Concerning memory requirements, scene updates do not incur any overhead contrary to other options. Table 4 gives memory overheads for several test scenes. This is mainly due to the fact that only relevant media elements and scene structures are loaded in the player scene manager when using adaptation scene updates. Besides, multimedia scenes with significant scene structures (i.e. (b) and (e) as described in Table 3) highlight the limitations of spatial adaptation using the *Switch* approach with more than 10% memory overheads with quite a few scene alternatives (9 alternatives in that case since K=3 and P=3). It could be argued that an optimized multimedia player could only load alternatives when activated. This is not implemented in the GPAC player: only media which are part of an inactivated branch of a `<Switch>` node are not loaded. Indeed, all scene elements can be updated over time, whether they are loaded or not. As a consequence, it would be difficult to reduce the memory overheads of Table 4 which mainly reflect the costs of the scene structure.

Table 4: Memory adaptation overheads.

Sequence	(a)	(b)	(c)	(d)	(e)
Updates	0%	0%	0%	0%	0%
Interpolations	1%	5%	0.9%	2%	**12%**
Alternatives (P=3)	2%	**10%**	1.8%	3.7%	**21%**
Alternatives (P=5)	3%	**15%**	2.6%	5.3%	**30%**

5.3.3.4 Bandwidth efficiency

The bandwidth required to broadcast (a), (b) and (e) multimedia services is illustrated in Figure 47. The bitrate required to broadcast images has not been taken into account in any approach (including simulcast) since images can always be referenced to avoid any data duplication. The scene is broadcasted every 500 ms (carousel period) and no transport overhead is considered. The benefits of all adaptation approaches can be compared to the simulcast of independent presentations corresponding to their key states.

In Figure 47, it can be seen that interpolations and scene updates roughly consume the same bandwidth when few intermediate states are defined (P<3). Furthermore, thanks to the use of our update-based approach, a bandwidth-optimized scene broadcast (noted as Update*) can even be defined using two different carousels. A base carousel only repeats data for key adaptation states (every 500 ms) while an enhancement carousel provides intermediate adaptation states (every 2 s).

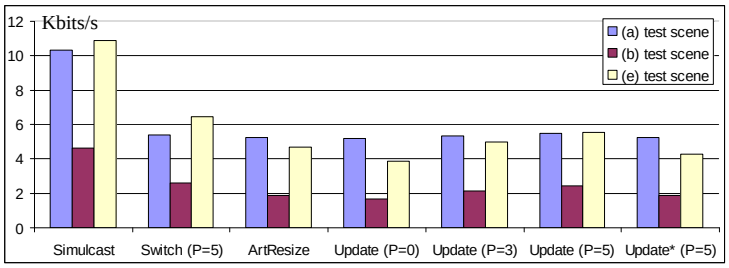


Figure 47: Bandwidths of adaptable multimedia scenes (a) (b) (e).

5.3.3.5 Adaptation flexibility

The strength of our approach is that it has an excellent flexibility since appending new elementary media is natively supported and no particular adaptation scene structure is required. Indeed, the use of scene updates do not mandate that presentation alternatives and possible elementary media versions are described in the scene that is provided to the user (by relying on specific scene structures for instance): they are transmitted and integrated in the scene in the same way as content is updated over time. However, the accuracy of the adaptation process against the user's context, i.e. adaptation granularity, should not be minimized by technical requirements. Hence, an adaptation technique that addresses any spatial configuration is more attractive than one that focuses on a set of typical screen resolutions. In the

interpolation approach, the adaptation process is in charge of providing a complete adapted layout to each spatial configuration. In our approach, only a finite number of resolution are targeted and scaling techniques can be additionally used to address others terminal requirements. In order to achieve fine adaptation granularity, two configurations can be proposed.

First, the number of intermediate state can be significantly increased. This does not imply any authoring efforts since interpolation algorithms can applied to generate these adaptation updates. However, the progressive (and possibly overwriting) layout scene updates have an impact on the processing required for high resolution terminals. Indeed, numerous writing accesses to the scene tree decrease overall adaptation efficiency. Furthermore, even though predictive coding is used, the consumed bandwidth still depends on the number of targeted screen resolutions.

Second, our approach can be combined with scene plasticity (Section 4.2.4) as natively implemented on receivers. In that case, adaptation scene updates allows the selection of an author-driven presentation alternative which might suffer from minor modifications through scene plasticity.

5.3.4 Conclusion

The approach presented in this section shows how scene updates can be used to achieve the context-oriented adaptation of multimedia services in broadcast environments. Our experimental results showed that the spatial adaptation through scene updates has excellent performances on constrained handsets with small display, low processing power and little memory. Additionally, it was shown that adaptation scene updates can grant content providers with fine-grained spatial adaptation or coarse-grained spatial adaptation depending on the acceptable bandwidth costs. Although our experiments have been conducted using the MPEG-4 BIFS scene format only, scene updates can also be implemented in other languages, as previously explained in section 4.3, by changing spatial properties through `<links>` in the NCL format, for example, and by triggering scene updates using editing commands in that case.

A major finding in our experiments was that scene adaptation updates constitute an incremental scene transformation as illustrated in Figure 48, which overcome the limitation of the redundancies introduced by the scene alternatives approach (Section 4.2.3). Indeed, although it may seem evident that evaluating a set of rules that simultaneously select several scene alternatives is more efficient that evaluating a set of rules that individually select each scene alternative, this was not obvious. There could even be some cases where dividing this evaluation could be advantageous for flexibility reasons. Additionally, scene plasticity (Section 4.2.4) based on customized interpolation-based scene adaptation has shown some significant overheads for spatial adaptation that will necessary increase when using constraint-based decision-taking engine. As a consequence, scene alternatives identified through resolution labels and dynamically generated using incremental scene updates succeeded in fulfilling all of our adaptation requirements on the screen resolution use case.

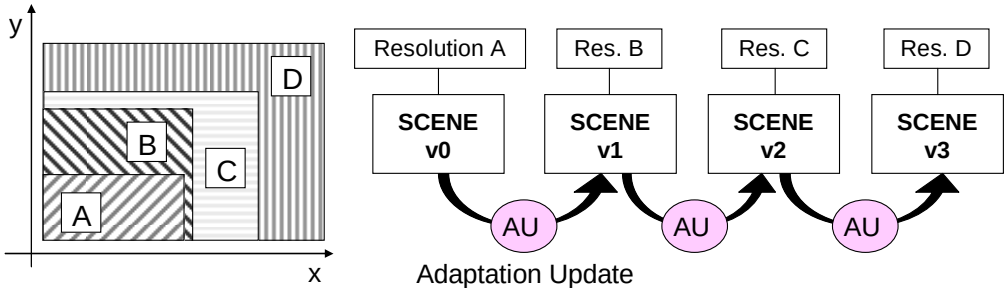


Figure 48: Generating presentation alternatives through adaptation updates.

This analysis led us to the use of these incremental scene updates as the core adaptation transformations of our *Scalable MSTI* model. In short, we propose to express a set of presentation alternatives designed for several adaptation scenarios as scene updates (and not scene alternatives) by describing the set of scene alternatives corresponding to all these presentation alternatives as a sequence of scene updates.

Although the adaptation of presentations according to the display resolution is a usual scenario, the proposed adaptation approach has two main limitations.

First, although our adaptation decision-taking engine can handle several parameters from the user context, the organization of scene properties into progressive adaptation scene updates becomes less efficient as the number of steering adaptation parameters increases. For instance, a scene can be first adapted according to the resolution of the screen and then selected according to the user's level of interest. In that case, a highly-detailed presentation for a small resolution will probably include media components that might not be available in the low-detailed presentation for a larger resolution. In the same way, the organization of the progressive presentation characteristics of scenes alternatives can be broken down by introducing specific scenes targeting a different aspect ratio (e.g. from landscape to portrait). In both cases, the performance of our adaptation approach falls back to a level that common alternative-based scene adaptation can reach. For this reason, our *Scalable MSTI* approach introduces three independent but flexible steering adaptation parameters for the scene adaptation decision-taking engine (*Spatial*, *Temporal* and *Interactive* adaptation parameters) in order to efficiently cope with multiple steering adaptation parameters.

Second, the resolution of a receiver screen remains static in many cases but highly variable user context parameters should also be taken into account. In that context, it should be possible to revert a scene adaptation transformation. Contrary to the selection of scene alternatives, incremental scene updates cannot be easily cancelled. A complete reload of the presentation would not be an acceptable option for highly dynamic usage environments. Instead, it should be possible to undo scene updates in order to enable seamless transitions between adaptation states. For that purpose, specific scene updates (*Random Access Layers*) have been introduced in the *Scalable MSTI* approach described in Chapter 6. These scene updates enable a random access to some key states of the presentation with an optimized memory footprint.

Chapter 6 The Scalable MSTI model

The approach proposed in this chapter is the core of our dissertation and consists in introducing new scalability concepts for multimedia scenes. Our *Scalable MSTI* model enables the adaptation of multimedia presentations according to the user's context by defining *a generic, autonomous and low-overhead transformation process that guarantees the controlled adaptation of advanced multimedia services* (see Section 5.1).

Our research was driven by industrial constraints (see Section 1.4) that tend to minimize the implementation efforts required to move from a one-service-fits-all approach toward several flavors of the same services. Indeed, the large ecosystem of existing authoring facilities, transportation hints and playback optimizations cannot be ignored for adaptation reasons. In practice, these aspects even remain a priority for content producers since the introduction of adaptation features into a multimedia delivery chain is not harmless. In particular, authoring paradigms cannot be easily changed nor upgraded. Streaming or broadcasting standards evolutions must always ensure backward compatibility with non-adaptable services already deployed. Finally, playback requirements dedicated to adaptation processing represent implementation efforts that must be marketed to the end user with a competitive added value compared to costs overheads. Based on these observations, one of our fundamental design choices differs from many state-of-the-art document models intended for adaptation such as ZYX [24], AHM [48] or Madeus [79] in that we do not assume any meta-model (see Chapter 2). Instead, the *Scalable MSTI* model defines non-semantic extensions for existing scene formats such as SVG [118], MPEG-4 BIFS [59], SMIL [119] or HTML [120] to express scene adaptation capabilities. These extensions are compatible with non-adaptive multimedia platforms. In short, our scalable model for multimedia scene adaptation consists in dividing the scene properties (see Chapter 2) of several authored presentation alternatives into a hierarchical structure, also called an adaptation graph.

In the following, the different components of our scalable model are described. First, the separation of the scene structure (*Media* description) from the scene properties is described in Section 6.1. Then, the splitting of scene properties into STI components (*Spatial*, *Temporal* and *Interactive* descriptions) is explained in Section 6.2. Finally, the organization of the *STI* components of multiple adapted presentation alternatives in progressive STI scalable layers is described in Section 6.3. These abstract scalable layers enables the generation of an adaptation graph (our document model) designed to drive the adaptation of the multimedia content to various usage environments. For illustration purpose, several scalable multimedia presentations examples are provided in Section 6.4 and conclusions drawn from our experiments are given in Section 6.5.

6.1 Scene transformation principles

Our *Scalable MSTI* model is a novel approach to represent multimedia scenes. The scene descriptions associated with this model are based on their targeted language. Each scene description expressed in a given language is split according to its media structure and its scene properties. The MSTI scene descriptions are divided into four components: *Media*, *Spatial*, *Temporal* and *Interactive* descriptions. The *Spatial* description defines the layout of all media components that are part of the document; the *Temporal* description provides the timing of the multimedia presentation and of each elementary media; and the *Interactive* description adds interactive aspects to the multimedia document. In the MSTI model, these three *Spatial*, *Temporal* and *Interactive* (STI) components are all related to the same *Media* description and possibly share logic elements as illustrated in Figure 49.

In order to regenerate a scene description from an MSTI scene representation, scene transformations are required. In the following, the principles of the scene description transformations of the *Scalable MSTI* model are described. First, the *Media* description, which constitutes the fulcrum of our scene transformations, is introduced in Section 6.1.1. Then, the updates that describe the scene properties of STI descriptions are specified in Section 6.1.2.

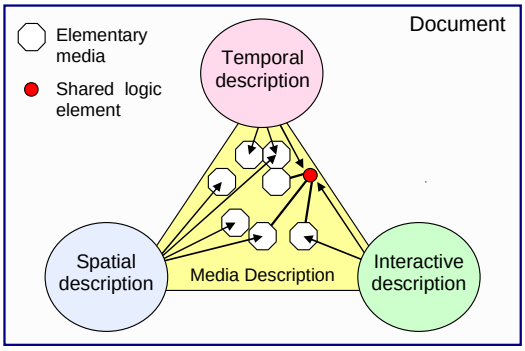


Figure 49: A multimedia document in the Scalable MSTI model.

6.1.1 The Media description

The *Scalable MSTI* model is based on state-of-the-art multimedia document design that clearly separate media and logical structures from the multimedia presentation. This presentation-agnostic view of a document is a common authoring practice: e.g. defining the *Base* document using presentation agnostic language like XML (and even to some extent HTML) and the layout of web pages using W3C Cascading Style Sheets [123]; or using SMIL Time Sheet [127] to apply temporal properties to an existing document or using an external JavaScript to add interactive behaviors. In the MSTI model, selected media components and their metadata are specified (or referenced) in the *Media* description. The *Media* description constitutes a structured XML document that can be queried and repurposed independently from its presentation. With such a structure, media components can be associated with metadata such as defined in W3C Resource Description Framework [117] in order to ease semantic search and can also be used in several application scenarios where their presentation will be different (possibly using the `defs` element in SVG). The *Media* description is expressed in the targeted language such as SVG in Code 36.

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.2"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"/>

  <metadata id="meta-rdf">
    <rdf:RDF
      xmlns:dc = http://purl.org/dc/elements/1.1/>
      <rdf:Description
        dc:date="2008-10-04"
        dc:publisher="RTL"
        dc:language="fr"
        dc:Title="DMB Digital Radio at Paris Motor Show 2008">
      </rdf:Description>
    </rdf:RDF>
  </metadata>

  <g id="g_0">
    <rect id="rect_0"/>
    <video id="video_0" xlink:href="salonAutoRTL.h264"/>
    <img id="img_0" xlink:href="koleos-RTLNumerique.jpg"/>
    <text id="text_0" xlink:href="pressRelease.svg#text"/>
  </g>
</svg>
```

Code 36: A Media description including RDF metadata in SVG.

The *Media* description specifies or references the media components that compose the multimedia document. This includes not only audio sequences, video clips, or images, but also text, graphic elements or any configurable sub-scene as defined in Section 3.1.2. The *Media* description also includes the structures and the application logic that define the relationships between these media components. For instance, if two images have been included in the same document to be shown as a slideshow, the display order and priorities should be described in the *Media* description since they do not depend on presentation choices. In other words, the image presentation order is the same if the slideshow is automatically animated or if transitions are triggered by the user. An example of *Media* description (using SVG and JavaScript) is given in Code 37 for an image gallery composed of 6 images. It should be

noticed that we use JavaScript code in this SVG example because this slideshow requires it, but our approach does not mandate the use of JavaScript code.

The *Media* description of the *MSTI* model could be compared to the 'logical' model of Madeus [79] with two main differences. First, the language used to define the *Media* description is the same as the final presentation language. Therefore, no specific syntax needs to be defined at the *Media* level. Second, the *Media* description does not contain the intrinsic properties of media elements (e.g. duration, size). It rather provides information related to their semantics (e.g. an image gallery). Hence, the *Media* description of the MSTI model can be seen as a multimedia document that does not include any presentation means.

```
<svg id="root" version="1.1">
 <script>
   image0 = document.getElementById("img_0");
   …
   image5 = document.getElementById("img_5");
   rect0 = document.getElementById("rect_0");
   …
   rect5 = document.getElementById("rect_5");
   slideshow_token = 0;

   function activateNextSlide(){
     slideshow_token = (slideshow_token + 1) % 6;
     updateSlideshow(slideshow_token);
   }
   function updateSlideshow(token) {
     switch (token) {
       case 0: <!-- Apply presentation for image0 --> break;
       …
       case 5: <!-- Apply presentation for image5 --> break;
     }
   }
  </script>
  <g>
    <rect id="rect_0" visibility="hidden"/>
    <image id="img_0" visibility="hidden" xlink:href="pic0.png"/>
    …
    <rect id="rect_5" visibility="hidden"/>
    <image id="img_5" visibility="hidden" xlink:href="pic5.png"/>
  </g>
</svg>
```

Code 37: A Media description for an image gallery content in SVG.

Compared to the work of S. Boll [23], the *Scalable MSTI* model does not comply with the 'presentation-neutral' definition. In our approach, the multimedia document is expressed in the 'presentation-specific format used for playout of the multimedia materials'. As a consequence, our approach cannot be used to enable publication of content in multiple formats. However, the MSTI model enables the complete separation of the multimedia scene properties from the semantics of the multimedia document without requiring a publication phase, in which an internal document model needs to be converted into a standardized multimedia format. As opposed to meta-model or meta-format approaches described in Section 4.2.2.1 and in Section 4.2.2.2, the *Scalable MSTI* approach can take full advantage of the richness, including specific low-level features, of any of the multimedia standards such as SVG [118], MPEG-4 BIFS [59], SMIL [119], HTML [120], NCL [105] or Flash.

6.1.2 The update of scene properties

The *Media* description alone cannot be presented to the user. Some scene properties need to be added, to be applied to the media components. In the MSTI approach, this is done using scene description transformations (Section 4.3). These transformations are described in three additional descriptions: in the *Spatial*, *Temporal* and *Interactive* descriptions. As a consequence, the *Spatial*, *Temporal* and *Interactive* descriptions of the *MSTI* model are closely related to their *Media* description and complement it to build a complete multimedia document as described in Figure 49.

In existing multimedia scenes, specifically in web scenes expressed in HTML, styling is an important part of the description. The style of a scene is defined as a set of decorative properties assigned to each media component, such as color, border width or transparency values of graphics elements. We believe that these decorative properties do not directly relate to the spatial, temporal or interactive composition of media components. Hence, decorative properties constitute specific visual properties that should typically be defined as part of the spatial layout of a multimedia document (*Spatial*). However, they may also be applied by timed animations in which case they should be part of the *Temporal* description; or by user action, in which case they should be part of the *Interactive* description as further explained in Section 6.2.2.

The transformations that can be applied to a multimedia scene description such as the *Media* description of the *Scalable MSTI* model have been described in Chapter 4. We evaluate here how they can be used for our purpose. In practice, the spreading of scene attributes (Section 4.3.2) can be defined as part of the document structure using format-specific paradigms. The replacement of scene attributes (Section 4.3.1) and the update of scene elements (Section 4.3.3) are classical transformations for XML documents. We can cite four technologies that fit in these two categories and that need to be considered. Declarative Stylesheet Languages, such as XSL [122], are quite powerful for modifying the structure of a document and can indeed be used to apply presentation updates to a *Media* description. Cascading Style Sheets (CSS) [123] are also widely used to apply styles to documents and therefore to transform them. However, they cannot be used for our purpose since they do not allow the modification of the structure of an XML document and only allows for scene attribute replacements. In the same way, programmatic DOM-based specification, such as a JavaScript code, can also be used to describe advanced scene description transformations but this technology largely overpowers our needs. Finally, scene updates as defined by the MPEG-4 BIFS [59], MPEG-4 LASeR [62] and Flash standards can also be used for that purpose.

The syntax of the scene transformations of the *Scalable MSTI* model has been inspired by the MPEG-4 BIFS updates so that they can be expressed with a compact and language-agnostic representation that let implementation choices open. The syntax of these scene transformations allows the insertion (Section 6.1.2.1), the deletion (Section 6.1.2.2), the replacement (Section 6.1.2.3) or the move (Section 6.1.2.4) of a scene element by pointing to an existing identifier`(id)` into the *Media* description. The *Scalable MSTI* model currently relies on the identifier mechanism of existing multimedia standards to link presentation characteristics to scene attributes. For example, the SVG language defines an attribute called `id`. Another example is the `DEF` attribute in BIFS. However, generic technologies such as XML identifiers [128] or XPath expressions [129] could also be used for that purpose.

6.1.2.1 The Insert scene command

The `Insert` command of the *Scalable MSTI* model corresponds to the insertion of scene elements as described in Section 4.3.3.1. The scene description to be inserted is provided as a child of the `<insert>` element and refers to an existing element of the *Media* description using the `ref_id` attribute. The scene description is inserted as a top child of the referenced scene element.

```
<insert ref_id="root"> timedSlideshowUpdate(); </insert>
```

Code 38: An Insert command.

6.1.2.2 The Delete scene command

The `Delete` command of the *Scalable MSTI* model corresponds to the deletion of scene elements as described in Section 4.3.3.2. The scene description to be deleted is defined by referencing an existing element of the *Media* description using the `ref_id` attribute of the `<delete>` element.

```
<delete ref_id="root"/>
```

Code 39: A Delete command.

6.1.2.3 The Replace scene command

The `Replace` command of the *Scalable MSTI* model corresponds to the replacement of scene attributes or scene elements as described in Section 4.3.1 and in Section 4.3.3.3. The scene properties to be replaced are provided as a child of the `<replace>` element and refer to a specific attribute (`attribute`) of an existing element of the *Media* description using the `ref_id` attribute. In the same way, the scene description to be replaced is provided as a child of the `<replace>` element and only refers to an existing element of the *Media* description using the `ref_id` attribute (no `attribute` value is specified in that case).

```
<replace ref_id="root" attribute="viewBox">0 0 250 200</replace>
<replace ref_id="root"> </replace>
```

Code 40: A Replace command.

6.1.2.4 The Move scene command

The `Move` command of the *Scalable MSTI* model corresponds to the move manipulation of scene elements as described in Section 4.3.3.3. The scene description to be moved is defined by referencing an existing element of the *Media* description using the `src_id` attribute of the `<move>` element and point to a destination element using the `ref_id` attribute. The moved scene description is inserted as a top child of the referenced scene element.

```
<move src_id="live" ref_id="root"/>
```

Code 41: A Move command.

6.2 Splitting scene properties

Our preliminary work described in Chapter 5 showed several limitations when using a generic adaptation decision-taking engine (such as defined in MPEG-21) for multimedia presentations as far as the diversity of context parameters is concerned. On the one hand, a media-based approach (Section 5.2) is restricted to technical context parameters related to the transmission and the playback of elementary media (e.g. audio bitrate, video codec support or image resolution). On the other hand, a context-driven approach (Section 5.3) opens broad possibilities (e.g. animation requirements, presentation level of details or text and audio languages). However, this context-driven approach still requires content priorities or automated generation rules in order to define a deterministic adaptation decision-taking process. A simpler context-driven approach consists in preparing multiple presentation alternatives targeting specific devices. However, the description of presentation characteristics based on individual context parameters enables more flexible adaptation scenarios. In both cases, the underlying decision-taking process remains similar: a set of multiple presentation alternatives are prepared or can be generated in order to address several usage contexts.

The *Scalable MSTI* model suggests another way for the adaptation of multimedia presentations. Indeed, the *Scalable MSTI* model first concentrates on the intrinsic properties of multimedia presentations without considering any adaptation issues. More precisely, the MSTI model categorizes all scene properties into three components: *Spatial*, *Temporal* and *Interactive* components (STI components). Even though this decomposition is only the first step of multimedia scene scalability, it already enables simple but interesting adaptation scenarios (see Section 6.4.1) by suggesting authoring practices: presenting media components along with multiple timed and interactive accesses to them. Besides, accessibility guidelines [126] militate for such a diversity of media access. For instance, a media component might be accessible though interactivity but can also be displayed in the course of the multimedia document playback in order to take into account usage conditions where user interactions are not possible. Therefore, our *Scalable MSTI* model defines independent STI components, whenever possible, by gathering all presentation logic in the *Media* description while maintaining its coherence during transformations by using shared variables as illustrated in Section 6.4.1.

In the following, STI components are described in Section 6.2.1. As explained in Section 6.2.2, this decomposition of scene descriptions also includes styling properties. Finally, the results of our experiments on the SVG, MPEG-4 BIFS and SMIL standards are discussed in Section 6.2.3.

6.2.1 The STI components

In the following, the descriptions of the *Scalable MSTI* components are specified: the *Spatial* description (Section 6.2.1.1), the *Temporal* description (Section 6.2.1.2) and the *Interactive* description (Section 6.2.1.3). For each of these STI components, some examples of scene properties from the SVG, MPEG-4 BIFS and SMIL languages are provided.

6.2.1.1 The Spatial description

The *Spatial* description of the *Scalable MSTI* model defines the layout for all the media components that are part of the *Media* description. This *Spatial* description entirely relies on the spatial model of the targeted presentation format. It includes the positioning, visibility, size and style of media components as described in the visual models introduced in Section 3.2. An example of *Spatial* description is provided in Code 42 and can be applied to the *Media* description illustrated in Code 37 to create a one-image presentation. The syntax of existing standardized scene descriptions can be used to illustrate the scene properties that are expected in a *Spatial* description (see Table 5).

```
<spatial>
  <replace ref_id="root" attribute="viewBox">0 0 250 200</replace>
  <replace ref_id="img_0" attribute="x">45<replace>
  <replace ref_id="img_0" attribute="y">20</replace>
  <replace ref_id="img_0" attribute="width">160</replace>
  <replace ref_id="img_0" attribute="height">160</replace>
  <replace ref_id="img_0" attribute="visibility">visible</replace>
</spatial>
```

Code 42: A Spatial description of a one-image gallery in SVG.

Table 5: Spatial properties of the SVG, BIFS and SMIL formats.

SVG
Positioning attributes: transform, x, y, preserveAspectRatio, viewbox, rotate, text-anchor, translate **Sizing attributes:** width, height, scale, skewX, skewY, font-size **Visibility attributes:** display, visibility, initialVisibility <animateTransform> attributes: type, from, to <animation> attributes: x, y, width, height, preserveRatio,
BIFS
Positioning attributes: *translation* **Sizing attributes:** *rotationAngle, scale, scaleOrientation, radius* **Visibility attributes:** transparency, whichChoice The <PathLayout> element and The <IndexedLineSet2D> attributes:*coord, colorIndex, colorPerVertex, coordIndex* <ScalarInterpolator>, <CoordinateInterpolator2D>, <PositionInterpolator2D> attributes: *key* and *keyValue* <ScalarAnimator>, <PositionAnimator2D>: *fromTo, key, keyType, keySpline, keyValue, keyOrientation, weight, keyValueType, offset*
SMIL
Positioning attributes: top, bottom, left, right, width, height, fit, z-index, regPoint, regAlign, mediaAlign, soundAlign <area> attributes: shape, coords <animate> attributes: from, to, by, accumulate, additive, calcMode, values, path, keySplines <transition> attributes: type, subtype, fadeColor, horzRepeat, vertRepeat, borderWidth, borderColor and <param> element (name, value)

6.2.1.2 The Temporal description

The *Temporal* description of the *Scalable MSTI* model defines the timing of the multimedia presentation as a whole (e.g. the presence of media components over time as described in 3.3.1) but also the timing of each individual elementary media (e.g. animations as described in 3.3.3). The *Temporal* description only exploits the functionalities of the targeted presentation format. Such a format usually includes the presence of media components over time, their synchronization and the timing of the presentation as introduced in Section 3.3. An example of a *Temporal* description is provided in Code 43 and can be applied to the *Media* description illustrated in Code 37 to create a timed image slideshow.

```
<temporal>
  <insert ref_id="root">
    <script id="script_timedUpdade">
      function timedSlideshowUpdate() {
        activateNextSlide();
        setTimeout("timedSlideshowUpdate()", 5000);
      }
    </script>
  </insert>
  <insert ref_id="root"> timedSlideshowUpdate(); </insert>
</temporal>
```

Code 43: A Temporal description of a timed slideshow in SVG.

The update of multimedia documents over time to define multimedia service is not covered at this level since a *Temporal* description only provides the temporal properties of the presentation for a finite period of time. The evolution of *Scalable MSTI* document in live scenarios is further discussed in Section 6.5.3.1. The syntax of standardized scene descriptions can be used to illustrate the scene properties that are expected in a *Temporal* description (see Table 6).

Table 6: Temporal properties of the SVG, BIFS and SMIL languages.

SVG
The <animate>, <set>, <animateMotion>, <animateColor>, <animateTransform> elements The <discard> elements **Attributes to control the timing of the animation**: begin, dur, end, min, max, restart, repeatCount, repeatDur, fill **Time-related events**: beginEvent, endEvent, repeatEvent, SVGTimer
BIFS
The <TimeSensor> (cycleInterval, loop, startTime, stopTime) element The <TemporalTransform> element (startTime, optimalDuration, active, speed, scalability, stretchMode, shrinkMode, maxDelay) The <TemporalGroup> elements (costart, coend, meet) <MovieTexture> and <AnimationStream> attributes: loop, speed, startTime, stopTime, isActive <Layout> attributes: scrollRate, smoothScroll, loop, scrollVertical scrollMode
SMIL
The <animate>, <set>, <animateMotion>, <animateColor> elements The <seq>, <excl>, <par> elements The <prefetch> element **Attributes to control the timing of the animation**: begin, end, dur, repeatCount, repeatDur, keyTimes, accelerate, decelerate, speed, **media attributes**: mediaRepeat, clipBegin, clipEnd, min, max, restart, restartDefault **Attributes that define animation values over time:** dur, startProgress, endProgress, direction, transIn, transOut **Time-related events**: beginEvent, endEvent, repeatEvent, repeat

6.2.1.3 The Interactive description

The *Interactive* description of the *Scalable MSTI* model adds interactive aspects to the multimedia document and defines the behavior associated with interactions as introduced in Section 3.4. It may specify control over media components, navigation schemes or user actions. An example of an *Interactive* description is provided in Code 44 and can be applied to the *Media* description illustrated in Code 37 to create a shadow when moving the mouse over an image of the slideshow. The syntax of standardized scene descriptions can be used to illustrate the scene properties that are expected in an *Interactive* description (see Table 7).

```
<interactive>
  <replace ref_id="img_0" attribute="onmousemove">
    mouseOverImage0(1);
  </replace>
  <replace ref_id="img_0" attribute="onmouseout">
    mouseOverImage0(0);
  </replace>
  <insert ref_id="root">
    <script id="script_is_over_img_0">
      function mouseOverImage0 (isOver) {
        if ( isOver ) { rect0.setAttribute("visibility", "visible"); }
        else { rect0.setAttribute("visibility", "hidden"); }
      }
    </script>
  </insert>
</interactive>
```

Code 44: An Interactive description of an interactive slideshow in SVG.

Table 7: Interactive properties of the SVG, BIFS and SMIL languages.

SVG
The <a> element **Graphics and text content elements attributes:** focusable, nav-next, nav-prev, nav-up, nav-up-right, nav-right, nav-down-right, nav-down, nav-down-left, nav-left, nav-up-left, focusHighlight **Interactive events:** DOMFocusIn, DOMFocusOut, DOMActivate, click, mousedown, mouseup, mouseover, mousemove, mouseout, mousewheel, textInput, keydown, keyup, scroll The <handler> and <listener> elements <text> and <textarea> attribute: editable
BIFS
<Anchor>, <TouchSensor> and <PlaneSensor2D> elements <InputSensor> elements (KeySensor, StringSensor, Mouse type)
SMIL
<a> and <area> elements (show, target, accesskey, tabindex, fragment, alt) **Media attribute:** sensitivity **User-related events**: focusInEvent, focusOutEvent, activateEvent, inBoundsEvent, outOfBoundsEvent

6.2.2 Styling properties

Styling properties[45] as defined in Section 3.2.3 are decorative features that are applied to media components (e.g. images, text or polygons) and that determine the overall aspect of the presentation which is usually based on a graphical charter. The *Scalable MSTI* model is very lenient about styling properties because they do not have any consequences on spatial, temporal or interactive properties of the MSTI model. Two main reasons have driven our choice to not mandate a specific style description. First, some styling properties can be closely linked to the semantics of the presentation once applied. In that case, they cannot be easily replaced without reconsidering the entire presentation or applied to another document without making sure that both documents have similar semantics. For instance, two key

[45] This definition of styling properties does not cover layout properties that can be specified using Cascading Style Sheet (CSS).

elementary media (e.g. an image and its caption) might be semantically linked in a presentation by encompassing them in rectangle with a border that materializes their close relationship. Such a border style would not be applied to the rectangles of another *Media* description unless similar media relationships were identified. A second reason is that styling properties are often defined according to a particular layout or designed to create a visual effect that is dynamically triggered through interactive or timed events. For instance, the width of a rectangle's border differs depending on its size to match human visual system and the color of a button will be different if it is pressed or not. We suggest defining all styling properties of media components in the *Spatial* description of MSTI document except for those that are dynamically or interactively assigned and that can be defined respectively in the *Temporal* or *Interactive* descriptions. An example of static and dynamic styling properties assignment is given in Code 45 where the *Spatial* and *Temporal* description can be applied to the *Media* description provided in Code 37. A non-exhaustive list of styling properties is given in Table 8 and gives an overview of the XML elements/attributes that can be considered as styling properties when generating *Scalable MSTI* documents.

```
<spatial>
  <replace ref_id="rect_0" attribute="fill">blue</replace>
  <replace ref_id="rect_0" attribute="stroke">green</replace>
  <replace ref_id="rect_0" attribute="stroke-width">4</replace>
</spatial>

<temporal>
  <insert ref_id="text_0">
    <animate id="animate_text_0_color"
      attributeName="fill" from="blue "to="green"
      repeatCount="indefinite" begin="0s" dur="2s" fill="freeze"/>
  </insert>
</temporal>
```

Code 45: Styling properties in Spatial and Temporal descriptions.

Table 8: Styling properties of the SVG, BIFS and SMIL languages.

SVG
The <solidColor>, <linearGradient>, <radialGradient> elements
The <font> and <font-face> elements
fill, fill-rule, fill-opacity, stroke, stroke-width, stroke-linecap, stroke-linejoin, stroke-miterlimit, stroke-dasharray, stroke-dashoffset, stroke-opacity, viewport-fill and viewport-fill-opacity attributes
BIFS
The <Material2D> element(*emissiveColor, filled, lineProps, transparency*)
<Color>, <LineProperties>, <LinearGradient>, <RadialGradient> elements
The <FontStyle> elements
<Background2D> attribute: *backColor*.
SMIL
<region> attributes: *backgroundColor* and *showBackground*
<root-layout> and <topLayout> attributes: *backgroundColor*

6.2.3 Experiments and results

The decomposition of multimedia scene descriptions into the four components of the *Scalable MSTI* model is the first step towards multimedia scene scalability. This design choice for our scalable model was driven by the intrinsic properties of multimedia presentations. In fact, the user can access media

components on a static display (spatial properties), over time (temporal properties) or through interaction (interactive properties). We have also taken into account the need to leverage all the specific functionalities and optimization of targeted scene formats. Although the separation of the scene structure, metadata and logic including media components and their presentation using dedicated spatial, temporal and interactive model are usual practice, our experiments showed that the actual division of scene description formats into separated components is not always a straightforward task.

In the following, the results of our experiments are presented. First, the test sequences and the experimentation process are described in Section 6.2.3.1. Then, the main difficulties raised by the decomposition of scene descriptions in the MSTI model are discussed focusing on two aspects: the dependencies between presentation characteristics and the scene structure (Section 6.2.3.2) and the separation of spatial, temporal and interactive properties into distinct but related descriptions (Section 6.2.3.3).

6.2.3.1 Testing environment

The objective of the tests we conducted was to experiment the feasibility of the complete separation of STI components on a large number of multimedia presentations. In practice, our experiments have been conducted on two different types of multimedia content: complex sequences and unitary test sequences in the MPEG-4 BIFS (ExtendedCore2D profiles mainly), the SVG (1.2 tiny) and the SMIL (3.0) standards that we successfully transformed into *Scalable MSTI* documents.

Scalable MSTI documents corresponding to unitary tests have been automatically extracted from original documents of the SVG, BIFS and SMIL regression test suite using a dedicated Java program. This program uses the XML schema of the respective languages and the decomposition of STI properties described in Table 5, Table 6, Table 7 and Table 8. Advanced SVG and BIFS presentations have been prepared manually having in mind the MSTI design. Indeed, we did not optimize our extraction tool to deal with CSS descriptions and JavaScript code that are often used but difficult to analyze automatically. Besides, the implementation of such an MSTI extraction tool quickly becomes cumbersome due to the issues explained below in Section 6.2.3.2 and in Section 6.2.3.3. As a consequence, some workarounds and exceptions depending on scene formats are required to generate an MSTI document from an existing standardized document.

Once generated, the validation of each MSTI document was performed by applying the *Spatial*, *Temporal* and *Interactive* descriptions to their related *Media* description in order to create a playable multimedia document. This operation, which consists in generating a standard scene description from a MSTI document, is called the *STI* composition and is illustrated in Figure 50. The decomposition of the scene descriptions into the MSTI model is validated if the original presentation was successfully restored, and if the order of STI transformations does not interfere with the expected results. Our experiments showed that the recreation of the original document from the MSTI document was possible and lossless as long as the *STI* components were designed to be independent from each other.

For the purpose of our experiments, each STI description was automatically translated into XSL documents and applied to the *Media* description. As explained before, we did not write directly our STI descriptions using XSLT because we also want the composition to be applicable on end-user terminals, where an XSLT engine would not be available. For testing purpose, we used XSLT as a convenient way to implement the composition.

6.2.3.2 Separating STI properties

The *Scalable MSTI* model relies on a complete separation of multimedia presentations into three different components: *Spatial*, *Temporal* and *Interactive (STI)* descriptions. However, these components might be closely related in some cases. For instance, the animation of an elementary media requires a timer (*Temporal*) that will drive the movements of the media along a path (*Spatial*). Similarly, an interactive anchor applied to the surface of an elementary media (*Interactive*) requires that this media is correctly positioned in the presentation (*Spatial*). The links between STI components can be handled

through the definition of specific elements that combine them as is the case of the `S-Relation` element in the Madeus model [79] for 'spatial animations'. However, this approach is not extensible to all STI dependencies and standardized presentation formats sometimes rely on low-level generic tools to cope with such authoring cases (e.g. absence of a specific animation node in MPEG-4 XMT). In our model, we do not define such elements.

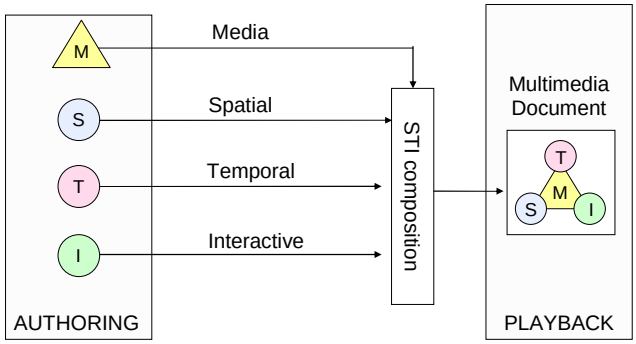
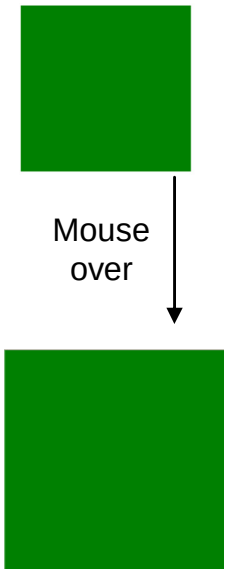


Figure 50: The STI composition and document generation.

In the MSTI model, all dependent scene properties are separated into STI components by referring to common elementary media or logical structures of the *Media* description. Hence, STI components do not depend upon each other but are all linked to the same *Media* description. For instance, an interactive button may animate slightly by changing its size when the mouse is over it as illustrated in Code 46. In that case, the shape of the 'button' is part of the *Media* description; the position, size and extended size of this shape is specified in the *Spatial* description; the duration and a method to trigger the animation of this shape is defined in the *Temporal* description and finally the interactive behavior triggered by the mouse is associated with the 'button' in the *Interactive* description as illustrated in Code 47. Furthermore, complex STI dependencies can also be handled through the definition of shared presentation elements such as variables, presentation states or methods. STI descriptions can be closely related through references to the same elements in the *Media* description but the STI descriptions never directly depend on each other.




```
<svg xmlns="http://www.w3.org/2000/svg"
  id="svg-root" viewBox="0 0 40 40">
 <rect id="rect_0" fill="green" x="5" y="5" width="30" height="30">
 <animate id="animate_rect_0_width_in" attributeName="width"
          from="30" to="40" dur="0.4s" fill="freeze"
          begin="rect_0.mouseover"/>
 <animate id="animate_rect_0_height_in" attributeName="height"
          from="30" to="40" dur="0.4s" fill="freeze"
          begin="rect_0.mouseover"/>
 <animate id="animate_rect_0_x_in" attributeName="x"
          from="5" to="0" dur="0.4s" fill="freeze"
          begin="rect_0.mouseover"/>
 <animate id="animate_rect_0_y_in" attributeName="y"
          from="5" to="0" dur="0.4s" fill="freeze"
          begin="rect_0.mouseover"/>
 <animate id="animate_rect_0_width_out" attributeName="width"
          from="40" to="30" dur="0.4s" fill="freeze"
          begin="rect_0.mouseout"/>
 <animate id="animate_rect_0_height_out" attributeName="height"
          from="40" to="30" dur="0.4s" fill="freeze"
          begin="rect_0.mouseout"/>
 <animate id="animate_rect_0_x_out" attributeName="x"
          from="0" to="5" dur="0.4s" fill="freeze"
          begin="rect_0.mouseout" />
 <animate id="animate_rect_0_y_out" attributeName="y"
          from="0" to="5" dur="0.4s" fill="freeze"
          begin="rect_0.mouseout"/>
 </rect>
</svg
```

Code 46: A rectangle being resized when hovered in SVG.

```
<svg xmlns="http://www.w3.org/2000/svg" id="svg-root">
  <rect id="rect_0">
    <animate id="animate_rect_0_width_in"/>
    <animate id="animate_rect_0_height_in"/>
    <animate id="animate_rect_0_x_in"/>
    <animate id="animate_rect_0_y_in"/>
    <animate id="animate_rect_0_width_out"/>
    <animate id="animate_rect_0_height_out"/>
    <animate id="animate_rect_0_x_out"/>
    <animate id="animate_rect_0_y_out"/>
   </rect>
</svg>

<spatial>
  <!-- Document decoration -->
  <replace ref_id="rect_0" attribute="fill">green</replace>

  <!-- Document composition -->
  <replace ref_id="svg-root" attribute="viewBox">0 0 40 40</replace>

  <replace ref_id="rect_0" attribute="x">5</replace>
  <replace ref_id="rect_0" attribute="y">5</replace>
  <replace ref_id="rect_0" attribute="width">30</replace>
  <replace ref_id="rect_0" attribute="height">30</replace>

  <replace ref_id="animate_rect_0_width_in" attribute="attributeName">width</replace>
  <replace ref_id="animate_rect_0_width_in" attribute="from">30</replace>
  <replace ref_id="animate_rect_0_width_in" attribute="to">40</replace>
  ...
</spatial>

<temporal>
  <replace ref_id="animate_rect_0_width_in" attribute="dur">0.4s</replace>
  <replace ref_id="animate_rect_0_width_in" attribute="begin">indefinite</replace>
  <replace ref_id="animate_rect_0_width_in" attribute="fill">freeze</replace>
  ...
</temporal>

<interactive>
  <replace ref_id="animate_rect_0_width_in" attribute="begin">rect_0.mouseover</replace>
  <replace ref_id="animate_rect_0_height_in" attribute="begin">rect_0.mouseover</replace>
  <replace ref_id="animate_rect_0_x_in" attribute="begin">rect_0.mouseover</replace>
  <replace ref_id="animate_rect_0_y_in" attribute="begin">rect_0.mouseover</replace>
  ...
</interactive>
```

Code 47: MSTI description of a resizing SVG rectangle when hovered.

6.2.3.3 Separating Media and STI properties

The *Scalable MSTI* model relies on the separation of the scene structure and multimedia components from the presentation characteristics. However, this strict separation might suffer some exceptions depending on application scenarios. For instance, multimedia documents in some formats such as SMIL may be structured on a temporal basis (*seq/par* elements). In that case, these temporal structures have to be defined in the *Media* description and only timing parameters (*dur*) are defined in the *Temporal* description as illustrated in Code 47 with the <animate> element. Such limitations are overcome by our *Scalable MSTI* by defining explicit dependencies as explained in 6.3.3.3.

Furthermore, some multimedia formats offer generic syntax in order to improve language learning time or to enable efficiency coding. As a consequence, such generic elements can be used in all three STI components. We chose to allow them in any *Spatial*, *Temporal* or *Interactive* description and to place them in the description they related to on a content basis. Typical examples are the multipurpose <*script*> elements of SVG which modify STI descriptions, the <*Route*> element of BIFS which can link STI events and the *begin* attribute of the SMIL animation elements which can contain time or interactive values. Any element that does not fit into one single STI component due to its usage context is then placed in the *Media* description.

In the same way, some multimedia formats define optimization properties that are essential to improve the usability of the content in some specific usage environments: the `image-rendering` attribute in SVG allows the indication of author preferences between quality and speed of rendering; or the `<QuantizationParameter>` element in BIFS enables authoring the trade-off between quality and bandwidth. These properties are not directly related to *STI* components (and their support is often optional on receiver's side) but may impact on applicable usage conditions for *STI* descriptions. In practice, these scene properties can be defined in the *Media* description or classified according to the adaptation parameters applied to STI descriptions as defined in 4.1.

In the MSTI model, the separation of presentation characteristics from the document structure can remain flexible. Indeed, the main objective of the *Media* description is not to provide a presentation-agnostic description of the presentation but to propose a common basis for the *Spatial*, *Temporal* or *Interactive* transformations. So, the inclusion of presentation characteristics into the *Media* description does not introduce any compliance issue with our *Scalable MSTI* model as long as the document features *Spatial*, *Temporal* and *Interactive* descriptions offering scene properties in their respective domain. For instance, all style properties can be kept in a CSS file attached to the *Media* description if these properties are not supposed to be modified during an adaptation process. As further explained in Section 6.3.1, this flexibility of the *Media* component constitutes the basis of the scalability properties of our MSTI model.

6.3 Multimedia scene scalability

Several scalable media codecs have been standardized in recent years to cope with heterogeneous usage conditions and aim at always providing audio, video and image content in the best possible quality. The term 'scalable' and 'scalability' should be understood as in the SVC standard [104] and 'refers to the removal of [video] bit-stream in order to adapt it to the various needs or preferences of end users as well as to varying terminal capabilities or network conditions'. In this section, we propose to transpose the concept of scalability to the world of multimedia documents by defining the scalable properties of a scene, within the so-called *Scalable MSTI* model.

Although multimedia scenes do not have the same properties as video sequences or images, scene properties can be decomposed into three main axes: *Spatial*, *Temporal* and *Interactive* dimensions as described in Section 6.2. Additionally, if we consider a set of versions of a presentation adapted for different contexts, the *Scalable MSTI* model further organizes the scene properties of each *Spatial*, *Temporal* and *Interactive* description of all versions into progressive descriptions. Each such progressive description is a scalability layer and is made of adaptation updates (see Section 5.3.1.2). This organization follows custom adaptation parameters such as the display dimensions, the user's level of interest, the presentation duration or the receiver's capability to cope with CPU-demanding animations. As a consequence, the adaptation accuracy to the user's context is determined during content generation by defining the granularity of each adaptation axis.

In the following, the core transformation process of our multimedia scene scalability based on adaptation updates is introduced in Section 6.3.1. Then, the general presentation principles that contribute the organization of *Spatial*, *Temporal* and *Interactive* descriptions into scalable layers are introduced in Section 6.3.2. Finally, the adaptation graph that structures the adaptation capabilities of multimedia presentations and drives our adaptation decision-taking engine in defined in Section 6.3.3.

6.3.1 Cascading STI compositions

The division of STI components into scalable layers requires an incremental transformation of the *Media* description of the MSTI model to progressively compose the adapted multimedia document. Therefore, the STI composition defined in Section 6.2.3 is extended to allow cascading STI compositions in the *Scalable MSTI* model. Indeed, the STI composition outputs a multimedia document that can be used as *Media* input for a new STI composition as long as new STI descriptions update previous ones. This flexibility of the STI composition comes from the fact that a *Media* description of the MSTI model can be considered a MSTI document since they are both described in the same format and contain the same identifiers referenced by STI components. The extended STI composition of the *Scalable MSTI* is

depicted in Figure 51 and shows how STI compositions can be cascaded to generate an adapted multimedia document with two STI layers.

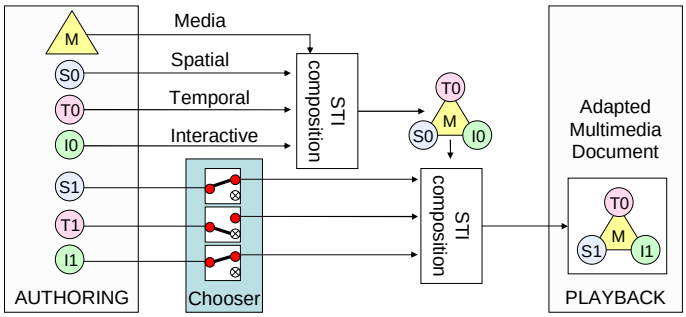


Figure 51: Cascading STI compositions.

In our scalable approach, the cascading property of the STI composition process is used to progressively provide spatial, temporal and/or interactive improvements to a base multimedia document. Of course, one main objective of multimedia scalability is to only provide the descriptions that are required from one layer to the next. As a consequence, a *Spatial* layer does not intend to completely update the previous *Spatial* layer but rather provides an update of the scene properties that are different from the previous layer. This approach is efficient compared to the generation of separate presentation alternatives because differential updates can achieve compact document representations as shown in our experiments detailed in Section 5.3.3.

In the *Scalable MSTI* model, a multimedia document is the result of successive STI compositions for each layer on each scalability axis. In practice, the multimedia document does not necessarily have to be reprocessed layer by layer if a progressive document rendering is not required. Indeed, STI descriptions can be aggregated prior to the composition process. During our experiments, this optimization has been performed by generating Extensible Style Sheets (XSL) that overwrite <`Replace`> MSTI commands when applying several layers on the same axis by tracking `ref_id` identifiers.

Finally, a scalable multimedia document relies on a base layer (S_0,T_0,I_0) that determines the minimal presentation of the multimedia document. This base layer contains spatial, temporal and interactive properties that define a simple presentation for the media components of the multimedia document. It may also contain some STI properties required for some enhancement layers as explained in Section 6.2.3.2 and decorative properties introduced in Section 6.2.

6.3.2 Scalable MSTI layers

In the context of scalability, the *Spatial*, *Temporal* and *Interactive* components of the MSTI model are called STI axes. The *Scalable MSTI* model is based on order relations that organize STI axes in a progressive manner. For this reason, each STI axis is associated with an adaptation parameter that drive the nature of the spatial, temporal and interactive properties contained in STI progressive layers. The scalability along an axis can be coarse-grained or fine-grained according to the length of adaptation steps. Each layer of a scalable multimedia document is associated with an adaptation parameter value related to its applicable usage environments. Since three scalability axes are defined in the *Scalable MSTI* model, at most three adaptation axes have to be defined during document generation and mapped onto STI scalability axes.

In the same way, a scalable image encoder can freely organize its quality layers according to some region of interests, the *Scalable MSTI* model allows the organization of the STI layers according to any adaptation parameter that is compatible with the progressive nature of spatial, temporal or interactive scene properties. Of course, the flexibility offered by scene descriptions is much higher than in the case of natural media such as images. For this reason, we do not mandate any adaptation parameter for the three scalability axes of the MSTI model. They can be freely defined during authoring according to targeted application scenarios.

The *Scalable MSTI* document syntax is defined for the *Spatial*, *Temporal* and *Interactive* descriptions. STI descriptions are divided into `<layer>` elements to which a number is associated (`value` attribute) and which may provide informative parameters (`<adaptationParameter>` element) to guide the adaptation decision-making process. A `<layer>` element is composed of `<insert>`, `<delete>`, `<replace>` and `<move>` elements that can be used to modify the *Media* description by inserting, replacing, moving or deleting presentation fragments as introduced in Section 6.1.2

In the following, typical examples for the organization of STI axes are provided for the spatial (Section 6.3.2.1), temporal (Section 6.3.2.2) and interactive (Section 6.3.2.3) scene properties.

6.3.2.1 Example of Spatial layers

A typical adaptation parameter which can be mapped onto the *Spatial* axis of the *Scalable MSTI* model is the targeted screen resolution. Since this parameter is bi-dimensional, we can restrict targeted spatial dimensions to incremental values as for the generalized spatial scalability of SVC [104]. As a consequence, 'neither the horizontal nor the vertical resolution can decrease from one layer to the next'. Doing so, a layered description of the spatial properties of multimedia documents can address several screen resolutions. For example, from one layer to another (S_0,S_1,S_2), the *Spatial* description can move objects to better fit the updated screen size, or increase the size of some objects, possibly in a non-linear way as illustrated in Figure 52. Additionally, an example of scalable *Spatial* description is provided in Code 48.

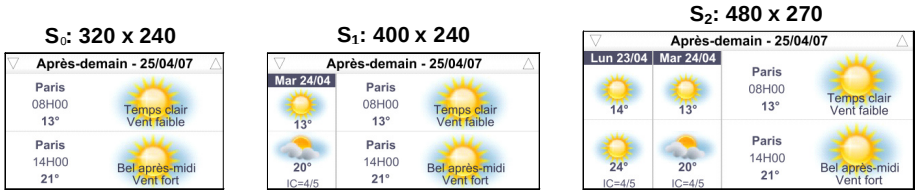


Figure 52: A spatially scalable scene driven by the screen resolution.

```
<spatial>
  <layer value="0">
    <adaptationParameter type="primary" value="320" // x display resolution
      href="urn:mpeg:mpeg21:2003:01-AdaptationQoSCS-NS:6.5.9.1"/>
    <adaptationParameter type="primary" value="240" // y display resolution
      href="urn:mpeg:mpeg21:2003:01-AdaptationQoSCS-NS:6.5.9.2"/>
    <replace ref_id="root" attribute="viewBox">0 0 320 240</replace>
    <replace ref_id="img_0" attribute="x">45</replace>
    <replace ref_id="img_0" attribute="y">20</replace>
    <replace ref_id="img_0" attribute="width">160</replace>
    <replace ref_id="img_0" attribute="height">160</replace>
    <replace ref_id="img_0" attribute="visibility">visible</replace>
  </layer>
  <layer value="1">
    <adaptationParameter type="primary" value="400"
      href=" urn:mpeg:mpeg21:2003:01-AdaptationQoSCS-NS:6.5.9.1"/>
    <adaptationParameter type="primary" value="240"
      href=" urn:mpeg:mpeg21:2003:01-AdaptationQoSCS-NS:6.5.9.2"/>
    <replace ref_id="root" attribute="viewBox">0 0 400 240</replace>
    <replace ref_id="img_0" attribute="x">50</replace>
    <replace ref_id="img_1" attribute="x">225</replace>
    <replace ref_id="img_1" attribute="y">20</replace>
    <replace ref_id="img_1" attribute="width">160</replace>
    <replace ref_id="img_1" attribute="height">160</replace>
    <replace ref_id="img_1" attribute="visibility">visible</replace>
  </layer>
</spatial>
```

Code 48: A scalable Spatial description.

A layered description of the spatial properties of multimedia documents can address many screen resolutions through the use of fine-grained scalable descriptions but possibly implies overwriting spatial parameters from one layer to the next. Performance evaluations that have been carried out in Section 5.3.3 show that spatial adaptation through presentation updates is very efficient on constrained multimedia terminals but grows as the number of intermediate adaptation updates increase. In that case, presentation updates can be combined with one-way constraints (Section 4.2.4.2) or linear interpolations (Section 4.2.4.1) to achieve fine grain scalability with a better efficiency.

6.3.2.2 Example of Temporal layers

The transformation of the *Temporal* component of a MSTI document into a scalability axis can be performed by providing progressive timed properties. Examples of adaptation parameters that can be mapped onto the *Temporal* axis are the available processing power or battery state. In that case, *Temporal* layers of the scalable multimedia document are ordered in terms of processing requirements. For instance, the timing of an animation can be composed of different *Temporal* layers that define incremental levels of smoothness: from a two-state approach (T_0) to the continuous rendering of complex animations (T_2) as illustrated in Figure 53. Additionally, an example of scalable *Temporal* description is provided in Code 49.

The layers of the *Temporal* axis should be as progressive as possible from an authoring point of view. However, there might be good reasons to completely change the *Temporal* description from one layer to the next. Such cases might be triggered by the authoring need to deeply modify the temporal properties of a layer towards an enhanced timed multimedia presentation requiring advanced capabilities for playback. Thus, this kind of layer implies overheads in terms of bandwidth and may even require removing some description parts of the previous layers (see Section 6.3.3.1).

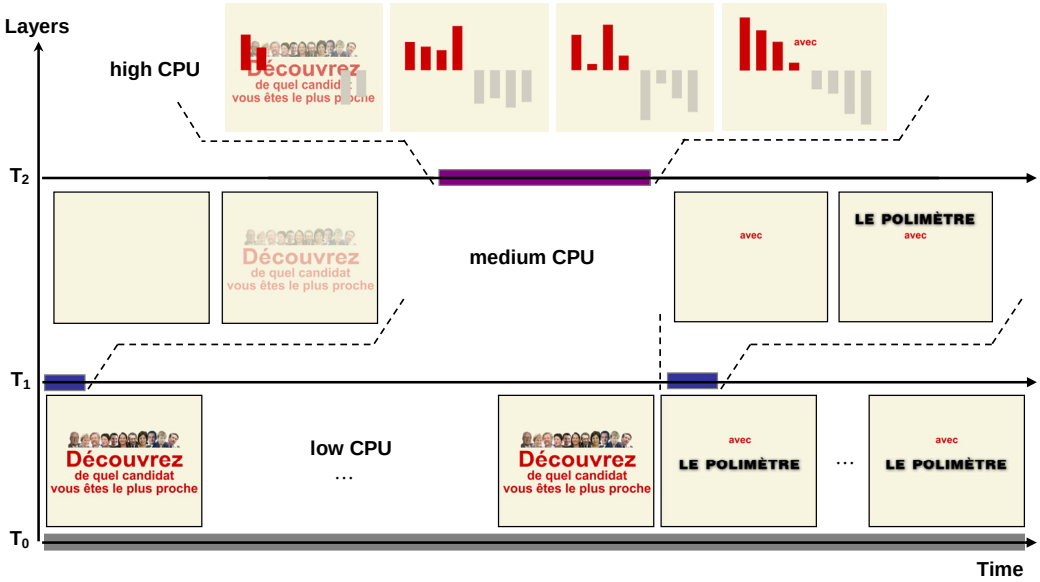


Figure 53: A temporally scalable scene driven by processing capabilities.

6.3.2.3 Example of Interactive layers

The *Interactive* component of the *MSTI* model can be divided into layers by progressively providing interactivity functions to the user. An adaptation parameter that can be mapped onto the *Interactive* axis is the number of elementary media required to be loaded when accessing auxiliary media through interactions. For instance, accessible documents through user interactions can be prioritize according to author's wishes and defined in incremental *Interactive* layers *(I_0,I_1,I_2)* so that memory requirements are progressively evaluated during playback as illustrated in Figure 54. Additionally, an example of scalable *Interactive* description is provided in Code 50.

```
<temporal>
  <layer value="1">
    <adaptationParameter type="primary" value="400.0" // low-CPU
     href="urn:mpeg:mpeg21:2003:01-DIA-CPUBenchmarkCS-NS:CFP2000"/>
    <replace ref_id="temporal_init">
      function temporal_init() {
        T_navigation_enabled = 1;
        no_running_anim = 0;
        timedSlideshowUpdate();
        no_running_anim = 1; }
    </replace>
    <replace ref_id="timedSlideshowUpdate">
      function timedSlideshowUpdate() {
        if (slideshow_temp_lock == 0) && (slideshow_perm_lock == 0)
            && (no_running_anim == 1)
        {
          activateNextSlide();
          navigation_direction = 0;
        }
       setTimeout("timedSlideshowUpdate()", 5000); }
    </replace>
  </layer>
  <layer value="2">
  <adaptationParameter type="primary" value="650.0" // medium-CPU
     href="urn:mpeg:mpeg21:2003:01-DIA-CPUBenchmarkCS-NS:CFP2000"/>
    <replace ref_id="temporal_init"> // Add animations
      function temporal_init() {
        T_navigation_enabled = 1;
        no_running_anim = 0;
        timedSlideshowUpdate();
        image0_anim = doc.getElementById("image_0_anim");
        image0_anim.addEventListener("endEvent", endOfAnim, false); }
      function endOfAnim() {
        no_running_anim = 1; }
    </replace>
    <insert ref_id="image_0">
      <animate id="image_0_anim" attributeName="x" begin="indefinite"
        dur="1s" fill="freeze"/>
    </insert>
    ...
  </layer>
<temporal>
```

Code 49: A scalable Temporal description.

The layers of the *Interactive* axis should be as progressive as possible from an authoring point of view. However, it might be difficult, in some cases, to order various types of media components on a single axis. Therefore, some interactive layers may be considered optional if they do not impact the enhanced *Interactive* layers of the same axis (see Section 6.3.3.2).

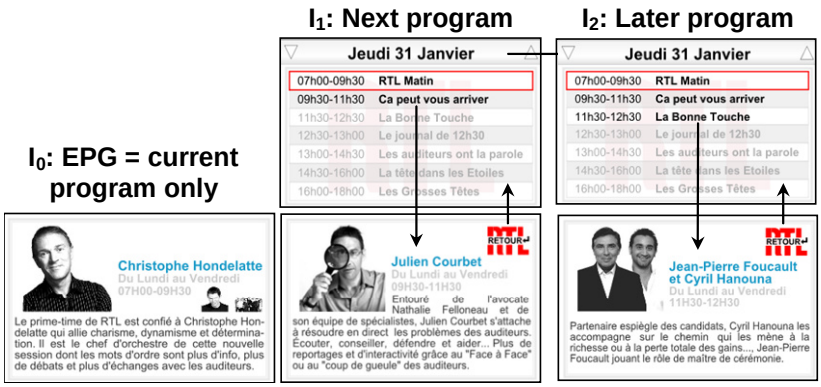


Figure 54: An interactively scalable scene driven by memory capacities.

```
<interactive>
  <layer value="1">
    <adaptationParameter type="primary" value="pen"
       href="urn:mpeg:mpeg21:2003:01-AdaptationQoSCS-NS:6.5.14"/>
    <replace ref_id="interactive_init">
       function interactive_init() {
        I_navigation_enabled = 1;
        previousButton.addEventListener("click", previous, false);
        nextButton.addEventListener("click", next, false); }
    </replace>
    <insert ref_id="root"> // Add previous-next navigation
      <script id="script_previous" type="text/ecmascript">
        function previous(evt) {
          updateTimeFreq();
          slideshow_temp_lock = 1;
          navigation_direction = 1;
          activatePreviousSlide();
          // This timeout only relate to the user interface
          // and not to the Temporal description of the document
          setTimeout("lock_slideshow()", 2000); }
      </script>
      <script id="script_next" type="text/ecmascript">
        function next(evt) {
          updateTimeFreq();
          slideshow_temp_lock = 1;
          navigation_direction = 0;
          activateNextSlide();
          setTimeout("lock_slideshow()", 2000); }
      </script>
      <script type="text/ecmascript">
        function lock_slideshow() {
          slideshow_temp_lock = 0; }
      </script>
    </insert>
  </layer>
  <layer value="2" skippable="true">
    <adaptationParameter type="primary" value="mouse" // User interaction input
     href=urn:mpeg:mpeg21:2003:01-AdaptationQoSCS-NS:6.5.14.4"/>
    <replace ref_id="interactiveUpdate0"> // Add mouse-over events
      function interactiveUpdate0() {
        image0_anim = doc.getElementById("image_0_anim");
        ...
        if (nb_visible_image > 0){
          image0.setAttribute("onmousemove", "mouseOverImage0(1)");
          image0.setAttribute("onmouseout", "mouseOverImage0(0)");
        }
      }
    </replace>
    <insert ref_id="root"> // locks slideshow if mouse over an image
      <script id="mouseOverImage" type="text/ecmascript">
        function mouseOverImage0(isOver) {
          if (isOver == 1) {
            if (no_running_anim == 1) {
              activateRect0();
              slideshow_perm_lock = 1; } }
          else {
          desactivateRect0();
          slideshow_perm_lock = 0; }
        } ...
      </script>
    </insert>
  </layer>
</interactive>
```

Code 50: A scalable Interactive description.

6.3.3 Adaptation graphs

Scalable multimedia authoring consists in defining and filling the successive layers of *Spatial*, *Temporal* and *Interactive* axes of a *Scalable MSTI* documents. All these scalable layers can be represented as an adaptation graph. These incremental layers represent improvements in the multimedia presentation but also often assume increasing capabilities in usage environments. Therefore, a specific type of adaptation parameter has to be associated with each STI axis. Typical device-oriented examples of adaptation parameters for STI axes are: resolution or memory consumption (*Spatial*), processing power (*Temporal*), input capabilities or memory consumption (*Interactive*). However, these mappings are not mandated and may be very different from one use case to another. For instance, user-oriented adaptation parameters could be: accessibility to partially-sighted people *(Spatial),* presentation duration *(Temporal)* and level of expertise *(Interactive).* Once an adaptation parameter has been assigned to each STI axis, a total order relation needs to be chosen. Given an order relation, chosen adaptation parameter values are ordered into a progressive manner and mapped onto individual scalability layers. This process determines the expected number of scalability layers for each STI axis. It should be noted that adaptation parameter values do not have to be accurate (e.g. processing power values). In fact, only the ordering of values is essential since precise adaptation parameter values may only be known at playback time (e.g. memory usage depends on the implementation).

When considering adaptation parameters that are mapped onto the three STI axes of a scalable multimedia document, it might not be satisfactory to discard some parameters just because there are too many of them. For instance, the three selected adaptation axes might be: resolution (*Spatial*), processing power (*Temporal*) and bandwidth (*Interactive*) but memory requirements and the availability (or not) of a sensitive screen might also be critical for some devices. The *Scalable MSTI* model enables adaptation scenarios to be driven by more than three adaptation parameters. Indeed, several types of adaptation parameters can be assigned to one STI layer. In that case, the additional parameters are called *auxiliary* adaptation parameters and are signaled in the syntax of scalable STI description using the type attribute of the `<adaptationParameter>` element (`auxilary` type instead of `primary` type). As illustrated in Figure 55, such auxiliary adaptation parameter can either be present on a single layer (e.g. sensitive screen applied to the layer I_1 of the *Interactive* axis) or distributed on several layers (e.g. memory requirements that depends on *Spatial* and *Interactive* scalability axes, e.g. I_1 and S_2). However, it should be noted that these auxiliary adaptation parameters are only adaptation hints and that the *Scalable MSTI* model may not provide the optimal presentation when considering them. In fact, the *Scalable MSTI* model guarantees that the best presentation (according to the author) is selected only when the adaptation process is driven by the three scalable adaptation parameters. For instance, the auxiliary memory value (RAM) provided in Figure 55 indicates a measure corresponding to (S_2,T_0,I_1). As a consequence, (S_3,T_0,I_0) might be the optimal presentation for a device with a 480x320 resolution, an 8 kbps bandwidth and 8 Mbytes of available RAM but an adaptation engine, which cares about memory, could stop at (S_2,T_0,I_0) since S_2 is labeled with a 8 Mbytes RAM auxiliary adaptation parameter. Therefore, auxiliary adaptation parameters can be used to guide the adaptation of scalable multimedia documents with much more than the three parameters defined as the adaptation axes of the scalable multimedia document but they also introduce limitations in some usage environments.

In addition to the `value` attribute which is mandatory for the `<layer>` element for the scalable MSTI syntax, eleven optional attributes have been defined in the *Scalable MSTI* model to specify adaptation paths in three-axis adaptation graphs: `randomAccessLayer` (Section 6.3.3.1), `skippable` (Section 6.3.3.2), `dependsOn{S|T|I}LayerValue` (Section 6.3.3.3), `requiredFor{S|T|I}LayerValue` (Section 6.3.3.4) and `blocks{S|T|I}LayerValue`.

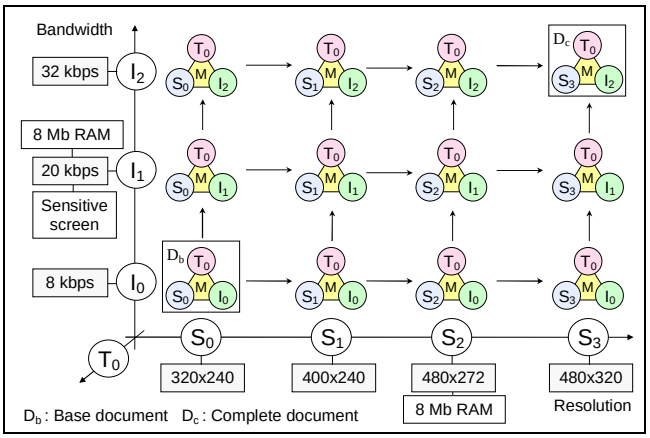


Figure 55: Example of scalable adaptation axes.

6.3.3.1 Key adaptation points

The adaptation graph of a scalable multimedia document defines layers for each *Spatial*, *Temporal* and *Interactive* (STI) axis. In fact, it defines a set of incremental presentation versions that can be selected according to usage environments. This set of presentation versions does not aim to be optimal for all usage configurations but the ordering of the three main adaptation parameters and the mapping between progressive parameters values and document STI layers guarantees that a suitable presentation can be found in any case, given that the *Base* scalable document addresses the lowest possible capabilities. However, content creators are usually asked to focus on some specific usage configurations that must be optimally addressed (e.g. the iPhone device). Such configuration must then be addressed by a point in the adaptation graph. However, because the author might want to ease or favor such targeted players when they perform their navigation in the adaptation path, the *Scalable MSTI* model lets content creators indicate such key points by using Random Access Layers.

From a playback point of view, the Random Access Layer indication helps in two situations. First, if the path to this key point is long in the adaptation graph, a fast and direct access to the presentation is enabled. Such shortcuts in the adaptation graph significantly reduce processing overheads that could happen when accessing to an advanced multimedia presentation (the highest STI layers) of a fine-grained scalable multimedia documents. Second, if the receiver follows a blind strategy to navigate in the adaptation graph, by processing the incremental STI layers of the scalable document one after another without matching its capabilities with their adaptation parameters, key points enable fallbacks that avoid the reloading of the whole document when a maximum layer has been identified. Indeed, although the MPEG-21 adaptation framework can be used to describe the mapping of the usage environment onto the adaptation parameters for scalable content (see Section 5.2.1), the data needed to describe this mapping when fine-granular multimedia documents are created might be too large in some application scenarios such as broadcast services.

A Random Access Layer (RAL) flag is assigned to a STI layer by defining a `randomAccessLayer` attribute. This attribute can be used for fast access to an advanced presentation and also be used to signal a valid fallback solution (without reloading all the content). Therefore, RAL provides a backward and forward refresh of STI layers on a single axis. For instance, a S_k RAL will reset all the modifications incurred by $[S_{k+1}, S_n]$ *Spatial* layers (forward refresh), it will collapse all *Spatial* descriptions from the S_0 to the S_{k-1} layer (backward refresh) and it will include the *Spatial* description specifically corresponding to layer k. In that case, the S_k *Spatial* layer can be activated even though none of the $[S_0, S_{k-1}]$ *Spatial* layers are activated. Therefore, despite the fact that the introduction of Random Access Layers in the adaptation graph reduces coding performances because of information redundancy, it can provide other entries than (S_0,T_0,I_0) in the adaptation graph and backward adaptation paths as illustrated in Figure 56.

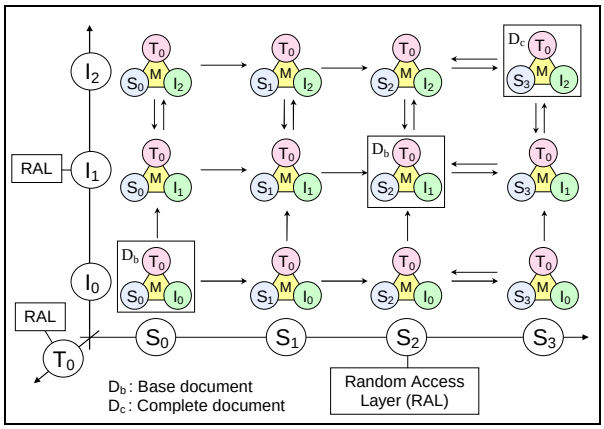


Figure 56: Adaptation paths with Random Access Layers.

6.3.3.2 Digressing adaptation paths

When preparing the adaptation graph of a scalable multimedia document, three main adaptation parameters are selected with their order relations, a set of progressive parameters values are defined and mapped onto the scalability layers of the *Scalable MSTI* model. However, it may happen that one specific parameter value does no fit with the overall ordering of its adaptation axis. For instance, a scalable multimedia document may target several resolutions ordered first by their horizontal dimensions: (S_0)240x240, (S_1)320x240, (S_2)400x240, (S_3)480x272 and (S_4)480x320. However, the content creator may be asked to provide a vertical layout for one of these targeted resolutions (e.g. 400x240) because some receivers have the ability to dynamically switch from an horizontal layout to a vertical layout (e.g. Samsung's Player Addict). Such vertical resolution (e.g. 240x400) does not fit with the ordering of the resolution-based adaptation axis but the presentation associated with this specific configuration may roughly look like some other presentations available in the adaptation graph (S_2 in our example). In that case, this adaptation parameter value can be defined as a digression in the adaptation paths.

In the *Scalable MSTI* model such digressing adaptation parameter values are signaled by `skippable` layers as illustrated in Figure 57. In fact, these layers related to a digressing adaptation parameter value depend on their preceding layers, may not be applied. In that case, the S_{k+1} Spatial layer can be activated although an the $[S_k\text{-}S_{k-i}]$ `skippable` layers are not activated. Digressing adaptation parameter values provide additional flexibility as far as adaptation is concerned since they increase the number of possible paths in the adaptation graph. However, `skippable` layers cause some redundancy (in S_3 in the example) and should therefore be used carefully. In our example illustrated in Figure 57, this means that layer S_3 should not assume that values of S_1 are unchanged. It should rewrite all values that can be modified in S_2.

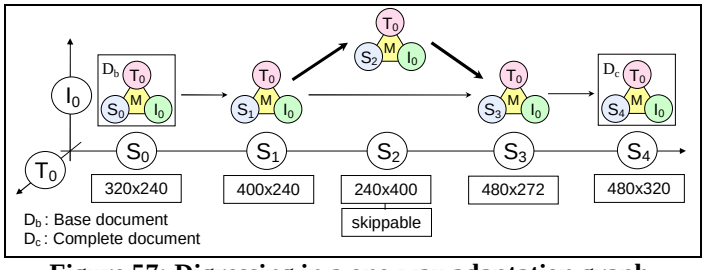


Figure 57: Digressing in a one-way adaptation graph.

6.3.3.3 Constrained adaptation paths

Spatial, temporal and interactive presentation characteristics are separated into STI components that do not refer to each other but share common structures and variables in the *Media* description. However, STI layers are not independent since the spatial, temporal and interactive properties of a multimedia

presentation can be closely coupled to build progressive adaptation scenarios. Hence, for a typical scene description describing an animation triggered by a user action, one option is to describe all the dependent properties of the multimedia presentation in the *base* layer of the STI scalability axis. This approach is advantageous since all adaptation paths are possible and will lead to the complete presentation. However, it may not be efficient since the *base* layers of STI scalability axes may include many unused descriptions. For instance, the position of all buttons defined in the *Interactive* description can be given in the S_0 *Spatial* base layer whereas the actual use of one of these buttons may be defined in the I_2 *Interactive* description. This is not optimal since a multimedia document in configuration (S_0,T_0,I_0) will perform the positioning of buttons that may never be visible to the user.

In order to cope with such dependencies, the *Scalable MSTI* model defines layer dependency using symmetric attributes: `dependsOn{S|T|I}LayerValue` and `requiredFor{S|T|I}LayerValue`. When used, the *Scalable MSTI* model does not grant the user with a complete freedom in the adaptation graph but proposes a set of adaptation paths to the complete presentation defined during the content generation phase. In that case, the I_j Interactive layer that depends on the S_k Spatial layer cannot be activated until the S_k Spatial layer is activated. The `requiredFor{S|T|I}LayerValue` attributes are optional but ease the implementation of a stateless adaptation decision-taking process that simply follows layers requirements when available. The layer dependency mechanism is illustrated in Figure 58.

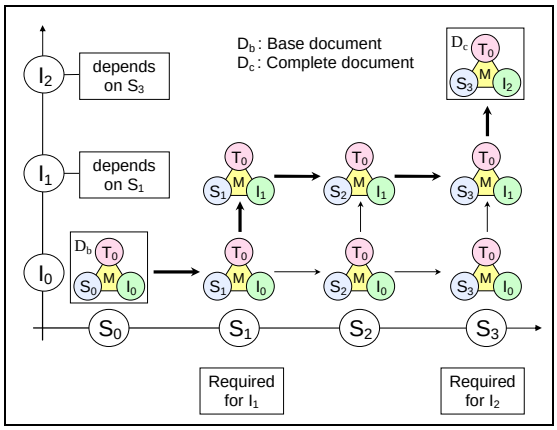


Figure 58: Constrained adaptation paths.

6.3.3.4 Dead-end adaptation paths

A single multimedia document may be used in several usages for which adaptation paths may diverge. These diverging versions do not fit well with the STI progressive approach because they do not aim to converge toward the same *Complete* presentation. Such a typical application scenario is illustrated by a print view of a multimedia document. Indeed, a printable multimedia document does not require any temporal and interactive presentation characteristics but requires greater resolutions than usual multimedia terminals. In that case, the *Spatial* description can be built progressively by applying *Spatial* layers to reach a suitable printable size but this layout will probably be incompatible with the temporal and navigation scheme defined by the content creator to grant the user media element access.

The *Scalable MSTI* model copes with such specific adaptation needs by defining the `blocks{S|T|I}LayerValue` attribute on an STI layer. When applied to a layer, this attribute specifies the highest layer from the STI component it is compatible with. In that case, the activation of the S_k *Spatial* layer that blocks the T_j *Temporal* layer prohibits the activation of the $[T_j, T_n]$ *Temporal* layers. In the opposite way, the S_k layer cannot be activated if one of the $[T_j, T_n]$ layers were previously activated. For instance, the print layer of the *Spatial* component may block the adaptation path on the *Temporal* and the *Interactive* axes to the base layer of the scalable document (`blocksTLayerValue="0"` and `blocksILayerValue="0"`). As a consequence, this mechanism introduces dead-ends on purpose in the adaptation graph as illustrated in Figure 59.

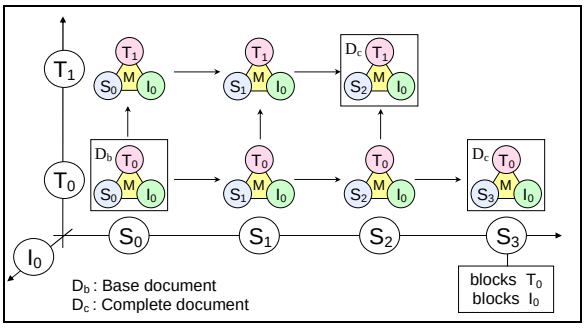


Figure 59: Adaptation path dead-end.

6.4 Examples of scalable scenes

The adaptation parameters that determine the organization the spatial, temporal and interactive properties of multimedia scenes into progressive STI scalability layers are left open in the *Scalability MSTI* approach. Hence, every domain-specific application which relies on a finite set of identified environment constraints can organize them into *primary* and *auxiliary* adaptation parameters as defined in Section 6.3.3. Of course, the best performances for our *Scalability MSTI* approach will be obtained by selecting adaptation parameters whose order relation is compatible with progressive *Spatial*, *Temporal* or *Interactive* descriptions. As previously illustrated in Section 6.3.2, each STI scalability axis has obvious progressive properties. In Figure 52, several display resolutions provide an increasing space for displaying the presentation (e.g. 320x240, 400x240 and 480x270). New empty space can be advantageously filled in with new media components (e.g. providing weather forecast for several days instead of one). In Figure 53, several levels of CPU requirements (e.g. low, medium and high) allow speeding up the rendering rate of the presentation. So, new media components can be inserted into the presentation or new animations can be defined to introduce presentation transitions (e.g. introducing optional fade-in or fade-out effects or even an animated bar graphs component into a commercial). In Figure 54, different memory requirements provide an increasing space for storing decoded images. Extra memory space can be used to store additional media components and ease the navigation within the content (e.g. providing coming EPG items instead of the current one only).

In the following, we illustrate the scalability of multimedia scenes on two additional examples. First, a two-level scalable scene is presented in Section 6.4.1. Although, such scene scalability seems a simplification of the coarse-grain granularity presented so far, it provides useful adaptation functionalities in a simple way. Second, a document targeting summarization scenarios is presented in Section 6.4.2. In that case, user's interests for the presentation determine the suitable layers of the scalable scene.

6.4.1 Two-level scene scalability

The example described in this section is an image gallery that can be presented as an advanced image slideshow illustrated in Figure 61 using the MSTI syntax (S_1,T_1,I_1). A simplistic or *Base* document presentation $(S_0,T_0,,I_0)$ has also been created as illustrated in Figure 60. This *Base* document only displays one slide that changes when the presentation is updated (no timed or interactive navigation).

These two MSTI documents can be merged into a single scalable scene which provides simple but common adaptation configurations that are described in Table 9. These eight versions illustrate a simple two-level MSTI scalability.

This image gallery has been described using SVG and JavaScript and some parts of its code are provided in Code 42, Code 43, Code 44, Code 48, Code 49 and Code 50[46]. The *Media* description of this document is composed of 6 images, 6 rectangles that create a 'shadow' effect behind the images and 4 'arrows' that can be enabled for forward, fast-forward, backward and fast-backward navigation. Two variables are also defined in the *Media*: `slideshowTempLock` and `slideshowPermLock` which

[46] A full version of this example can be found at http://www.tsi.enst.fr/mm/MSTI/ImageGallery.html

can be used to temporarily or indefinitely lock the slideshow when both the *Temporal* and *Interactive* components are jointly applied to the image gallery.

Figure 60: Example of a Base document for a scalable image gallery content.

Figure 61: Example of a Complete document for a scalable image gallery content.

Table 9: MSTI alternatives of an image gallery.

STI			Description
S_0	T_0	I_0	S_0 provides the position for a single image.
T_0 and I_0 are empty descriptions and do not grant the user access to other images.			
S_1	T_0	I_0	S_1 provides the position for 3 images aligned horizontally.
This layout can be activated when a higher resolution than S_1 is available and if the capabilities of the terminal are sufficient (e.g. display size, decoding memory…).			
S_0	T_1	I_0	T_1 animates a one-pane image slideshow on a periodic time basis of 5 seconds.
Transitions from one slide to the next are performed by sliding the current image to the left and inserting the next from the right. The user can therefore have access to the slideshow without any interactivity but this multimedia document requires the support of a timing function and sufficient processing power to move the images.			
S_0	T_0	I_1	I_1 enables the four graphic ‘arrows’.
These arrows belong to the *Media* description and have been positioned in S_0. I_1 appends an event listener to these four “button” shapes and describes the mechanisms which allow the user to replace the current image by the next (or the previous) one according to interactions on an available user interface (pointing device, keypad). A step of 3 images is defined for fast-forward and fast-backward buttons. Furthermore, I_1 also triggers a ‘shadow’ effect when focusing on an image with a pointing device (e.g. mouse).			

S_1	T_1	I_0	The combination of S_1 and T_1 generate a 3-pane image slideshow.
The spatial activation of the two additional images, that were not visible in S_0, enables animations that are part of T_1 but that were not executed in the (S_0,T_1,I_0) configuration. The transition between slides is performed by removing the last image on the left, sliding all the remaining images to the left and inserting a new image on the right.			
S_1	T_0	I_1	The combination of S_1 and I_1 generate a 3-pane interactive image gallery.
S_1 provides the *Spatial* description of the slideshow for every state illustrated in Code 48 and I_1 only triggers the slideshow state corresponding to a user's requests.			
S_0	T_1	I_1	The combination of T_1 and I_1 achieves a 1-pane (S_0) interactive slideshow.
In practice, the coexistence of the automated navigation paradigm and the user-centric navigation is managed by sharing common variables in the *Media* description. For instance, if the user is interacting with the slideshow, I_1 enables the `slideshowTempLock` variable of the *Media* description that will disable the slideshow scheduler defined in T_1. Indeed, T_1 checks this variable before automatically animating images and releases this lock after 2 seconds when triggered. Similarly the `slideshowPermLock` variable of the *Media* description is used to lock the slideshow when an image remains in selected mode (mouse over an image). Moreover, interactive features of the slideshow are enhanced when T_1 is applied since animations used as automated transitions between images are triggered when interacting with the multimedia document.			
S_1	T_1	I_1	The combination of all STI components results in a complete interactive slideshow with a 3-image display.

6.4.2 Summarization of scalable multimedia documents

The summarization of a multimedia document is a challenge that requires the summarization of media components combined into a document but also relies on an appropriate adaptation of its presentation. Our proposal published in the proceedings of the International Workshop on Image Analysis for Multimedia Interactive Services [2] consists in summarizing *Scalable MSTI* documents based on three adaptation parameters: a targeted level of expertise (Section 6.4.2.1), a preferred duration (Section 6.4.2.2) and a level of expectation for extended information (Section 6.4.2.3).

The layers of such a *Scalable MSTI* document can be configured according to user's interests. However, the simultaneous use of some enhancement layers may become awkward: a very detailed presentation that has been minimized to a very short duration may lead to a speedy and therefore unreadable succession of text-heavy content. For this reason, restrictions on the combined use of some scalability layers using *blocking* or *dependant* layers as introduced in Section 6.3.3 may be defined during the content generation phase to disable some configurations during playback.

6.4.2.1 Region Of Interest (ROI)

Document summarization may require reducing the quantity of multimedia data conveyed by the document at a given time to enable "at a glance" presentations. For instance, the summarization of the spatial properties of a multimedia scene can be performed by reducing the number of media components, by modifying the nature of elementary media (e.g. through transmoding techniques) or by simplifying the layout of the presentation. The efficiency of such a summarization process heavily relies on an appropriate evaluation of the importance of elementary media (e.g. key images but also styled rectangles that may associate an image and its caption). This evaluation can be given by the author using semantic languages and captured in the MSTI model through layers.

Using the *Scalable MSTI* approach, the level of details of the multimedia presentation at a specific time can be controlled by defining *Regions Of Interest (ROI)* adaptation parameters. Such adaptation axis can organize spatial properties into scalable layers that progressively complete the presentation with additional details while maintaining the same scene size. The position and size of a *ROI* can be modified from one *Spatial* layer to the next and the elementary media that are part of a *ROI* can also be updated. Three *ROI-based* summarized presentations of a scalable multimedia document are illustrated in Figure 62.

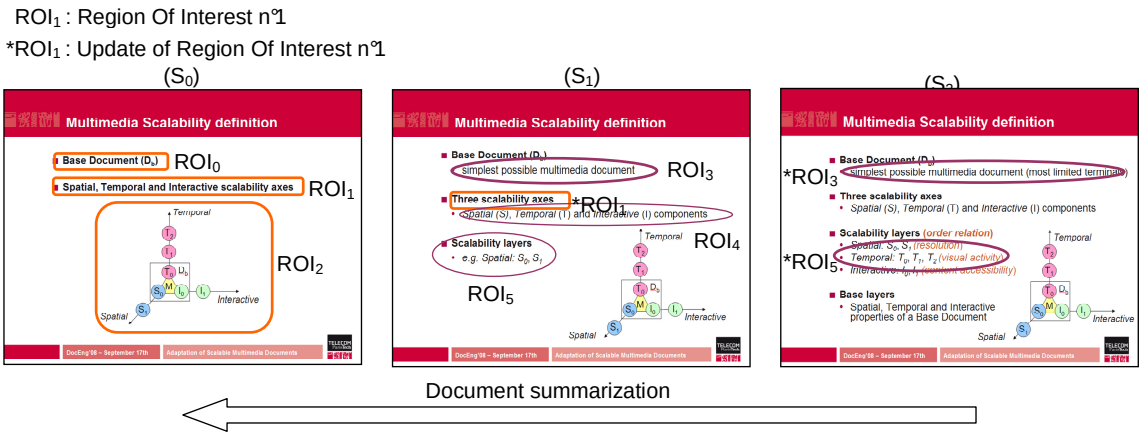


Figure 62: ROI-based document summarization.

6.4.2.2 Sequence Of Interest (SOI)

Document summarization may require reducing the quantity of multimedia data proposed by the document over time to shorten the presentation duration. For example, the summarization of the temporal properties of a multimedia presentation can be accomplished by reducing the number of elementary media to be sequentially displayed, by shortening the duration of some of them or by modifying the timing properties of the scene. The optimization of such a summarization process mainly depends on the selection of elementary media of the timed summary (e.g. degree of relevance for the presentation understanding).

Using the *Scalable MSTI* approach, the quantity of multimedia data over a period of time can be specified by defining appropriate *Temporal* layers. Indeed, each layer of the *Temporal* axis can be organized according to a *Sequences Of Interest (SOI)* adaptation parameter that progressively extends the presentation with additional media components relevant to the presentation topic as content duration increases. The starting date and duration of a *SOI* can be modified from one *Temporal* layer to the next. Two *SOI-based* summarized presentations of a scalable multimedia document are illustrated in Figure 63.

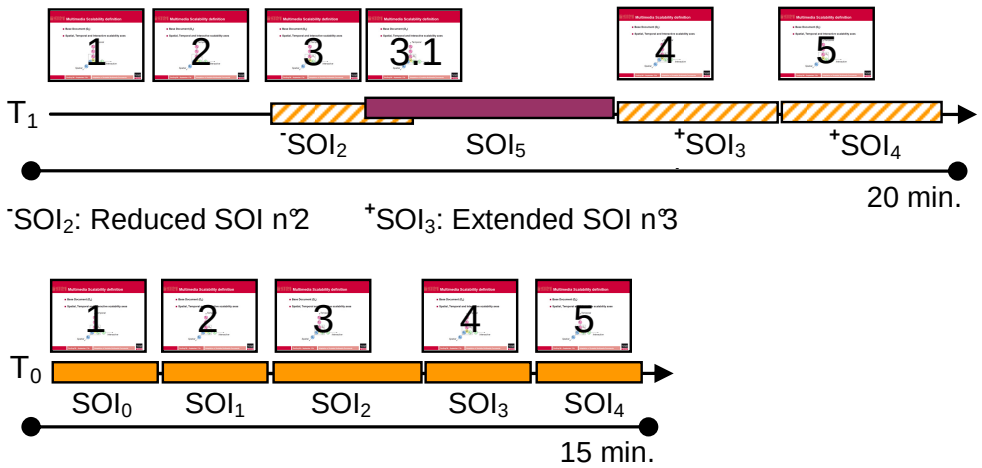


Figure 63: SOI-based document summarization.

6.4.2.3 Action Of Interest (AOI)

Document summarization may require simplifying the visual interface of a document to enhance presentation usability. For instance, the summarization of the interactive properties of a multimedia scene can be performed by reducing the number of user interactive means (e.g. hyperlinks), by reducing the number accessible elementary media or by defining limited but simple navigation paradigms. The efficiency of such a summarization process mainly relies on the quality of the media selection that is available because a user interface is only a tool to access them. Indeed, a limited access to media components enable a simpler interface but such improvement in the content usability is not satisfactory if essential media components cannot be accessed.

Using the *Scalable MSTI* approach, the quantity of accessible multimedia data through user actions can be defined by *Interactive* layers. Therefore, *Actions of Interest (AOI)* adaptation parameters can be associated with each layer of the *Interactive* axis so that they progressively enable an in-depth understanding to interested users by providing a growing access to auxiliary media components. The navigation paradigms of an *AOI* can be modified from one *Interactive* layer to the next and media components that are part of an *AOI* can be updated. Two *AOI*-based summarized presentations of a scalable multimedia document are illustrated in Figure 64.

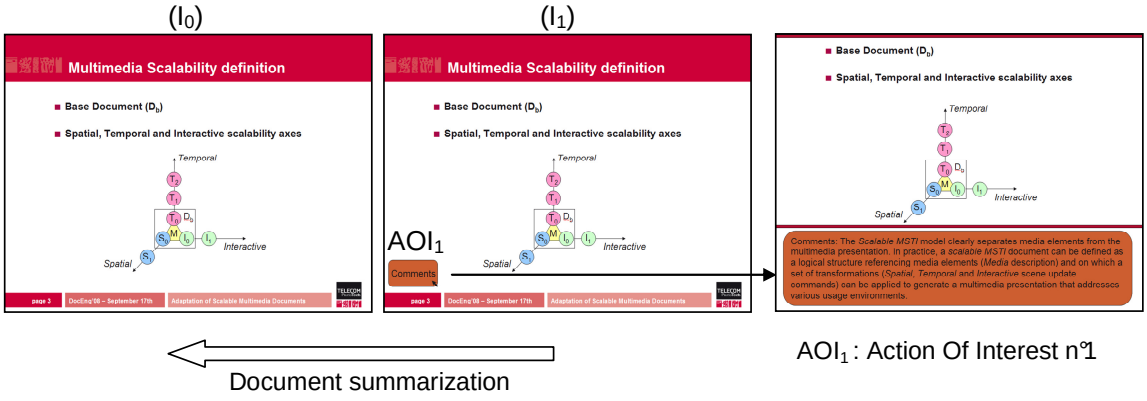


Figure 64: AOI-based document summarization.

6.5 Experiments and results

In this section, we present experiments that have been conducted on *Scalable MSTI* documents. First, our *Scalable MSTI* documents have been validated using standalone adaptation tools and local playback. Section 6.5.1 discusses the results of these tests. Then, experiments have been conducted in the adaptation environments previously described in Section 5.2.2 (streaming in the MPEG-21 framework) and in Section 5.3.3 (broadcast in the T-DMB framework). In particular, a focus on the joint adaptation of scalable media and a *Scalable MSTI* scene is presented in Section 6.5.2. Additionally, the dynamic adaptation of one-axis *Scalable MSTI* scenes is described in Section 6.5.3.

6.5.1 Scalable MSTI document validation

The first objective of our experiments was to evaluate if the various adaptation paths proposed by our *Scalable MSTI* scenes were addressing targeted usage environments independently from the ADTE algorithm or implementation. For that purpose, we developed several adaptation decision-taking engines in JavaScript and applied them to a set of *Scalable MSTI* documents such as those described in Section 6.4. These different decision-taking engines take into account different implementation constraints. For instance, some of them are stateless while others progressively use the previous steps in the adaptation path to influence the selection of the new path. In the same way, some of them process scalability layers in the document order while others first build the adaptation graph and then decide on the adaptation path. These algorithms have been designed such that they can be instructed to reach a desired STI

configuration (i.e. an S_i, T_j, I_k triplet). One significant result that comes out from our experiments is that for all algorithms and for all STI configurations, a path cannot always be found but a fallback is always available, the worst case being the *base* presentation.

The second objective of our experiments was to evaluate if the adaptation paths found by our algorithm led to relevant presentations. Obviously, the answer is true since any algorithm will provide options designed by the author, this option approaching the maximal quality as defined by the author himself.

The observation of generated scenes through local playback also showed that the adaptation process conforms to author's expectations as soon as the separation of the STI scene properties is respected as explained in 6.2. In particular, the generation of each STI layer as defined by the *Scalable MSTI* model ensures that the adapted presentation will be the same whatever adaptation path was used to reach an STI configuration. So, the validation of content quality requires testing each presentation alternative. This can be a tedious task but probably less than the separate authoring of each scene alternative. Of course, the check of each possible presentation of the adaptation graph becomes optional as soon as the multimedia production chain relies on standardized templates as described in Appendix D. However, an additional validation for `skippable` STI layers has to be done in order to guarantee their appropriate authoring. Indeed, `skippable` layers introduce *digressing* (Section 6.3.3.2) or even *forking* adaptation paths (Section 6.5.2.2) that need to be tested. For this reason, the use of `skippable` layers has been limited during our experiments because this optional feature potentially complicates an extensive validation by significantly increasing the number of possible adaptation paths. When used, these optional STI layers have always been coupled with immediate RAL enhancements (although it does not have to be case) and limited to well-identified adaptation scenarios even though they could also be used to further optimize performances when quickly stepping through an adaptation graph.

During our experiments, we tested different decision-taking algorithms to navigate through the adaptation graph. For instance, a straightforward adaptation decision-taking engine consists in first stepping into spatial scalability layers until the limit is reached, then into temporal and interactive scalability layers. Another straightforward approach consists in applying alternatively spatial, temporal and interactive scene properties. These experiments revealed that the progressive steps in the adaptation graph can influence the final decisions. In particular, the combined use of `skippable`, `dependsOn` and `blocks` STI layers can generate independent paths (as illustrated in Sections 6.3.3.2, 6.3.3.4 and 6.5.2.2) that memoryless implementations cannot step back. As a consequence, several levels of priority can be set for the three *primary* adaptation parameters of a *Scalable MSTI* document so that the adaptation decision-taking engine can maximize content quality based on this additional criterion. As illustrated in Figure 59, a *Spatial* layer that blocks *Temporal* and *Interactive* axes in order to introduce specific spatial presentation characteristics (e.g. a print-view of the document) should only be applied if the adaptation parameter associated with the *Spatial* scalability axis has an absolute priority.

Finally, our trials showed that although a decision-taking engine can obviously select any presentation alternative of the adaptation graph, some layers do not necessary lead to presentation enhancements. For instance, some layers possibly introduce scene elements that are required for other *dependent* layers (`dependsOn`) but do not provide presentation enhancements on their own. As a consequence, different MSTI scene options of an adaptation graph may lead to the same presentation. In order to guide the adaptation decision-taking engine to optimized scene, we recommend defining *required* STI layer (`requiredFor`) in *Scalable MSTI* documents so that each adaptation step in the STI graph can be made by following the adaptation mainstreams. As a consequence, general rules can be set when implementing an adaptation decision-taking algorithm for the processing of *Scalable MSTI* scene. For instance, one of our adaptation decision-taking engines has been developed to favor all paths leading to *random access* layers or *dependent* layers. Indeed, such adaptation paths are more likely to follow author's intention as illustrated in Figure 65.

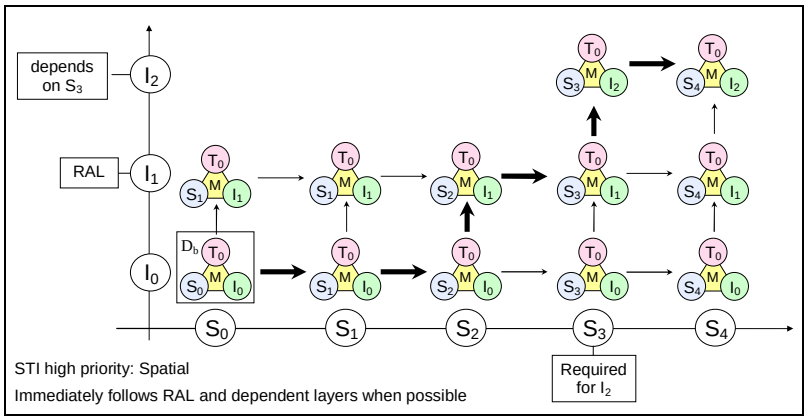


Figure 65: Author-oriented adaptation path.

6.5.2 Media-oriented scene adaptation

Multimedia scene scalability has been experimented as part of the MPEG-21 framework as introduced in Section 5.2.1. Contrary to our previous contribution (Section 5.2) which consisted in selecting and synchronizing independent scene updates according media-level decisions, *Scalable MSTI* scenes can be handled by an MPEG-21 DIA engine as any scalable content using the MPEG-21 Bitstream Syntax Description Language [35] (i.e. without requiring any specific scene adapter). As a consequence, the on-demand streaming of adapted scene according to the user's context did not require any additional implementation efforts than configuring an MPEG-21 DIA engine dealing with multiple `AdaptationQoS` by relying on global UCD [91].

In the following, two main aspects related to the concurrent adaptation of scalable media along a scalable scene are described. First, the combined scalability of media and scene is presented in Section 6.5.2.1. Then, the extension of adaptation features of the *Scalable MSTI* model to fork some adaptation parameters is described in Section 6.5.2.2.

6.5.2.1 Scalable media and scalable scene

The experiments conducted in Section 5.3.3 on the adaptation of a presentation according to media-level decisions were driven by the need to adapt greedy MPEG-4 SVC video content. Additionally, the extension of adaptation scenarios to simple receivers, such as handheld terminals unable to cope with the MPEG-4 SVC codec at that time, was also considered. However, the use of *Scalable MSTI* scenes opens new adaptation scenarios since presentation decisions can be taken without any media-level decision impulse. As a consequence, two compatible ways to handle media and scene scalability can be envisaged in the MPEG-21 framework.

First, the scalability axes of the MPEG-4 SVC video can drive the scalability axes of the *Scalable MSTI* scene. For instance, the display resolution *(Spatial)*, the image frame rate *(Temporal)* and the content quality *(Interactivity)* can define the adaptation parameters of the STI scalability layers. In that case, each media-level decision corresponds to an appropriate scene decision and can trigger specific scene transitions.

Second, the scalability of the MPEG-4 SVC video can be considered as a standalone adaptation. Indeed, image resolution, frame rate and PSNR quality can all contribute to reduce the required bandwidth to transmit a video sequence displayed at a given screen size. In that case, the adaptation parameters of the STI scalability layers can be freely defined independently from the media scalability.

Of course, both scenarios can be combined as illustrated in the `AdaptationQoS` descriptors provided in Code 51 and Code 52. In this example, the resolutions that drive the adaptation of the video have been mapped onto the scene size (400x240, 800x480 and 1280x1024) but still correspond to typical video resolution (320x240, 640x480 and 720x540). The scalable video and the *Scalable MSTI* scene share a common `IOPin` (display width) that is handled through a global UCD as described in [90]. Hence, each

modification of the video resolution results in a suitable scene adaptation update. Furthermore, the video can be adapted to the available bandwidth modulating its quality or frame rate. In the same way, the multimedia scene can be adapted according the receivers capability to decode animations and their memory.

```
<DIA xmlns="urn:mpeg:mpeg21:2003:01-DIA-NS"
     xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

  <Description xsi:type="AdaptationQoSType">
    <Module xsi:type="LookUpTableType">
      <Axis iOPinRef="DISP_WIDTH">
        <AxisValues xsi:type="IntegerVectorType">
          <Vector>400 800 1280</Vector>
        </AxisValues>
      </Axis>
      <Axis iOPinRef="TEMPORAL">
       <AxisValues xsi:type="IntegerVectorType">
        <Vector>0 1 2 3</Vector>
       </AxisValues>
      </Axis>
      <Axis iOPinRef="FGS1">
        <AxisValues xsi:type="IntegerVectorType">
          <Vector>0 20 40 60 80 100</Vector>
        </AxisValues>
      </Axis>
      <Content iOPinRef="VIDEO_WIDTH">
        <ContentValues xsi:type="IntegerVectorType" mpeg7:dim="3">
          <Matrix>320 640 720</Matrix>
        </ContentValues>
      </Content>
      <Content iOPinRef="VIDEO_HEIGHT">
        <ContentValues xsi:type="IntegerVectorType" mpeg7:dim="3">
          <Matrix>240 480 540</Matrix>
        </ContentValues>
      </Content>
      <Content iOPinRef="PSNR">
        <ContentValues xsi:type="FloatMatrixType" mpeg7:dim="3 4 6">
          <Matrix>31.09 31.72 32.35 33.01 33.95 34.79 28.62 29.03 ...</Matrix>
        </ContentValues>
      </Content>
      <Content iOPinRef="targetBitrate">
        <ContentValues xsi:type="IntegerMatrixType" mpeg7:dim="3 4 6">
          <Matrix>136 160 183 207 231 254 81 98 ...</Matrix>
        </ContentValues>
      </Content>
    </Module>
  </Description>
</DIA>
```

Code 51: A scalable video MPEG-21 AdaptationQoS.

6.5.2.2 Forking adaptation parameter values

The *Scalable MSTI* model enables three main incremental adaptation axes. When one of these three main adaptation axes is reserved to define media-oriented adaptation, it cannot be used for another purpose. More generally, all adaptation parameters that cannot be mapped onto an existing presentation in the adaptation graph cannot be simply defined as *auxiliary* adaptation parameters or *digressing* values (as introduced in Section 6.3.3 and Section 6.3.3.2 respectively). Indeed, such exceptions constantly repeated in the adaptation graph are possible but significantly increase content redundancy since the progressive features of our *Scalable MSTI* model cannot be used in that case. Instead, a fork can be defined in order to extend the adaptation graph with presentation exceptions that are entirely incompatible with the selected scalable adaptation parameters and that do not reduce coding efficiency of the main presentation.

Forking adaptation parameter values are mapped onto scalable layers that depend on their preceding layers, that may be applied (or not), as *digressing* adaptation parameter values do, but that cannot be updated to return to the main adaptation path. Such isolated *Spatial*, *Temporal* or *Interactive* layers can

be used when further enhancements are not envisioned or if embedded scene properties would significantly impact succeeding layers (and therefore introduce undesirable redundancy). Technically, these isolated layers can be defined by the *Scalable MSTI* model through the combined use of *skippable* and *blocking* layers or can originate from diverging adaptation paths associated with a combination of *skippable* and *dependent* layers that were individually introduced in 6.3.3. Both approaches are illustrated in the following.

```
<DIA xmlns="urn:mpeg:mpeg21:2003:01-DIA-NS"
     xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

  <Description xsi:type="AdaptationQoSType">
    <Module xsi:type="LookUpTableType">
      <Axis iOPinRef="RESOLUTION_X">
        <AxisValues xsi:type="IntegerVectorType">
          <Vector>400 800 1280</Vector>
        </AxisValues>
      </Axis>
      <Axis iOPinRef="ANIM_SUPPORT">
        <AxisValues xsi:type="BooleanVectorType">
          <Vector>false true</Vector>
        </AxisValues>
      </Axis>
      <Axis iOPinRef="INTERNAL_MEM">
        <AxisValues xsi:type="IntegerVectorType">
          <Vector>0 100 200 500 1000</Vector>
        </AxisValues>
      </Axis>
      <Content iOPinRef="RESOLUTION_Y">
        <ContentValues xsi:type="IntegerVectorType" mpeg7:dim="3">
          <Matrix>240 480 1024</Matrix>
        </ContentValues>
      </Content>
      <Content iOPinRef="targetBitrate">
        <ContentValues xsi:type="IntegerMatrixType" mpeg7:dim="3 2 5">
          <Matrix>4 7 12 30 40 4 8 13 ... </Matrix>
        </ContentValues>
      </Content>
    </Module>
  </Description>
</DIA>
```

Code 52: A Scalable MSTI MPEG-21 AdaptationQoS.

First, a *forking* adaptation parameter can be related to a very specific usage configuration that requires a customized presentation which clearly differs from the *Complete* presentation of scalable multimedia document. For instance, a typical adaptation axis for the *Interactive* layers can provide progressive and complementary interactive means: 2-state keypad (left-right), touch screen (click), 5-state keypad (left-right-up-down-ok), trackball (isOver) as illustrated in Figure 66. Integrating the iPhone's multi-touch navigation paradigm as a *digressing* adaptation parameter in such an adaptation graph is possible but would not be efficient because the whole interactivity mechanism will be very different. Indeed, the iPhone's interface releases left-right-up-down button requirements and very specific interactive behaviors can be triggered by the built-in accelerometer. As illustrated Figure 66, such specific adaptation parameter values can be mapped onto layers identified as *skippable* and *self-blocking*.

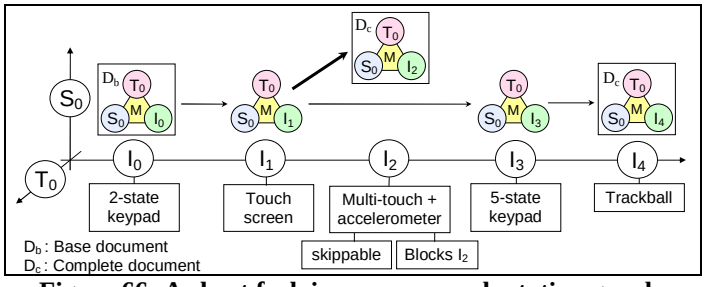


Figure 66: A short fork in a one-way adaptation graph.

Second, forking adaptation parameters can also be defined to initiate a scalable alternative to the main adaptation axis. In fact, one (or more) forks could be defined in parallel to their main adaptation axis in order to define alternative scalable versions of the same document. One typical example of such forking adaptation parameter is document internationalization. Indeed, multi-lingual support cannot be considered as an adaptation axis since an order relation cannot be defined between several languages. However, language selection can be considered as visual activation and can have significant consequences on the document layout. Language selection would impact the *Spatial* component of a scalable multimedia document. To cope with such a use case, a language-based fork can be defined in the adaptation graph as illustrated in Figure 67. In that case, this *forking* adaptation parameter would be mapped onto layers identified as *dependent* and *skippable*.

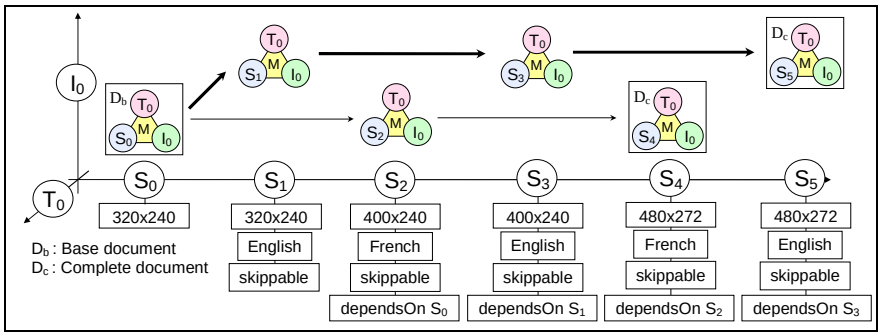


Figure 67: A long fork in a one-way adaptation graph.

6.5.3 Scene adaptation in broadcast environments

Multimedia scene scalability has been experimented in the T-DMB broadcast environment as introduced in Section 5.3.3. In our previous contribution (Section 5.3), the basis for scene scalability has been defined by experimenting adaptation updates that address progressive resolution capabilities. However, no scene description model was defined at that time to organize scene properties according to adaptation parameters. Both aspects (adaptation updates and adaptation parameters) could even remain independent by organizing presentation alternatives by just minimizing their difference. In that case, adaptation parameters were not progressive at all and now correspond to the *auxiliary* adaptation parameters of the *Scalable MSTI* model.

In the following, two main aspects related to the broadcast of *Scalable MSTI* multimedia services are discussed. First, the dynamic computation of the *Scalable MSTI* adaptation graph is discussed in Section 6.5.3.1. Then, the transposition of a *Scalable MSTI* scene description to a one-dimension scalable bit stream is described in Section 6.5.3.2.

6.5.3.1 Dynamic scene adaptation

In a first step, a multimedia service has been considered as a sequence of independent multimedia documents as far as the adaptation of its presentation is concerned. As a consequence, two complementary adaptation approaches were experimented in our case.

First, the test sequences we developed for digital radio services basically include two types of dynamic content: dynamic scene parameters (e.g. the quotes of the stock exchange) and dynamic media components (e.g. at a glance presentation of the on-air content). These scene properties are constantly replaced over time but the scope of these transformations always remain limited. As a consequence, we assumed that the modifications brought by such scene updates did not impact the decisions taken for the adaptation of the current scene. In that case, each *Scalable MSTI* scene update can be processed as an individual document without taking into account previous adaptation decisions.

Second, the programs of a radio station rely on very different presentation canvas. Therefore, switching from a multimedia program to another (e.g. from music to talk) implies significant scene element replacements. As a consequence, we assumed that the modifications brought by such scene transformations were so important that the complete adaptation process had to be recomputed from the

start. In that case, the *Scalable MSTI* scene update constitutes a complete refresh of the document in its current state and can be processed as an individual document without taking into account previous adaptation decisions.

Thanks to these two assumptions, we have been able to show the seamless adaptation of *Scalable MSTI* services in the course of a radio program. Additionally, visual transitions have been used to hide adaptation artifacts that may occur in between two programs due to the reloading of the presentation. However, this approach is not satisfactory because it does not leverage the full flexibility of the *Scalable MSTI* model where *Random Access Layers* can be defined. In the same way, these layers can be used to step back in the adaptation graph when dynamic adaptation parameters are considered. They could also be used to deal with dynamic content. In fact, such an enhanced walkthrough into a dynamic adaptation graph would require additional signaling attached to each scene update. These additional descriptors would indicate which scalable layers are affected by scene transformations and would possibly update the adaptation parameter values of the current adaptation graph. In that case, the adaptation decision would be reprocessed starting from the lowest modified STI layers. This optimization to be experimented is illustrated in Figure 68 on an example. At time t, the presentation is composed of (S_4, T_0, I_2) scalability layers. A time t+1, a presentation update is received and targets the following scalability layers: S_2 and I_2. Since these updates may impact the current adaptation decision, it has to be reprocessed, at least, starting from (S_2, T_0, I_2) layers according to their updated adaptation parameter values. Since the closest random access layers lead to the (S_1, T_0, I_1) layers, new adaptation decisions can be taken from that point.

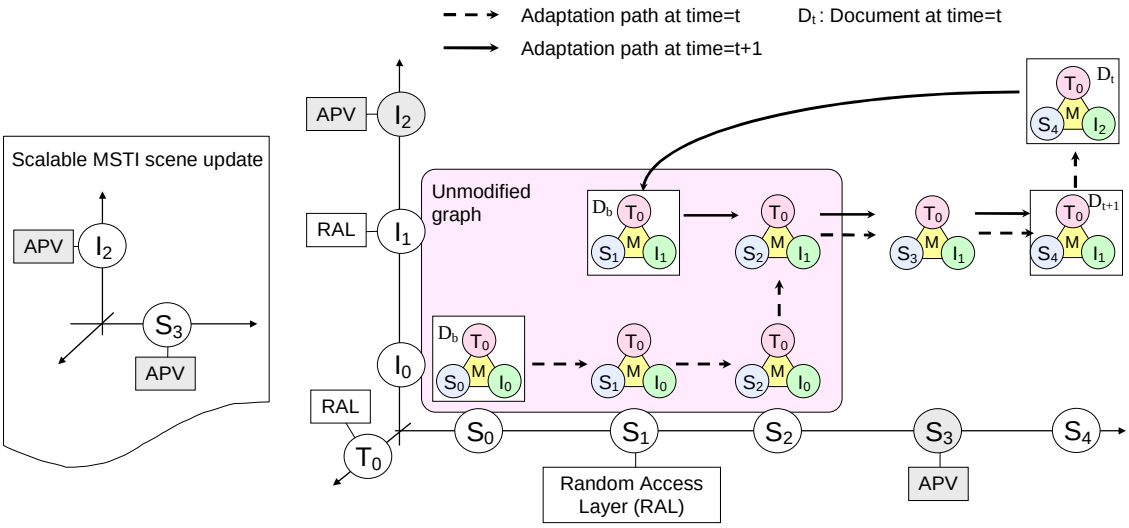


Figure 68: Dynamic adaptation path with Random Access Layers.

6.5.3.2 Collapsing adaptation parameters

Although the *Scalable MSTI* model was designed to handle three progressive adaptation parameters each focusing on one of its *Spatial*, *Temporal* and *Interactive* axis, the adaptation requirements of some application scenarios might be restricted to a single scalability axis. For instance, the spatial adaptation of multimedia scenes presented in Section 5.3 only covers a single progressive adaptation parameter (i.e. the display resolution). Still, this progressive adaptation modifies all scene properties. We present how the *Scalable MSTI model* can be used to represent the experiments of Section 5.3. In order to reproduce such simplified multimedia scalability, two approaches have been used: one where the content is prepared with the one adaptation parameter in mind, and the other one where the content was prepared with several adaptation parameters.

First, the *Scalable MSTI* scene can be prepared according to a specific use case focusing on a single adaptation axis. For instance, temporal and interactive enhancements can be directly driven from the spatial properties of the presentation. In that case, each *Temporal* and *Interactive* scalability layers strictly depends on a specific *Spatial* layer so that the adaptation graph is restricted to a single but progressive path as illustrated in Figure 69.

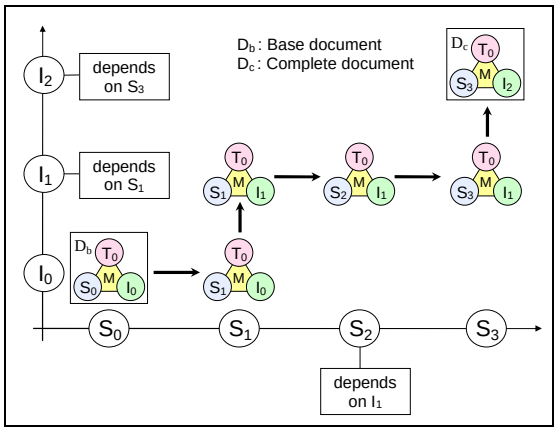


Figure 69: Example of a single-path adaptation graph.

Second, the *Scalable MSTI* scene can be prepared with several adaptation parameters, taking into account broader use cases, and collapsed before being transmitted as illustrated in Figure 70. In the scope of the *Scalable MSTI* approach, collapsing means aggregating scene transformations of different STI layers into one adaptation update. Once collapsed, the STI layers can still be labeled with adaptation parameters, possibly with the aggregation of the individual layer's adaptation parameter, in order to define a limited set of typical configurations. During our experiments, *Scalable MSTI* layer collapsing has been used to generate scalable multimedia documents that can be progressively transmitted over narrow-bandwidth networks. Indeed, scene scalability is used to organize scene properties in such a way that a progressive playback of the document will provide multimedia enhancements over time. In that case, the scalable scene bit steam has lost all its adaptation functionalities during the collapsing phase but in some scenarios, adaptation is not required.

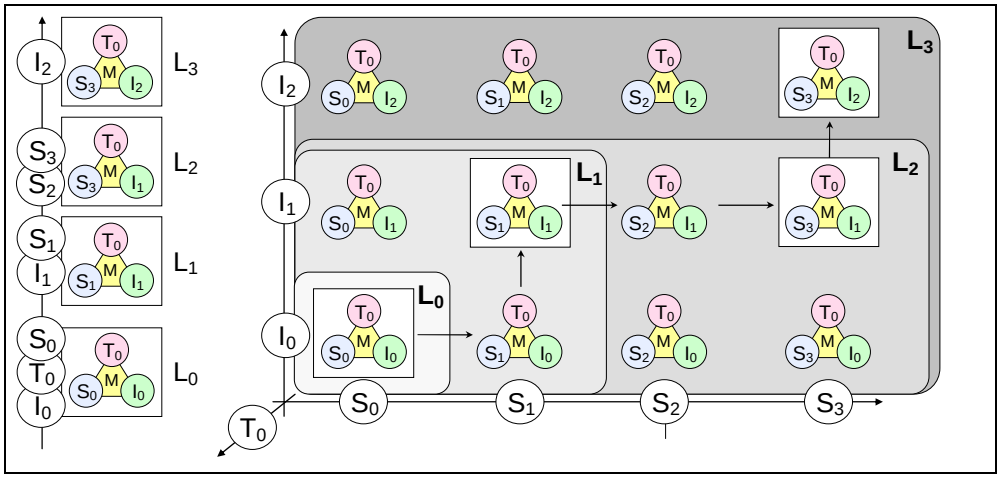


Figure 70: Collapsing scalability layers.

6.6 Conclusion

In this section, we presented our *Scalable MSTI* model which specifies the separation of the media structure (*Media*) and the scene properties in progressive scalable layers (*Spatial*, *Temporal* and *Interactive*). The flexibility offered by *Scalable MSTI* scenes has been illustrated through the concept of adaptation graph. The proposed adaptation decision-taking algorithms rely on a concise set of rules that define multiple adaptation paths within this graph. Adaptation paths guide the selection between several content alternatives to a presentation optimum by selecting the highest scalability layers that fit its usage environment. The proposed transformation process for the adaptation of scene properties is divided into cascading steps, called STI compositions. An initial non-adaptable multimedia content, called a *Base* document, is enhanced into several content alternatives according to the user's context by applying progressive adaptation updates that enrich its presentation.

The combination of any of our adaptation decision-taking algorithms and adaptation updates define an adaptation framework that fulfils the requirements defined in Section 5.1. Thus, the direct handling of standardized multimedia formats enables the display of all the fancy features that may be present *(advanced multimedia)*. Abstract STI scalability layers allow the implementation of an adaptation engine that does not have to deal with the specific presentation models of the numerous scene formats *(generic)*. The hierarchical organization of scene properties, possibly tagged with adaptation parameters, is self-sufficient so that an adaptation engine can take an appropriate decision and can transform the content accordingly without any outside parameters *(autonomous)*. The selection of adaptation parameters and the authoring of the scene properties of a limited set of STI scalability layers guarantee a full control over the content quality for editors *(control)*. The dynamic nature of an adaptation graph which can be re-computed while minimizing presentation transformations can smoothly handle content evolution over time and context fluctuation *(multimedia service)*. Finally, scene adaptation updates are efficient presentation transformations which also enable a configuration of adaptation overhead through adaptation granularity *(low-overhead)*.

In terms of content generation, each scalability layer of a *Scalable MSTI* document needs to be appropriately filled in with suitable scene properties in order to address targeted adaptation scenarios. Although, we did not propose authoring paradigms for our model, general authoring techniques can be defined in order to address these new challenges raised by the production of adaptable presentations based on the *Scalable MSTI* approach [1]. In short, four major steps in the authoring of scalable multimedia documents can be identified as illustrated in Figure 71.

The first step (step 1) follows traditional authoring paradigms: it consists in selecting elementary media and designing an appropriate presentation that is simple enough to fit the most constrained usage environments. This initial authoring phase aims at generating the *Base* presentation of a scalable multimedia document where the presentation is split into *Spatial*, *Temporal* and *Interactive* descriptions.

The second authoring step (step 2) adds one enhancement layer for each *STI* axis to create a *Complete* presentation that targets the most advanced usage conditions. Such an advanced presentation can benefit from a close cooperation of spatial, temporal and interactive scene properties in order to offer to the user an access media components in various ways. At this step, a coarse-grained scalable multimedia document featuring height presentation versions is already available and might provide sufficient adaptation options for targeted application scenarios.

The two last authoring steps aim at giving adaptation granularity to the generated multimedia documents in order to address a larger set of usage environments. They consist in defining targeted adaptation scenarios and in (step 3) and designing incremental presentation versions that will ensure an appropriate presentation in most usage environments (step 4).

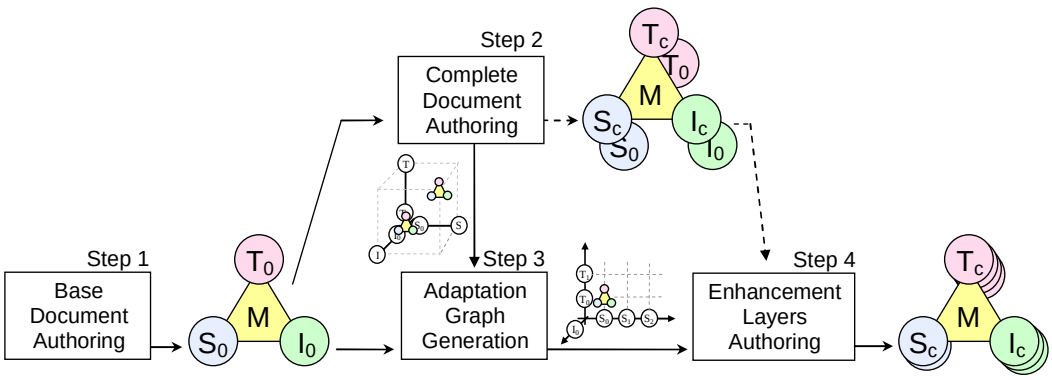


Figure 71: Four-step authoring of a Scalable MSTI document.

Although the *Scalable MSTI* approach has been designed in a broader scope than multimedia digital radio scenarios, multimedia scene scalability is not always the best solution for all adaptation use cases or even for all the adaptation aspects of a presentation. In particular, a closer cooperation between the *Scalable MSTI* approach and the alternative-based approach, which is commonly implemented in scene formats (Section 4.2.3), is an interesting hybrid model where presentation enhancements can be expressed using one or the other approach. For instance, such editorial choices can be driven by the need to enable some adaptation scenarios based on non-progressive context parameters (e.g. media codecs). In the same way, adaptation approaches based on scene plasticity (Section 4.2.4) can be advantageously combined with multimedia scene scalability. For instance, the interpolation of spatial scene properties, which is commonly available in vector graphics formats, enables fine adaptation granularity at a minimal cost. Since the *Scalable MSTI* model is a novel approach to represent multimedia scenes that covers multiple aspects of multimedia presentation (multimedia scene adaptation but also progressive scene playback, live scene streaming or broadcasting, etc.), the STI layers of our scalable model can also be used to convey additional scene adaptation mechanisms.

Chapter 7 Conclusion

In this dissertation, we have studied scalability concepts for multimedia scenes by introducing a new adaptation decision-taking method for multimedia presentations based on a 3D graph and a new presentation transformation based on scene updates. The *Scalable MSTI* model we developed stands for scalable *Media*, *Spatial*, *Temporal* and *Interactive* scene description model and defines a concise representation for scalable multimedia documents. It enables *a generic, autonomous and low-overhead transformation process for the controlled adaptation of advanced multimedia services*.

7.1 Summary of this work

7.1.1 Scene description model

Our study of state-of-the-art approaches tackling the adaptation of multimedia presentations in Chapter 4 concludes into a clear opposition between general document models such as ZYX [24], AHM [48] or Madeus [79] and abstract document models such as MSSA [75], MM4U [101], constraint-based [84] and artistic resizing [36] approaches. On the one hand, general document models, also called meta-model approaches in our classification, are designed to identify sophisticated presentation requirements and to integrate them in a unique model which represents a multimedia presentation. On the other hand, abstract document models, also called meta-format approaches, are based on the analysis of existing scene formats and represent some of the properties of their underlying presentation model (or all of them in the case of meta-format approaches such as MM4U). Both approaches have advantages and limitations. The meta-model approach enables an adaptation flexibility but significantly narrows down the presentation capabilities in order to allow content export into standardized formats. The meta-format approach can take full advantage of all presentation facilities of standardized formats but the adaptation flexibility remains limited by their underlying presentation model.

In this dissertation, our design choices were driven by the assumption that adaptation functionalities constitute an added value to an existing non-adaptable multimedia service. Indeed, the need for adaptation features often comes from the fact that some users cannot enjoy the proposed multimedia experience with a sufficient quality even though the service has been tailored to maximize its audience satisfaction. Such an assumption disqualifies meta-model approaches because they would require throwing away existing multimedia platforms which possibly provide optimized services to a large majority of their users. Instead, we propose an abstract document model, the so called *Scalable MSTI* model, which cannot be considered as a genuine meta-format designed for multi-publication, since only the native characteristics of presentation (spatial, temporal and interactive properties) of a given scene format – the concrete expression of a scene description – are abstracted to specify adaptation scenarios as in CSVG [17] or enable progressive playback [102]. Therefore, the *Scalable MSTI* model is a scene description model that structures the scene of multimedia documents (or services) into hierarchical parts and does not depend on their formats.

7.1.2 Multimedia scene adaptation

The core of our approach relies on the definition of scene scalability principles that aim at extending the presentation characteristics of an existing multimedia document. In adaptation scenarios, an initial document can be tailored to address most, if not all, usage environments. Multimedia scene scalability can then be used to enhance a poor multimedia experience, provided by such a *Base* document, with additional presentation characteristics according to the user's context. Due to our broadcast requirements inherited from the multimedia digital radio scenarios, as described in Chapter 2, our design choices were essentially driven by the minimization of adaptation processing overheads on handheld platforms and of bandwidth overheads dedicated to the adaptation signaling. On the one hand, processing overheads (e.g. extra processing power, battery consumption or memory requirements) that must be supported by radio receivers to access multimedia enhancements always need to be weighed against user's benefits. On the other hand, the division of a multimedia digital radio service into several presentation layers is always a

trade-off between content quality (e.g. content access time and/or the weight of content data) and adaptation capabilities given a fixed and limited bandwidth. As a consequence, our *Scalable MSTI* model relies on two processes. First, it relies on an efficient transformation that performs the adaptation of the multimedia presentation (adaptation updates). Second, it integrates a simple but powerful decision-taking algorithm that drives the adaptation process according to the author's wishes (adaptation graph).

The source of the adaptation functionalities of the *Scalable MSTI* model, introduced in Chapter 6, is the adequate selection of a presentation option within several alternatives. In our approach, the representation of these presentation alternatives has been optimized by gathering their spatial, temporal and interactive scene properties into progressive descriptions (respectively *Spatial Temporal* and *Interactive* scalability layers) that enhance an initial *Base* description (the *Media* description). In order to guide context-driven content adaptation approaches, each STI scalability layer can be associated with several adaptation parameter values which provide their application requirements (e.g. a minimal display resolution of 320x240 pixels). As a consequence, the adaptation of the various media components of the scene is processed as a group (STI layers) in the *Scalable MSTI* approach, instead of being processed individually for each media component, as it is proposed in several state-of-the-art alternative-based models introduced in Chapter 4. This approach has the benefit to explicitly take into account the semantic links that gather the elementary media of a presentation.

The serialization of such scalability layers dedicated to adaptation scenarios, called adaptation updates, is defined in Chapter 5 and has been experimented within two test beds. First, the adaptation of scalable video content along with its presentation using adaptation update has been explored and implemented in an MPEG-21 IP-streaming environment [6]. Second, the performances of adaptation updates for advanced multimedia presentations have been evaluated against state-of-the-art approaches applicable for the spatial adaptation of multimedia presentations in a T-DMB broadcast environment [5]. In both cases, adaptation updates have proven to be more efficient in terms of processing power, memory requirements and bandwidth consumption. Additionally, our experiments showed the adaptation flexibility offered to content providers. Indeed, the progressive requirements of adaptation updates allows to efficiently support very limited receivers by carrying most of adaptation overheads over enhancement layers targeting more advanced terminals in broadcast environments. Finally, the bandwidth consumption of our approach can be controlled by collapsing adaptation updates (thus configuring the granularity of scalable content) or by introducing progressive multi-channel transmissions [13].

The *Scalable MSTI* model is a transposition of the scalability of media standards such as MPEG-4 SVC [104] or JPEG 2000 [85] to the multimedia document domain. As a consequence, it includes scalability axes (*Spatial*, *Temporal* and *Interactive* axes) that are each associated with an order relation. The mapping between scene properties (scalability layers) and some context requirements (adaptation parameters) is not specified in our model in order to ease the implementation of domain-specific application scenarios. Instead, we drive decision-taking algorithms by guiding the selection of abstract presentation alternatives. In context-oriented adaptation scenarios, such decision-taking algorithms can be represented as an adaptation graph where the selection of scalability layers is driven by a concise set of rules as defined in Chapter 6 [3]. Our scene adaptation decision-taking algorithms were experimented using typical usage scenarios. For instance, adaptation parameters for STI axes have been configured for device-oriented context based on resolution *(Spatial)*, processing power *(Temporal)* and memory consumption *(Interactive)* [1]. In the summarization domain [2], adaptation parameters have been configured for user-oriented context based on region of interest *(Spatial)*, sequence of interest *(Temporal)* and action of interest *(Interactive)*. Our experiments concluded that the flexibility of our adaptation graph is sufficient to guide adaptation paths to an optimal presentation while maintaining a straightforward selection process.

7.1.3 Multimedia scene scalability

The term "scalable" and "scalability" should be understood in the *Scalable MSTI* approach as in the MPEG-4 SVC standard [104] and "refer to the removal of [video] bit-stream in order to adapt it to the various needs or preferences of end users as well as to varying terminal capabilities or network

conditions". Through our experiments, the scalability of multimedia scenes was tackled in three different but compatible ways.

First, the adaptation of multimedia presentations including scalable media constitutes a first step towards content scalability [7]. We showed within the DANAE[47] test bed that the adaptation decisions taken for scalable media can be used to steer the adaptation of multimedia scenes using adaptation updates in the MPEG-21 digital item adaptation framework. Furthermore, scene scalability, as defined by the *Scalable MSTI* model, can be combined with media scalability by either defining dependent adaptation graphs (e.g. combined scene and video resolution) or independent adaptation graphs (e.g. scene resolution and language along with video quality).

Second, the adaptation of *Scalable MSTI* scenes including non-scalable media also introduces content scalability by proposing a set of selected media composed in space and time into an interactive presentation tailored to usage environments. Such scene scalability can be combined with the scalability of graphics elements as defined in SVG [118], the scalability through level of details of 3D-graphics as defined in VRML [55] or scalable fonts as specified in the TrueType format [63]. In that case, scene properties (and mainly spatial properties) of *Scalable MSTI* documents might benefit from such plasticity without any consequences on the adaptation graph.

Third, the progressive decoding and rendering of *Scalable MSTI* scenes can make use of content scalability to provide a coherent and relevant multimedia presentation to the user during content loading. We showed within the RADIO+ project[32] test bed that the broadcasting of collapsed *STI* scalability layers over dependent MPEG-4 elementary streams can be used to optimize the trade-off between content quality and transmission delays in the T-DMB environment.

7.2 Application of work

The multimedia scene scalability concepts discussed in this dissertation have been experimented on various examples[48] and using different multimedia formats such as SVG [118], MPEG-4 BIFS [59] and SMIL [119]. Additionally, we conducted experiments based on two complete test beds as part of the DANAE[47] and the RADIO+[32] research projects.

7.2.1 MPEG-21 test bed

The adaptation concepts we developed for multimedia presentations in the scope of the DANAE project have been applied to the MPEG-21 Digital Item Adaptation framework focusing on its `AdaptationQoS` descriptor [91]. Our studies concluded into the standardization [60] of a new tool, called `ConversionLink`, which allows the adaptation of a non-scalable media according to the adaptation decisions of a master scalable content. Within the MPEG-21 test bed, we developed a scene adapter implementing our adaptation updates. This scene adapter can be driven by a scalable scene (a *Scalable MSTI* document) or a scalable media (e.g. a scalable video) as master scalable content in order to perform the adaptation of the scene along with all its media, scalable or non-scalable. Indeed, due to the abstract concept of scalability layer applicable to scene and media, the same implementation can be used to simultaneously adapt both of them in tandem.

7.2.2 T-DMB test bed

The scalable concepts we developed for multimedia presentation influenced the development of the multimedia digital radio platform conducted by RTL in the scope of the RADIO+ project. In particular, we contributed to the specification of an authoring tool (see Appendix D) which decomposes multimedia services into spatial properties that can be manipulated through a WYSIWYG interface according to user rights and temporal/interactive templates which can be configured by the user. This internal hierarchy of live content (mainly spatial) and asynchronous services (mainly interactive) can been serialized into a

[47] EU IST-1-507113 - Dynamic and distributed Adaptation of scalable multimedia coNtent in a context-Aware Environment http://danae.rd.francetelecom.com
[48] See http://www.tsi.enst.fr/mm/MSTI

generic description interfacing a multimedia content production chain with a T-DMB multiplexer (see Appendix E). We contributed to the specification of this DMB Markup Language interface by focusing on the optimization of the progressive transmission of a scalable multimedia scene over multiple streams. Experiments were conducted on an MPEG-2 TS multiplexer simulator which simulates content carrousel durations and content synchronization based on configurable multiplexing algorithms.

As far as the adaptation of digital radio services to the user's context is concerned, we participated in the specification [8][9][10][11] and the standardization [60] of new MPEG-4 BIFS tool, called `EnvironmentTest`, which allows the specification of adaptation updates as defined in the *Scalable MSTI* approach. Additionally, new mechanisms to access stored user preferences (`Storage` tool) or context information (`ReplaceFromExternalData` commands) from a digital radio device have been defined. These new features have been gathered into a new pair of MPEG-4 BIFS profiles (scene graph and graphics), called ExtendedCore2D profiles, which extend the Core2D profiles currently referenced by the T-DMB standard.

7.3 Perspectives for future work

The work presented in this dissertation can be continued in several directions ranging from the enhanced production of multimedia scalable scenes, optimizations for scene scalability, investigations about the scalable documents for television broadcast and the scalable adaptation to user preferences.

7.3.1 Scalable scene production

In terms of content production, the multimedia scene scalability features introduced in the *BIFSEdit* authoring tool have been limited to automated optimizations enabling the delivery of scalable multimedia services for a progressive playback (see Appendix D). As a consequence, the MSTI templates configured during authoring help the content creation by proposing a set of prepared multimedia canvas to the editor. Furthermore, they also ease the progressive transmission of the multimedia service by identifying the relevant pieces of information that must be conveyed with the highest priority. When considering an authoring tool for the production of *Scalable MSTI* scenes, these MSTI templates could also be used to disclose to the user the adaptation scenarios they cover. In that case, an appropriate multi-view approach for the visualization of scalable documents into an authoring tool would benefit from incremental XML transformations such as those defined by L.Villard [112].

Although the integration of spatial, temporal and interactive scene properties into authoring paradigms has already been tackled in the past, for instance by the Madeus project [79], the composition of multiple *Scalable MSTI* documents into a single scalable document needs to be further investigated. Several approaches have already been identified when composing two *Scalable MSTI* scenes. First, one of the scalable scenes might be adapted and integrated, as a non-scalable scene, into the master scene. In that case, a part of the presentation adaptability is lost during the document composition. Second, the *STI* scalability layers of both scenes may be dispatched into a new scalable scene. Such a scalable scene fusion would improve the granularity of the adaptation features of the resulting presentation but would require compatible adaptation axes. Finally, one of the scalable scenes might be 'inlined' in the master scene thus defining a scalable sub-scene. In that case, the resolution of dependent adaptation graphs has to be explored.

7.3.2 Scene scalability optimizations

Our contributions to the standardization of a new MPEG-4 BIFS profile enabling adaptation updates still need to be referenced in the T-DMB standard so that presentation adaptation through scene scalability becomes available for the digital radio market. Besides, the ongoing standardization of the carriage of MPEG-4 BIFS over DAB+ [39] by the WorldDMB forum will broaden the application field of multimedia scene scalabiliy to the entire DAB family [41]. The MPEG-4 BIFS ExtendedCore2D profile has been defined as an extension of the Core2D profile to guarantee backward compatibility with existing T-DMB receivers. However, the broadcasting of hierarchical T-DMB services, where a Core2D scene is extended by ExtendedCore2D scene elements, still needs to be specified, possibly using multiple

MPEG-4 Initial Object Descriptors in an MPEG-2 stream. So, scene scalability based on the hierarchical organization of format profiles as defined by SMIL scalable profiles [119] still needs to be evaluated.

We decided to specify an in-band description of our scalability layers as part of the scene bit stream using, for instance, the `EnvironmentTest` descriptor in the MPEG-4 BIFS format, XSL stylesheets [122] for HTML [120] or JavaScript codes [64] for SVG [118]. However, a modification of the MPEG-2 TS [54], MPEG-4 SL [56] or RTP [51] signaling layers for scene adaptation purpose would optimize the transmission of scalable scenes by enabling FEC-based unequal error protection algorithms [26], distributed content delivery [97] or unequal interleaving of the scene scalability layers [133]. Such extensions or customizations of these standards could be investigated considering a Network Abstraction Layer for multimedia scalable scenes [106].

Finally, the dynamic nature of multimedia services is taken into account in our *Scalable MSTI* approach by directly upgrading the STI scalability layers of the adaptation graph over time. In that case, the adaptation graph only needs to be recomputed starting from the previous *Spatial*, *Temporal* or *Interactive* layer that was modified. A new adaptation path might be decided and can lead to a partial modification of the current presentation starting from the nearest *Random Access* layer. However, the modification of the *Media* description of a *Scalable MSTI* document, which includes media references, requires a new rendering of the entire presentation. This limitation of our approach can also be found in MPEG-2 TS broadcasting while incrementing the PMT version number and in RTP streaming due to the static nature of SDP. Workarounds have been found for T-DMB digital radio broadcasting by repurposing MPEG-2 Elementary Streams using the MPEG-4 OD framework. However, the continuity of a multimedia service over time remains an open issue especially when the available bandwidth requires a progressive loading of the content.

7.3.3 Scalable scenes for television broadcasting

The requirements that guided our design choices in this study came from the terrestrial digital broadcasting of multimedia radio services. Although our contributions can be directly applied to television broadcasting, the scalability of multimedia scenes might be very different due a larger bandwidth and thanks to the enhanced capabilities of set-top boxes compared to handheld digital receivers. As a consequence, further investigations are needed to explore multimedia scene scalability including numerous scalability layers and possibly taking into account hybrid broadcast-broadband TV scenarios where multimedia services can be complemented through an IP-based connection. [88]

In particular, the availability of a JavaScript [64] engine in the HbbTV standard [40] and the respectable processing power capabilities of set-top boxes raise again the question of constraint-based adaptation as introduced in Chapter 4. Indeed, the layout capabilities of spatial scene properties of our *Scalable MSTI* remain limited when considering the granularity of the adaptation to the screen resolution. However, adaptation constraints often need to be expressed into a procedural format which cannot be easily handled [84]. Therefore, a combination of both approaches may constitute an optimized solution.

7.3.4 Scalable scene adaptation to user preferences

The adaptation of the multimedia presentation to user preferences was willingly moved away from our study in order to restrict the adaptation domain to the user's context. However, our *Scalable MSTI* model features digressing and forking adaptation paths that can be used to provide auxiliary adaptation parameters related to user preferences.

In particular, the *Scalable MSTI* approach could enable selection algorithms based on implicit feedback approaches [96] by defining our adaptation graph as a configurable 3D-space for preferred topics. Thus, inferred favorite media would constitute lower scalability layers (the ones with the highest priority) while new topics would require more user actions to become accessible. Additionally, this mapping could constantly evolve over time according to user decisions analyzed during content playback. Furthermore, user preferences could also be leveraged by prefetching highly scored scene scalability layers, thus optimizing content loading delays.

Chapter 8
Appendix A: Digital radio scenarios

In the following, our digital radio scenarios [12] are described into two different sections: Live Services (identified as 'S-Lx' in Section 8.1), and Additional Services (identified as S-Ax in Section 0). A limited set of generic technical requirements (TR) can be extracted from these scenarios and are described in Section 8.3. These technical requirements led most of the developments that were conducted as part of the Radio+ project and aimed at defining a unitary test suite for the enhanced production chain developed in the context of this project.

8.1 Digital Radio live scenarios

8.1.1 Program announcements (S-L1)

As soon as the visual service attached to a radio program is displayed, some additional information indicates the time progress status for the current program, together with information about the next program to come. This information is directly included in the program visual service. Hence, the user do not have to browse the program guide to find what is showing now and what will come next. The service also provides the time elapsed from the beginning of the current radio program (along with its start/end times) so that user always knows the status of current program when switching on his Digital Radio device or when having a look at the screen of his radio receiver. For instance, a progress bar (very frequently updated) keeps listeners aware of the time status of the current song (elapsed/remaining time).

Some pieces of information (e.g. SMS sent by radio listeners) can be displayed in an auto-scrolling panel included in the global service. Some shows encourage the radio listeners to send opinions or other personal messages, related to the radio program subject. This data can be sent to the radio station through an IP connection (via the radio website) or through SMS. Text messages are then filtered and included to the visual service as scrolling text. The overlay constraints induced by the inclusion of a scrolling panel into the main service are managed by the multimedia service so that the insertion of the panel does not affect the ergonomic aspects of the service.

A summary service offers a synthesis of what happened so far in the current program. This content is frequently updated, so that any user who steps in the visual service can quickly fill the gap or retrieve information about what he has just been listening to.

S-L1.1	Display of current and coming next program
S-L1.2	Progression of the current program
S-L1.3	Message area
S-L1.4	Auto-scrolling text
S-L1.5	Visual summary of the current program

8.1.2 Music programs (S-L2)

During music programs, the visual display provides contextual information about the current song's title, album and artist. For instance, the visual service displays the cover of the album (or the single) for the current song being broadcast. The picture shows up immediately when the music starts, and remains available for the duration of the audio track. Along with the album picture, textual information about the music title (e.g. the song, album and artist names) is displayed in the visual service. One or more web links are included in the visual service, offering to get further information about the current song, album or artist if an IP connection is available. The name of the next song to come can also be included in the visual service associated with the music program as a textual message.

Additionally, a VU-meter mode can be activated anytime, regardless of the program type, allows a real-time visualization of the audio signal level. As a song is being played, synchronized lyrics can also be displayed. Thanks to some underlying mechanism, the lyrics are dynamically highlighted throughout the song broadcast:

- song lyrics are presented in structured paragraphs that group text lines into verses and chorus.
- lyrics words for the current song are emphasized at the very moment when they are pronounced by the singer: highlight, color change, italics font…

S-L2.1	Music title information
S-L2.2	Artwork of the title being broadcast
S-L2.3	Textual information
S-L2.4	Web links
S-L2.5	Lyrics structure over time
S-L2.6	Highlight of synchronized lyrics
S-L2.7	Display of the title to come
S-L2.8	Temporal progression of the current title
S-L2.9	Display and update of a VU-meter

8.1.3 Talk programs (S-L3)

The “talk” service intends to offer a visual projection of the dynamics of the radio talk show. While a guest is being interviewed, relevant or humorous sentences extracted from the conversation are inserted into the visual service. Tiny pictures of the radio hosts and guests are also included in the visual service associated with the talk program. When someone speaks up, the appropriate image is synchronously highlighted. Moreover, sometimes other related pictures and humorous drawings can be inserted in the service. Additional textual information gives a whole description of the current talk program (the different debate topics, the guests’ names and so on).

S-L3.1	Display of contextual information during an interview
S-L3.2	Live citation inclusion
S-L3.3	Dynamic illustration during the show
S-L3.4	Display of various information

8.1.4 News programs (S-L4)

A rich visual service is associated with the live news programs being broadcast. This service remains permanently synchronized to the live audio contents, and has enough reactivity to cope with live incidents (last minute themes reordering; canceled or delayed guest…). The most relevant sentences or expressions extracted from the host speech are included as textual quotations in the visual service. When an additional geographic information can be important for a given news topic, then a picture can be included in order to facilitate the event localization. Headlines, various themes and guests are presented in the visual service. Those data can be refreshed very quickly to handle delays or reordering issues.

S-L4.1	Display of the headlines
S-L4.2	Insertion of citations
S-L4.3	Visualization of geographic location of news
S-L4.4	Display of textual headlines and subject of the news program

8.1.5 Entertainment programs (S-L5)

Some programs invite listeners to play games by calling the radio station. Listeners with digital radio receivers can also virtually join the game and possibly get a reward. The game interface is strongly interactive. All remote participants are given the same quiz (through the broadcast channel), each of their answer will be interpreted on the receiver and their points are stored. When the game is done, the user may have access a code that was broadcast to all but that is only accessible to players with a given amount of points. A reward can be then collected by calling the radio station and giving this code.

Throughout the game, a tight synchronization of the visual interface with the audio content makes the user feel he is really part of the game. The quiz questions show up to the remote participant by the time they are revealed to the radio listener, and both are given the same amount of time to answer the question. Radio receiver owners can be given visual hints during the game, which the 'live' participant (on the phone) does not have.

S-L5.1	Management of the logic of a game
S-L5.2	Synchronization between "on-line" and "on-air" players
S-L5.3	Visual hints

8.1.6 Commercials (S-L6)

Visual commercials are associated with the broadcasted advertisements (possibly localized). Time (short duration) and money (agreement with the advertiser) constraints require that these data be tightly synchronized with the audio stream. When a commercial audio sequence starts, the advertiser logo immediately shows up on the screen, and remains for the exact duration of the advertisement. The picture must have disappeared when the next commercial begins. Some textual elements that summarize or provide further information about the audio content of the commercial can be coupled with the advertiser logo. Web links which allow the interested listener to get further commercial information (and possibly get an access to the advertiser's website) can be included into the commercial visual service. In case local radio station can broadcast localized advertisements, the associated visual service can either display commercial content related to the localized advertisement, or content related to the main advertisement program.

The global visual service contains reserved areas dedicated to commercial contents display. The commercial content can be displayed in a dedicated banner included in the main visual service. The commercial visual content being broadcast can be activated by user interaction (by touching the dedicated banner or a focus on the commercial area). This action can also trigger the activation of an advertisement audio clip.

When an advertisement is broadcasted, a special offer can be proposed through the visual interface. This discount remains available a few minutes after the advertisement broadcast so that the user can access a ticket through interactivity. This ticket can be stored either by recording the radio program or by a dedicated image storage mechanism on the radio receiver.

During advertising campaigns, commercials sequences are created off-line and can be repeated many times. Because the purpose of advertisement is to catch the radio listener's attention, the contents of the visual service can be more sophisticated than content produced on-the-fly. On the one hand, the use of short animations can increase the visual impact of the message. On the other hand, the use of animations can be disapproved because it distracts the user and could be dangerous (e.g. in the car driving context).

S-L6.1	Advertiser's logo
S-L6.2	Textual elements of commercials
S-L6.3	Animated commercials

S-L6.4	Local commercials
S-L6.5	Web link to advertisers
S-L6.6	Storing special offers
S-L6.7	Insertion of commercials content in the main service
S-L6.8	Triggering advertisement on user's action
S-L6.9	Commercial audio clip

8.2 Digital Radio asynchronous services scenarios

Additional multimedia services that complement services synchronized with the audio program can be proposed to radio listeners. These additional services are accessible to the listener either through interactive areas on the radio receivers screen (dedicated virtual buttons or links) or through interactive keys (keypad Up-Down-Left-Right-OK).

8.2.1 Weather forecasts service (S-A1)

A daily weather forecast is proposed to auditors. This service illustrates weather previsions with small pictures (pictograms) along with more precise figures (e.g. temperature, strength of the wind) for some big cities in France. Furthermore, weather forecasts are presented to users according to their location by providing a direct access to previsions for their own area. The weather service proposes forecasts for several city/region of France. The user can have access a complete weather service through interactivity that is presented in accordance with the reception area or selected based on a user profile. For instance, local weather forecasts are automatically displayed when accessing the weather service whereas the remaining parts of the weather information (e.g. longer term forecast or other regions) requires more interactions from the user to be become visible.

Weather forecasts are illustrated with a limited set of small images that give an overview of the previsions (e.g. sun, rain, fog, snow). Weather forecasts can be visualized as a general text comment for the whole day and some figures such as minimal/maximal temperatures, wind strength, air quality, etc. Images and textual data are typically updated every 12 hours in the multimedia service. During the on-air presentation of the weather forecast by the radio presenter, visual components of the weather forecasts are dynamically associated with the comments of the journalist (possibly thanks to animations). These visual elements can be images (pictograms, maps of France) that are already available in the interactive weather forecast service. When a weather forecast alert is triggered by Météo France, the visual weather service is updated by including high-priority information: storms, floods, important snow falls… This information is included in the existing weather forecast in such a way that the user clearly identifies high-priority alert information and usual weather forecasts.

S-A1.1	Weather forecasts pictograms
S-A1.2	Textual information of a weather forecast
S-A1.3	Prioritized display of local weather forecast
S-A1.4	Live composition of weather forecast information
S-A1.5	Weather forecast alerts
S-A1.6	Listening to the latest weather forecast report

8.2.2 Stock exchange service (S-A2)

A service that dynamically describes worldwide stock exchange information is proposed to auditors. The stock exchange service displays structured figures that drive the stock exchange news: graphics illustrating the key quotes of the day or tables summarizing the main changes in the market (up and down). These are coupled with comments and advices.

During the day, users that are interested in the stock quotes can follow the stock markets thanks to a service that is frequently updated (every 30 minutes). The regular update of the figures guarantees the user with up-to-date information. Analysis are provided by stock exchange experts and proposed to auditors on a daily basis.

S-A2.1	Display of stock markets information
S-A2.2	Comments on the stock markets
S-A2.3	Regular updates of stock market information
S-A2.4	Listening to the latest stock exchange report

8.2.3 Astrology service (S-A3)

Every day, the user can have access this horoscope from its multimedia radio receiver. The service offers a textual forecast to each sign of the zodiac in several fields (love, health, work, money for instance) and also a comparison with other signs (e.g. sign of the day). Every day, a sign of the zodiac is described as the "sign of the day" because it is the luckiest for instance and therefore have access to more detailed information. It can also be displayed first when accessing the astrology service while it is still possible to have access to other signs through interactivity. The interface of the astrology service features a simple mechanism to configure a favorite sign of the zodiac. Every time the user has access the horoscope, the presentation is then automatically organized in such a way that the favorite sign is always displayed first.

Every sign of the zodiac has a graphical representation that can have several image resolutions. For instance, it could be large for the heading of each forecast and small when displaying the current sign among others. The astrology content has several indices such as stars that indicate a positive day (depending of their number) or pictograms whose expressions (e.g. sad, happy, neutral) give an overview of the coming day. These images are updated according to the availability of an astrology forecast: every day or every week.

A numerology service proposes to listeners to give their date of birth as input (08/07/2009 for instance) and then provides a personalized numerology forecast based on theses figures.

S-A3.1	Astrology pictograms
S-A3.2	The detailed horoscope of the day
S-A3.3	The sign of the zodiac of the day
S-A3.4	Selection of the preferred sign of the zodiac
S-A3.5	Numerology
S-A3.6	Listening to the latest astrology report

8.2.4 Traffic service (S-A4)

The traffic service is essentially composed of images: snapshots of the traffic for instance. These data are frequently updated so as to keep the user informed about the current status of the traffic (car traffic but also public transport traffic). Traffic information related to the location of the user is always proposed first. In radio broadcast, the knowledge of the service coverage makes it possible to adapt the content

presentation so that the user has first access to the traffic information in accordance with its location. In case of an accident, unexpected traffic jams or unreported roadwork, visual traffic alerts are integrated in the initial traffic service with several warnings depending on their priority.

Traffic information is illustrated by images that can be of various types: pictures of the Parisian ring road, snapshots of the main road illustrated with traffic indices… While driving, these images allow users to quickly visualize the actual traffic on their usual way to work. This visual information is updated frequently (every hour and every 15 minutes during rush hours) so that users can always have access to an updated status of the current traffic. Some comments and traffic previsions (such as "A4→A86 : 25 min" or "Porte St Cloud : 3km traffic jam") are provided along with the visual traffic element and are regularly updated (every 30 minutes). For auditors who are listening to the radio in public transportation, disturbances of the railway and bus traffic are visually notified in the traffic service. During the on-air traffic information presented during rush hours in the morning and in the evening, the visual elements that are provided in the continuous traffic service are gathered in the main service area along with textual information and synchronized with the audio. During rush hours (morning/evening, week-end departure), the visual content of the service traffic, is frequently updated (every 15 minutes), so that users that are very cautious about traffic disturbances can take decisions according to the actual traffic status.

S-A4.1	Display of the traffic status
S-A4.2	Textual description of the traffic status
S-A4.3	Prioritized display of local traffic information
S-A4.4	Live composition of traffic information
S-A4.5	Regular updates of traffic information
S-A4.6	Traffic alerts
S-A4.7	Listening to the latest traffic report

8.2.5 News service (S-A5)

A news service provides a visual representation of daily headlines (global or local news). The news content is divided into several topics (general news, sports, politics, entertainment events…). The layout of the news service (e.g. topics display order) can be dynamically modified and defined as preference by the user so that they are automatically restored when coming back to the service. The news visual content is composed of a bunch of headlines complemented with summaries of newspaper articles, possibly provided along with pertinent analysis and comments on the subjects. Textual interviews extract or quotations can also be included. When local news content is available for a given receiving area, it is only broadcast to the targeted area. Visual data that are available for the asynchronous news service usually comes from newspaper or websites articles. These articles are also used by the radio journalists to build up the daily news program. The radio presenters can re-use this data to synchronously illustrate the live news.

Numerical data (possibly structured) can be provided along with the textual description of news (for instance, a pie-chart diagram that illustrates the employment distribution in France). Some pieces of news are illustrated with pictures, whose aim is to increase their attractiveness. Such figures and visual illustrations are provided with a limited lifetime depending on the amount of incoming news content (2 to 10 hours lifetime). In case of news alerts (e.g. a kidnapping alert), a specific visual content is generated and preempts the currently displayed news service, regardless of its status. A large number of topics are permanently available to the user (typically from 5 to 20 items). The user can decide to sort the display order of the news by setting layout priorities through the interface. The preferred topics will then become more accessible and visible. If the user quits the service and then comes back in, his preferences are restored.

S-A5.1	News based on figures
S-A5.2	Visual illustration of news
S-A5.3	Textual news
S-A5.4	Local news
S-A5.5	Live composition of news elements
S-A5.6	News alert
S-A5.7	News customization
S-A5.8	Listening to the latest news report

8.2.6 Program guide (S-A6)

The scheduling and description of all the programs being broadcast is synthesized in a "program guide", whose content might vary from one receiving area to another (in case of localized radio programs). The layout of the program guide depends on the currently broadcasted program. When the user accesses the program guide service, visual data (name of the show, description, radio host picture) related to the current show is immediately displayed, along with some "coming next" information. However the user is free to navigate through the whole guide. In case localized program data are available within a given receiving area, the program guide is adapted in order to include the specific contents. This operation is done before the program guide is broadcasted to this specific area. A complete program guide service mixes several standalone program guides from different radio stations. This generic program guide enables the scheduled recording of several programs from different radio stations, through a single interface.

Each program item in the guide is tagged with the picture of the radio personality that presents the program. Those images are seldom updated (once a trimester or once a year). A short description is associated with each program in the guide. This description is typically composed of the program's name, its duration and the name of the radio hosts.

S-A6.1	Pictures of radio animators
S-A6.2	Textual information of the program guide
S-A6.3	Automatic positioning in the program guide
S-A6.4	Local program switching
S-A6.5	Cross-radio program guide

8.3 Digital Radio use cases

The various digital radio scenarios for live content and additional services lead to a set of technical requirements which are summarized in this Table 10:. This abstraction for digital radio usages identifies the main technical mechanisms that are required to guarantee the successful deployment of the envisioned services. Technical requirements have been divided into height categories, identified as TRx.

Table 10: Digitial radio technical requirements for multimedia sevices.

	Code	Description
Images insertion / update (signalling & transport)	TR 1.0	Asynchronous image insertion
	TR 1.1	Asynchronous image update
	TR 1.2	High synchronization of image insertion/update (40 ms)
	TR 1.3	Low synchronization of image insertion/update (1 to 2 s)
Scene elements Insertion / update (signalling & transport)	TR 2.0	Asynchronous scene element insertion
	TR 2.1	Asynchronous scene element update
	TR 2.2	High synchronization of scene element insertion/update
	TR 2.3	Low synchronization of scene element insertion/update
Multimedia data insertion in the scene	TR 3.0	Multimedia stream contribution to the local transmitter and insertion into the digital radio service
	TR 3.1	Local multimedia production and local insertion into the digital radio service
	TR 3.2	Integration of multimedia data from several sources
Live data insertion	TR 4.0	Immediate insertion of multimedia data with a higher priority than originally planned data
	TR 4.1	Synchronized composition of multimedia data coming from existing multimedia streams
	TR 4.2	Dynamic insertion of multimedia data produced on-the-fly and synchronized with audio stream
	TR 4.3	Presentation of synchronized of live multimedia data
Dynamic data update	TR 5.0	Update of multimedia data (highly) dynamic and not related to the audio stream
	TR 5.1	Update of multimedia data (highly) dynamic and related to the audio stream
Multi-channel audio	TR 6.0	Management of the volume of simultaneous audio streams
	TR 6.1	Management of the selection of multiples audio streams
Scene logic	TR 7.0	Insertion / update of highly-structured multimedia data
	TR 7.1	Generation of advanced interactive multimedia data
	TR 7.2	Recording and restoring or user preferences
	TR 7.3	Access to receiver information
Connected mode	TR 8.0	Multimedia data push on a return channel.
	TR 8.1	Multimedia data push to the receiver.

Chapter 9
APPENDIX B: DMB Radio

Digital Multimedia Broadcasting (DMB) for radio, also called DMB Radio [38], is part of Eureka 147 digital family of standards and therefore relies on the same network layer (e.g. packet mode statistical multiplexing) and the same physical layer (e.g. error-correction coding and OFDM modulation) as the Digital Audio Broadcasting (DAB/DAB+) standards [41][39]. In a few words, DMB Radio differs from DAB/DAB+ in its presentation layer that is based on the MPEG-4 multimedia framework designed for the synchronized playback of audio and advanced multimedia content. Within the Terrestrial (T)-DMB specification, two application fields are identified: mobile television and interactive radio.

9.1 Overview

Digital Audio Broadcasting (DAB) is a technical evolution of traditional FM radio. The advantages of digital radio are numerous: it offers better population coverage, an increase in the number of radio services, provides a higher audio quality and allows for better integration in digital devices with recording capabilities (e.g. MP3/MP4 players). DMB Radio combines the traditional DAB Eureka 147 proven technology with the latest high-efficiency MPEG-4 audio codec [57]. However, improving FM radio programs on an audio-only basis is not a sufficient breakthrough to convince listeners to move from analogue to digital.

In the current environment radio broadcasters must cope with new challenges from both competitors and at the same time complementary digital radio technologies such as web radios accessible on WiFi-enabled devices (e.g. kitchen radios) or visual radio streamed on mobile phones through 3G/4G networks. Digital radio offers these broadcasters a unique proposition to upgrade into the digital age through visualization and interactivity. In particular, digital radio allows for the delivery of data components along with audio services. These data services can be synchronized with the audio stream such as *Dynamic Label Segment* (DLS) carried in *Programme Associated Data* (PAD) or may not be synchronized with an audio stream such as *Electronic Programme Guide* (EPG) or *Transport Protocol Experts Group* (TPEG) information carried as *Non Programme Associated Data* (NPAD). The relationship between Eureka 147 DAB multiplex, DMB Radio data and other digital radio technologies such as DAB+ is illustrated in Figure 72.

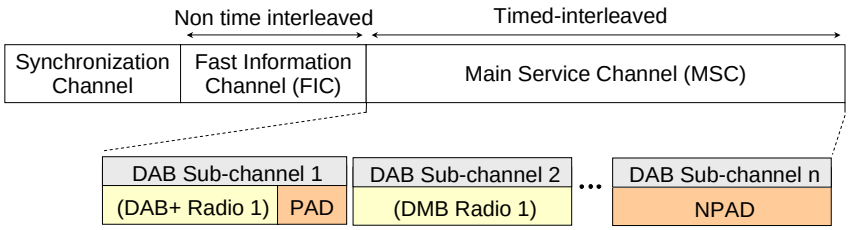


Figure 72: Structure of a DAB/DAB+/DMB transmission frame.

At a higher level, two types of auxiliary data can be distinguished: structured data and designed data. Structured information allows advanced data processing on the receiver side, such as scheduled recordings based on EPG data or geo-localized traffic information filtered from TPEG data streams. Such data is usually described using the XML format [116], which is based on schemes that ensure a common understanding of data semantics. Contrary to structured data, designed information provides data that is ready to be displayed according to the broadcasters' design. Examples are text information targeting two-line alphanumeric displays with DLS data, image-based slideshow defined in SlideShow (SLS) or interactive web-like information defined through Broadcast WebSite (BWS) HTML subset.

In order to address the full range of multimedia digital radio scenarios with one single broadcast-oriented technology DMB Radio relies on structured data carried in NPAD and offers advanced designed data synchronized with the audio stream and integrated in a single framework (the MPEG-4 framework).

Therefore, DMB Radio allows broadcasters to take full advantage of the multimedia capabilities of these new receivers by relying on a generic presentation engine: the MPEG-4 BIFS engine. More importantly, the MPEG-4 multimedia framework enables a tight synchronization of the audio stream with all multimedia data, which is a key feature given the live nature of radio.

9.2 The MPEG-4 multimedia framework

DMB Radio services are designed for an advanced multimedia experience. Therefore, the T-DMB specification entirely relies on the multimedia framework defined in the MPEG-4 Systems standard [56] to describe a visual interface that combines audio, video, image, text and graphic elements into an interactive and animated presentation over a period of time. The two fundamental components of the MPEG-4 multimedia framework are the MPEG-4 Object Descriptor (OD) and the MPEG-4 Scene Description also known as BInary Format for Scenes (BIFS) in its coded representation.

9.2.1 The MPEG-4 Object Descriptors

In broadcast environments, a multimedia service is composed of multiple media streams, called Elementary Streams (ES). These Elementary Streams can be audio/video media streams whose data is continuously updated over time. They can also be static media such as images that become streams in the broadcast environment because data is repeated over time to guarantee a random access to the multimedia service. In the MPEG-4 framework, all of these Elementary Streams are identified through an external description that is called an Object Descriptor (OD). An OD contains one or more Elementary Stream Descriptions (ESD) that define configuration parameters for each ES: decoding configuration (DecoderConfigDescriptor) and timing configuration (SLConfigDescriptor) for instance. All in all, MPEG-4 ODs create an object abstraction from media streams that can then be handled through identifiers (OD_ID) in the MPEG-4 framework. An example of Object Descriptor is given in Figure 73.

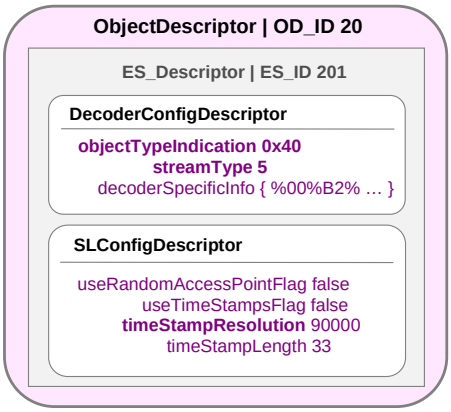


Figure 73: An MPEG-4 Object Descriptor example.

9.2.2 The MPEG-4 Scene Description

Multimedia content composed with several media elements needs a choreographer in order to build a complete presentation. This is the purpose of the MPEG-4 Scene Description that is responsible for grouping the various MPEG-4 Objects into a single content, describing a layout for them, introducing styling details, defining interactive behaviors and possible animations. In fact, an MPEG-4 Scene should be seen as a hierarchical tree that defines structural branches (describing 2D + ½ positioning, scaling, visual activation), media leafs (graphic elements, text paragraphs) with styling properties, timing and interactive leafs and additional leafs referencing media Objects through OD identifiers (OD_ID). An example of MPEG-4 Scene Description is provided in Figure 74. MPEG-4 Scenes can be described in XML-based languages such as the eXtensible MPEG-4 Textual (XMT-A) and encoded in its binary format BIFS.

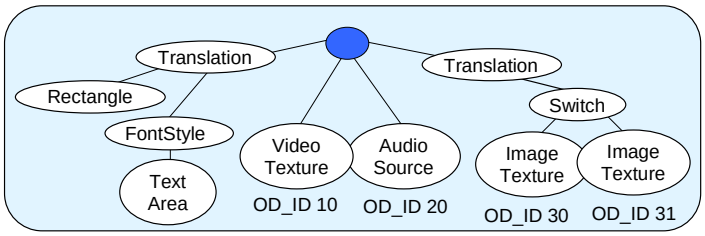


Figure 74: An MPEG-4 Scene Description example.

9.2.3 The MPEG-4 Framework in DMB Radio

The Scene Description (BIFS) and Object Descriptors (OD) are the two fundamental components of the MPEG-4 multimedia framework because they both determine the media that are part of the multimedia content and how they should be composed into an interactive and animated presentation. The MPEG-4 framework additionally defines an update mechanism that is used to insert, replace or delete media Objects or parts of the MPEG-4 Scene: OD Updates or BIFS Updates. This mechanism is essential in DMB Radio because it enables the continuous evolution of the presentation over time as illustrated in Figure 75.

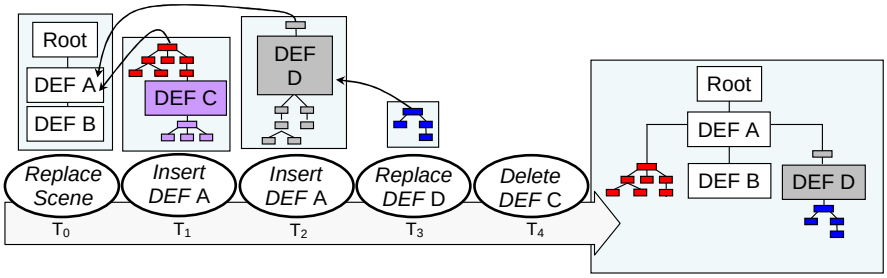


Figure 75: The MPEG-4 Scene Update mechanism.

These Updates are timed as the Access Units (AU) of traditional media streams and therefore also make up Elementary Streams. Since BIFS and OD Elementary Streams are essential for a radio receiver to access the MPEG-4 service, the two related Object Descriptors are described in a specific placeholder, called the Initial Object Descriptor (IOD), which is the entry point to the MPEG-4 framework in DMB Radio. The relationship between the IOD, BIFS and OD Elementary Streams, OD Updates, BIFS Updates and media Elementary Streams is illustrated in Figure 76.

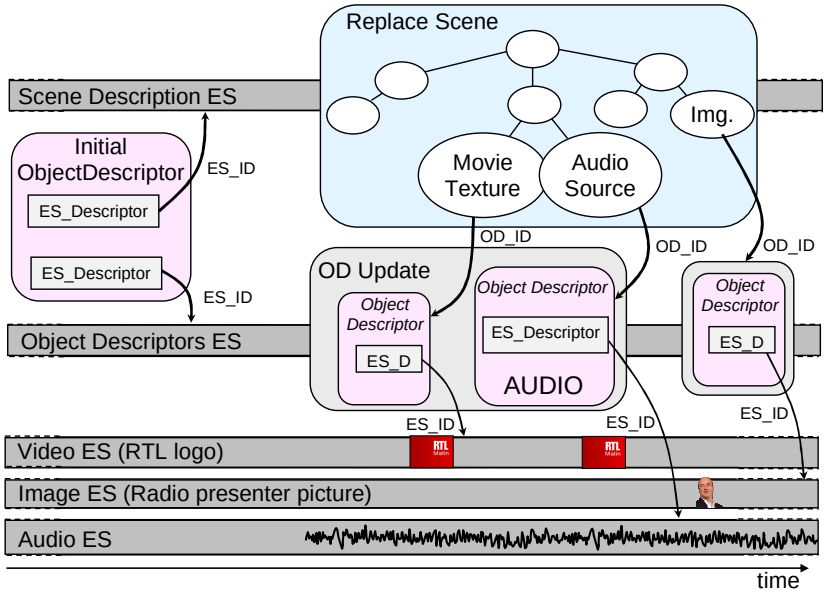


Figure 76: The MPEG-4 Framework in DMB Radio.

9.3 MPEG-4 broadcasting over MPEG-2 TS

DMB Radio is dedicated to radio broadcasting use cases. As a consequence, the audio component of a DMB Radio service is essential: most of the DAB sub-channel bandwidth is allocated to audio data. The performances of the MPEG-4 HE-AAC v2 codec used in DMB Radio are detailed in [87] and can also be complemented with MPEG Surround [69] data to achieve an advanced sound experience. Reed Solomon (RS) error-correction coding is applied to audio Access Unit to extend the signal protection applied at the Eureka-147 DAB level. Still, the DMB Radio architecture is clearly driven by the need for multimedia enhancements to enrich the audio component. Therefore, the MPEG-4 framework, which is the core of DMB Radio, ensures a dynamic trade-off between audio quality and associated data quantity.

All high-efficiency audio compression techniques are source-driven coding and generate Variable Bit Rate (VBR) audio steams. Indeed, audio source analysis is critical to achieve the best trade-off between audio quality and optimal bit rate at each moment. Therefore, the audio bit rate of a radio program (in kbits per second) is always an average bit rate. Additionally, the optimal configuration of an audio coding system will vary based on quality expectations, available bandwidth and envisaged sounds types for a given radio station: speech, classical music, pop music… One option for a radio broadcaster is to define a single coding configuration that it is not optimal in all cases but fits all use cases. A more efficient option consists in selecting several optimal coding configurations for a set of limited use cases that are activated depending on the on-air radio show.

Reducing the audio bit rate to an optimal value does not imply any savings from a broadcaster's point of view since radio licensees usually have access to a fixed bandwidth whether they use it or not. In practice, an optimal audio bit rate is the opportunity for a broadcaster to provide additional visual services that exploit this remaining bandwidth. Within a Eureka 147 DAB multiplex, dynamic DAB sub-channel reconfiguration is possible to take advantage of a significant reduction of the audio bit rate, but this impacts the whole DAB multiplex which usually cannot be directly addressed by radio broadcasters and would also require a synchronized extension of some NPAD data bit rate (by 8kbps steps) in the Main Service Channel (MSC). In practice, only a dynamic bit rate allocation strategy at the DAB sub-channel level is feasible. For that purpose, Fixed and eXtended Programme Associated Data (F-PAD and X-PAD) can be used for text information (DLS) and image-based slideshow (SLS) respectively. DMB Radio leverages this audio bit rate variation by introducing a generic multiplexing based on the well-known MPEG-2 Transport Stream (TS) [54] at the DAB sub-channel level as illustrated in Figure 77. This multiplexing allows sharing a fixed DAB sub-channel bandwidth between the audio stream and all other data services synchronized with the audio: text information (DLS), video content (H.264), images (JPEG or PNG) and graphics, text, animation or interactive elements (MPEG-4 BIFS).

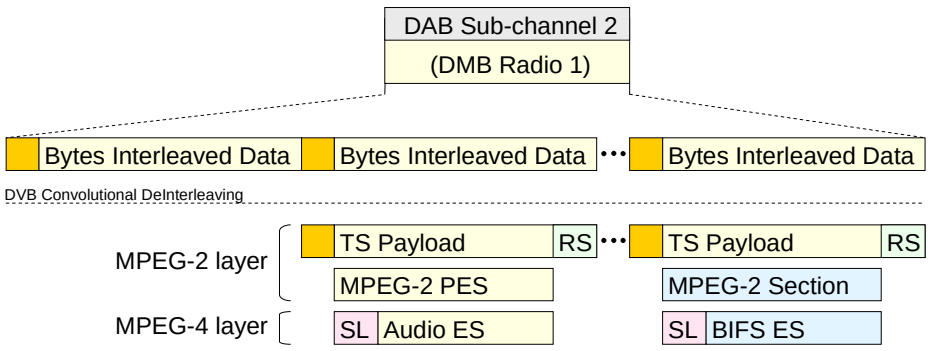


Figure 77: The Elementary Steam encapsulation in DMB Radio.

From a technical point of view, the DMB Radio data that is carried inside a DAB sub-channel is therefore a Single Program Transport Stream (SPTS). It therefore includes a Program Association Table (PAT) and a Program Map Table (PMT) that signal all available Elementary Streams and identify them with Program IDentifiers (PID). The IOD introduced in Section 9.2.3 is carried by the PMT and is used to identify the PID of the initial BIFS and OD ES among all other streams.

Since dynamic bit rate allocation require a clear control from the radio station, SPTS can therefore be generated in-house and then transmitted to a broadcasting site using traditional Digital Television MPEG-2 TS transmission mechanisms based on the Real-time Transport Protocol (RTP) and Forward Error Coding (FEC).

9.3.1 Elementary Stream synchronization

The essence of FM radio is the live creativity of radio presenters who establish this specific feeling of proximity with listeners. The challenge raised by digital radio is therefore to define a visual interface that illustrates this live reactivity through highly-synchronized associated data. DMB Radio relies on the MPEG-4 Synchronization Layer (SL) to guarantee the synchronization between the audio Elementary Stream and the MPEG-4 Scene by defining Composition TimeStamps (CTS) and possibly Decoding TimeStamps (DTS). These SL timestamps are carried in MPEG-2 TS packet as illustrated in Figure 77 (PES or MPEG-2 sections) and configured in the *SLConfigDescriptor* introduced in Section 9.2.1.

Additionally, even if professional solutions are stable and timed with accurate clocks, clock drifts between the encoder and decoders are inevitable. Therefore, MPEG-4 Object Clock References (OCR) are regularly transmitted inside SL packet headers (at least every 700 ms in DMB Radio) so that a clock drift can be controlled on the receiver side. These MPEG-4 OCRs are entirely based on the MPEG-2 *Program Clock Reference* (PCR) which is carried by a single *Elementary Stream* in the DMB Radio service (at least every 100 ms) and tuned according to the Presentation TimeStamp (PTS) defined in the PES header of OCR-enabled MPEG-2 TS packets.

As a consequence, DMB Radio is designed to guarantee the fine synchronization between the audio stream and all multimedia services. In particular, DMB Radio takes into account low-bitrate data scenarios (e.g. 8kbps) where an image needs to be transmitted beforehand (for example, 10s for a 8kbyte image) so that it may be displayed when the radio presenter makes an action like fading-in a programme item. This is achieved through a simple BIFS Update that only requires at maximum 204 bytes (e.g. one protected TS packet). Of course, the same mechanism can be used to trigger animations (during jingles or when changing from one CD cover to another) or to activate large tables displaying highly-structured and interactive textual values (complete list of soccer live scores).
On the receiver side, defining DMB Radio services as a SPTS allows a direct storage of audio and synchronized multimedia services into the straightforward TS file format (*.ts) that is also used in Digital Television and based on DVB-T [42] and DVB-H [44]. Furthermore, DMB Radio TS files can also automatically be converted into the well-known MP4 file format based on the MPEG-4 mapping of Elementary Stream defined in [61]. In that case, the MPEG-4 HE-AAC audio and H.264 streams are decoded by any mobile or PC multimedia player using QuickTime, Windows Media or VLC players whereas more dedicated devices (e.g. on radio devices) will also handle the MPEG-4 BIFS et OD streams such as in the GPAC open-source platform [81].

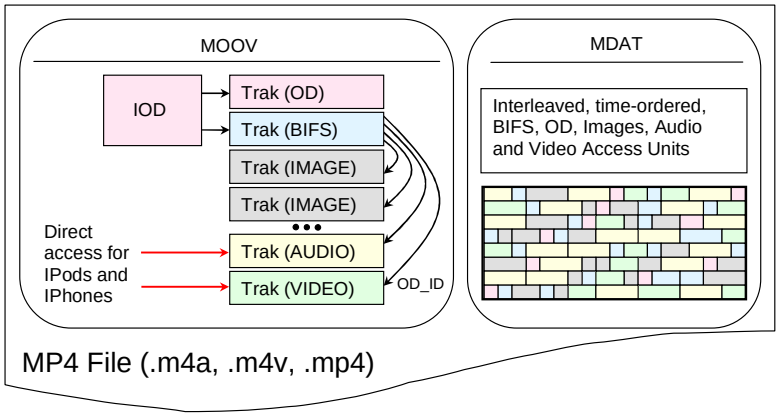


Figure 78: The ISO File format for DMB Radio services.

9.3.2 Elementary Stream carrouseling

Multimedia broadcasting is technically completely different from Unicast communication for two main reasons. First, no communication setup can be performed by the digital radio receiver thanks to an initial content request. Second, the communication channel can suffer from disruptions, which the radio receiver is unable to recover. These broadcast constraints must be overcome when coding media sources such as audio and video, which coded streams must provide regular Random Access Point (RAP) for radio receivers initialization or error-recovery. For these reasons, media Access Units (AU) are short in DMB Radio: about 20ms for audio AUs (1024 samples per frame at 48kHz sampling rate[49]) and no more than 2s for video AUs (one H.264 IDR at least every 2s). Other DMB Radio Elementary Streams such as JPEG or PNG images, BIFS, OD but also MPEG-2 signalling tables (PAT and PMT) and DLS must cope with the same broadcast constraints. The main strategy deployed in DMB Radio at the service level to improve robustness and reduce the user access time to the multimedia content is the introduction of RAP through the intentional repetition of multimedia data.

In DMB Radio, two types of Elementary Streams can be distinguished: natural media, which are continuous streams (audio, video and an image continuously repeated) and others that can be meta-data (e.g. PAT, PMT, OD) or synthesized media (e.g. BIFS). Hence, DMB Radio MPEG-2 TS packets can either contain PES packets (audio, video or image) or MPEG-2 sections (PAT, PMT, OD and BIFS[50]) as illustrated in Figure 77. In fact, MPEG-2 sections offer several advantages compared to PES packets and include: a 32-bit Cyclic Redundancy Check (CRC) that guarantees the exact correctness of carried metadata, they manage data fragmentation into several TS packets at the MPEG-2 level and they also notify updates of carried data. This update notification is managed through an incremental version number that DMB Radio receivers can use to distinguish data repetitions from data updates when listening to an Elementary Stream. This mechanism (also called carrouseling) is illustrated in Figure 79 for the PAT, the PMT, one BIFS and one OD ES that are broadcasted with a carousel period that must not exceed 500 ms for a satisfactory time access to the service.

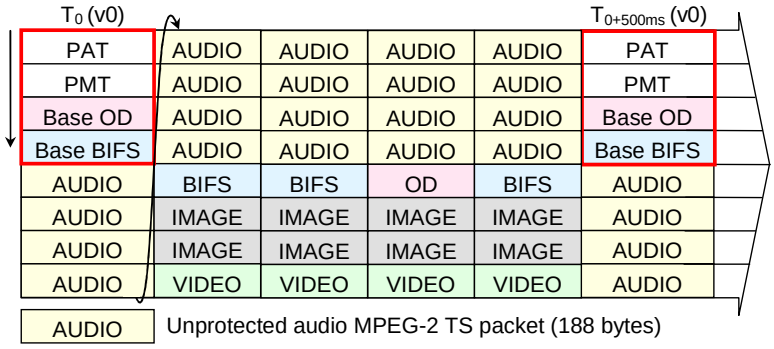


Figure 79: Example of a 128 kbps DMB Radio sub-channel.

The cyclic transmission of auxiliary objects can also be performed through the Multimedia Object Transfer (MOT) protocol [43] but in that case, there is no generic management for this data and the bandwidth cannot be easily optimized according to audio bit rate.

9.4 T-DMB bandwidth requirements

DMB Radio service configuration is signalled inside the DAB sub-channel by the MPEG-2 Program Association Table (PAT) and the MPEG-2 Program Map Table (PMT) introduced in Section 9.3.1 and by the underlying MPEG-4 Object Descriptor (OD) introduced in Section 9.2.1. Therefore, a broadcaster has complete control over its DMB service configuration at the MPEG-2 level and transmits a complete MPEG-2 Transport Stream (TS) to the broadcasting site for Reed Solomon (RS) error-protection coding and Eureka-147 multiplexing. The MPEG-2 multiplexing structure of DMB Radio allows broadcasters

[49] When Spectral Band Replication (SBR) is used, 2048 samples per frame are used.
[50] T-DMB also gives provision for the transport of BIFS data inside PES packets for BIFS-Anim streams but only BIFS-Command streams are currently envisaged for DMB Radio.

flexibility on the allocated bandwidth to each media Elementary Stream (Audio, BIFS, Images and Video data) of its DAB sub-channel within a Eureka-147 ensemble.

Table 1 introduces possible DMB Radio service configurations that can be selected over time depending on the programme for a typical DAB sub-channel of 128 kbps[51]. In this table, it can be noticed that Dynamic Label Signalling (DLS) data is provided in any configuration because it is essential for Profile-1 Digital Radio receivers that do not have a colour graphic display but only an alphanumeric display according to the EBU recommendations [37]. In the following, the different DMB Radio service configurations are further detailed according to several audio bit rates: high (Section 9.4.1), medium (Section 9.4.2) and optimized (Section 9.4.3).

Table 11: Typical DMB Radio service configurations.

	Audio	BIFS	Images	Video	PAT/PMT OD/DLS	RS coding
H1	96 kbps*	6 kbps	7 kbps	0 kbps	9 kbps	10 kbps
H2	96 kbps*	3 kbps	2 kbps	5 kbps	9 kbps	10 kbps
M1	71 kbps*	12 kbps	23 kbps	0 kbps	12 kbps	10 kbps
M2	71 kbps*	10 kbps	20 kbps	5 kbps	12 kbps	10 kbps
M3	71 kbps*	6 kbps	14 kbps	15 kbps	12 kbps	10 kbps
O1	38 kbps*	15 kbps	53 kbps	0 kbps	12 kbps	10 kbps
O2	38 kbps*	13 kbps	50 kbps	5 kbps	12 kbps	10 kbps
O3	38 kbps*	12 kbps	41 kbps	15 kbps	12 kbps	10 kbps
O4	38 kbps*	13 kbps	26 kbps	30 kbps	12 kbps	10 kbps
96 kbps* of DMB Radio data carries 88 kbps of MPEG-4 HE-AAC audio data, 71 kbps* correspond to 64 kbps audio and 38 kbps* correspond to 32 kbps audio.						

9.4.1 High audio bitrates

A high audio bit rate can be configured for radio programs that broadcast audio contents that are very challenging in terms of compression, even for high-efficiency audio coding techniques such as MPEG-4 AAC. Additionally, a radio program may also have an outstanding audio quality on common audio content because this specific program does not require a large multimedia bandwidth or because an MPEG Surround audio extension is preferred to advanced multimedia enhancements. In both cases, most of the audio bit rate would be allocated to the audio Elementary Stream. Two low-bitrates multimedia configurations (H1 and H2) can then be envisaged:

1) First, the DMB Radio service (H1) focuses on interactive content and proposes BIFS-only content with very few and light images. In that case, no video component is defined and the multimedia bandwidth is shared between highly compressed BIFS data (text and graphics elements) and low-resolution images that are either accessible through interactivity or displayed as a BIFS slideshow with a long period between images.

2) Second, the DMB Radio service (H2) favours content sharing by broadcasting a very-low-bitrate video component with light images (more precisely light IDR NAL Units in the scope of the MPEG-4 AVC/H.264 standard [58] used in DMB Radio). Such a DMB Radio service can then be imported in any usual storage format available on Personal Multimedia Player (PMP) such as the MP4 file format available on Apple's IPods and IPhones. In such a configuration, the BIFS component can be restricted to its minimal requirements to achieve a minimal bit rate: a simple visual layout with no interactivity.

[51] Typical DMB Radio service configurations are for instance 96 kbps (12 DMB Radio channels), 128 kbps (9 DMB Radio channels) or 160 kbps (7 DMB Radio channels) in a Eureka-147 ensemble.

9.4.2 Medium audio bitrates

A medium audio bit rate can be configured for most radio programs of a broadcaster because it provides a good trade-off between audio source diversity (music and talk) and audio quality using MPEG-4 HE-AAC. This medium audio bit rate can be defined based on general broadcast expectations but can also be configured dynamically according to the needs of a specific program or of a commercial break. This medium audio bit rate typically ranges from 56 kbps to 72 kbps. As a consequence, an audio bit rate of 64 kbps has been selected as an example in Table 1. Three multimedia configurations can then be envisaged (M1, M2 and M3):

1) First, the DMB Radio service (M1) leverages all BIFS functionalities by providing an interactive service that provides a high-quality visual enhancement of the current audio programs based on a large set of images (about 15 images with various resolutions), text paragraphs and animated graphic elements. This visual content is progressively retrieved and displayed by the radio receiver so that the user have access to a visual service in less than one second and can interact with a complete multimedia service in less than one minute after switching on a device. In this configuration, no bandwidth is allocated to a video component.

2) Second, the DMB Radio service (M2) has less or lighter images than the BIFS service of M1 configuration in order to allocate the released bandwidth to a very-low-bitrate video component similar to the one of H2 configuration. In such a configuration, it should be noted that the visual service based on BIFS and the video component are two alternatives than do not aim to be combined.

3) Third, the DMB Radio service (M3) aims at combining a BIFS interactive service with a low-bitrate video slideshow that dynamically displays QVGA images in the background (and not only light images as for the very-low-bitrate video component). In that case, profile-2 radio receivers [37] would have access to a complete interactive BIFS service while profile-3 radio receivers would additionally benefit from additional visual enhancements for live information.

9.4.3 Optimized audio bitrates

An optimized audio bit rate can also be configured for some radio programs whose audio source can be highly compressed using MPEG-4 HE-AAC whilst maintaining a very good audio quality (voice-only audio source in radio talk or news shows for instance). In that case, low bit rates can be configured for the audio Elementary Stream (e.g. 32 kbps as selected in Table 11) and a large bandwidth is then available for multimedia enhancements. Four configurations are envisaged in Table 11:

1) First, the DMB Radio service (O1) proposes multimedia details that deepen audio information and a large set of permanent multimedia services accessible through interactivity: weather forecasts, astrology, traffic information, sport results, music charts, box office… In this configuration, the video component can be omitted to allocate all multimedia bandwidth to interactive services.

2) Second, the DMB Radio service (O2) has less or lighter images than the BIFS service of O1 configuration in order to allocate the released bandwidth to a very-low-bitrate video component similar to the one of H2 configuration.

3) Third, the DMB Radio service configuration (O3) is similar to M3 configuration with a richer BIFS and image content: the video component displays live information and the BIFS component provides additional information in overlay.

4) Forth, the DMB Radio service configuration (O4) allocates a significant bit rate to the video component in order to enable low-resolution and low-motion video content (e.g. 12 fps in 160x120 resolution). In that case, BIFS and Image content can be temporarily reduced to release enough bandwidth for a short video sequence (e.g. football goal). This configuration can also be selected for a complete radio show in order to broadcast the live video capture of a talk show for instance.

9.4.4 Multimedia bitrate optimizations

Although the carrouseling of DMB Radio multimedia data (i.e. DLS, images, BIFS and OD) seems to be the ultimate solution to broadcast constraints, it would require a huge bandwidth to broadcast quite simple multimedia services with a satisfactory access time. Let's take the example of a simple 320x240 pixel image-based service including one additional service accessible through interactivity (e.g. weather forecast service). The worst case in terms of quality of service would be to carry two 320x240 images that would require about 40 kbytes with the usual JPEG compression factor of 90%. On the one hand, if the expected visual switching time for a radio auditor is estimated to a maximum of 2 seconds, as in DMB Radio, these two images would then require a bandwidth of 160 kbps. Such a bandwidth roughly represents about 1/7 of the resource of the whole DAB multiplex! On the other hand, if the listener expectations when switching from one radio program to another are ignored to address a typical bandwidth of 16 kbps for multimedia data, it then requires a waiting time of 20 seconds to have access to visual elements! Both solutions are unacceptable from a broadcaster and a listener perspective. DMB Radio proposes two complementary tools to reconcile bandwidth requirements and visual service switching time.

The first tool is a light graphic representation (BIFS) that avoids the use of full-screen images. This compact description significantly reduces content access time and also avoids long image carrousels that drastically increase the probability of errors in broadcast environments.

The second tool is scalable multimedia content where BIFS and OD data are decomposed into dependent carrousels with increasing carousel periods [4]. This mechanism guarantees a fast access to a light visual presentation when switching on a radio receiver while the complete multimedia service is still progressively retrieved by the receiver. Elementary Stream dependencies are signalled in the BIFS and OD Object Descriptors which are basically composed of several BIFS and OD ES as illustrated in Figure 80. These MPEG-4 ES dependencies are specified by the MPEG-4 Systems standard and allow the decomposition of an interactive BIFS service into several complementary streams: an independent stream and several enhancement streams that depend on the main stream. The support of such enhancement streams is part of the minimal requirements for digital radio devices and is the mainstream of the application of our scalable multimedia model to the digital radio domain.

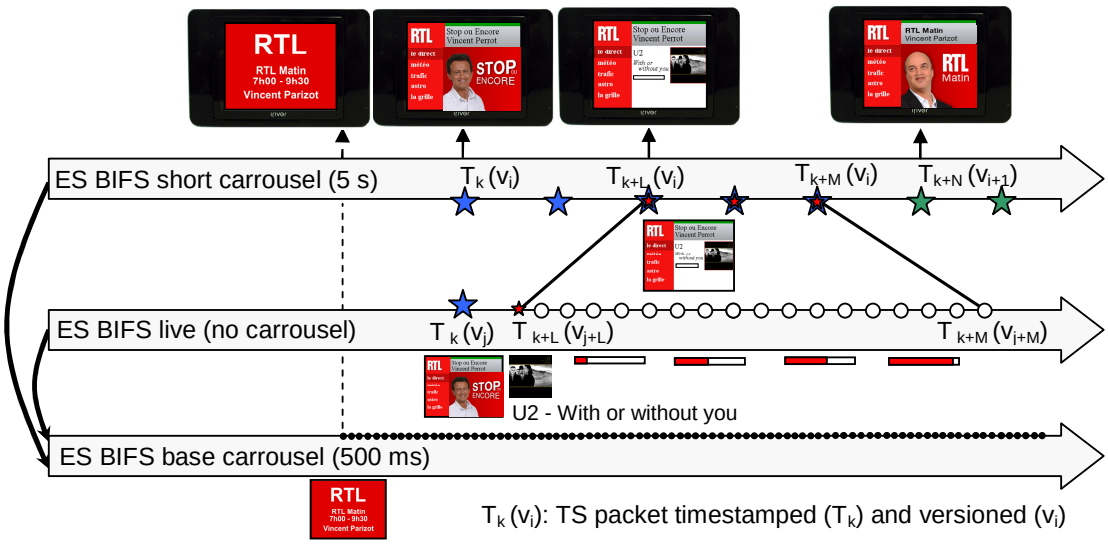


Figure 80: Scalable MPEG-4 BIFS services.

Chapter 10
APPENDIX C: Adaptation & digital radio devices

Before considering the adaptation of multimedia presentations to digital radio devices, it first sounds reasonable to make the context as suitable as possible for the service scenarios as they are already envisioned. Since the complete interoperability of multimedia services being broadcast and digital radio receivers require normative references and industrial endorsements, both ways have been followed to define multimedia profiles that a radio station can target when preparing a multimedia content. On the one hand, minimal requirements for digital radio devices have been specified (Section 10.1) and can be used to carefully design the *Base* documents of *Scalable MSTI* service. On the other hand, advanced requirements have been defined and can be implemented as an option in digital radio devices (Section 10.2).

10.1 Base requirements

Minimal requirements are essential in broadcast scenarios since they implicitly define the simplest service that must be broadcast by a radio station in order to be accessible to all digital radio receivers. In practice, our contributions to the Receiver Profile Task Force of the WorldDMB concluded into three main profiles[52]. These pan-European receiver profiles supported by the EBU and DIGITAL EUROPE feature hierarchical technical requirements which mandate multimedia standards: MPEG-4 BIFS (Receiver Profile 2) and an MPEG-4 AVC (Receiver Profile 3). Multimedia digital radio receivers (all for profiles except profile 1) must feature a colour screen display of at least 320 x 240 pixels.

These high-level requirements are further detailed in the technical specifications[53] produced by the SIMAVELEC, TDF and the GRN[54]. In particular, the handling of the Core2D@Level1 scene and graphics profiles of MPEG-4 BIFS remains mandatory as indicated by the T-DMB standard. Scene elements compliant to these profiles must be displayed on digital radio receivers above Profile 2 while provision for profile extensions is preserved (see Figure 81). As a consequence, a digital radio receiver must be able to decode and render some scene elements while it has to ignore any other that exceed its MPEG-4 BIFS decoding capability. Additionally, the key codes referenced by the multimedia scene in order to define presentation behaviour upon user interactions have been specified for classical buttons. More precisely, a keypad composed of a left-down-right-up and enter key set is mandated so that a complete navigation across the content can be guaranteed on all multimedia devices without pointing mechanisms.

In a similar way, the full flexibility of the Baseline level 1.3 profile of MPEG-4 AVC have been mandated on digital radio receivers (Profile 3). Since the video component of a digital radio service can be used to temporarily broadcast image slideshows, low frame rates and the dynamic activation of video stream must be appropriately handled by receivers. Additionally, image decoding capabilities of digital radio receivers have been specified in order to avoid the simulcast of image alternatives in various formats. Hence, digital radio receivers must support JPEG and PNG image format and transparency as allowed by PNG must be supported by all receivers. Moreover, image buffering strategies are also strongly suggested (see Figure 82) since they have a direct impact on the performance of the pre-fetching mechanisms that can be used to optimize the synchronization of the visual presentation with the audio.

[52] http://www.worlddab.org/public_documents/WorldDMB_Digital_Radio_Receiver_Profiles.pdf
[53] http://www.simavelec.fr/dossiers_traites.htm
[54] GRN (Groupement pour la Radio Numérique) gathers a large number of broadcasters (including RTL, RTL2 and Funradio) and aimed at promoting Digital Radio.

Extension of BIFS Profile of DMB

The T-DMB standard specifies the MPEG-4 BIFS profiles which define an interactive service that is synchronized with the audio stream. Indeed, sections 5.5.1 and 5.5.2 of the T-DMB standard directly refer to the Core2D@Level1 profiles for graphics elements and the scene graph. In order to provide extension mechanisms to these interactive services, T-DMB receivers SHOULD support multimedia services that contain multiple interactive services. These advanced T-DMB services are composed of one or more basic MPEG-4 BIFS elementary streams with different profiles, one of which has to be the Core2D@Level1 profile for graphics elements and the Core2D@Level1 profile for the scene graph.

The various interactive services available for a given T-DMB multimedia service are listed in the Program Map Table that describes the MPEG-2 TS program by defining several IODs, (one IOD per interactive service). It is up to the terminal to select the interactive service whose profiles (as described in IOD according to the applicable standard) correspond to its MPEG-4 BIFS decoding capability.

Figure 81: Provision for MPEG-4 BIFS profile extensions - quote from [53].

Image buffering

In order to offer fast navigation through a DMB Radio service being broadcast, receivers MUST buffer image Elementary Streams upon reception even if some images may not be currently activated or visible in the MPEG-4 scene. This buffer will not exceed 100Ko and is essential to guarantee a comfortable access to interactive content without unacceptable delays when all main images are locally available on the DMB receiver.

Additionally, DMB receivers MAY decode broadcasted images upon reception even if they are not activated or visible in the current MPEG-4 scene to avoid decoding delays when browsing the BIFS service of a radio program.

DMB receivers SHOULD allocate an image decoding buffer of at least 1Mo in order to cache all decoded images of a T-DMB service and offer a comfortable navigation to the user.

The image ES buffer and/or the image decoding buffer management can be performed by monitoring the image ES deletions and/or updates in PMT, ISO/IEC14496 Sections headers or OD Insert/Update/Delete. This buffer allocation can also be complemented with the tracking of referenced images in the MPEG-4 scene.

Figure 82: Minimal requirements for image caching and decoding buffer - quote from [53].

10.2 Advanced requirements

On the one hand, minimal requirements are essential for broadcast service to cope with a large panel of multimedia receivers. On the other hand, the compromises needed to define common requirements tend to reduce the richness of advanced devices. For this reason, optional requirements for digital radio devices[55] that are backward compatible with minimal requirements have been contributed to the

[55] http://mpeg.chiariglione.org/working_documents/mpeg-04/part11/New_BIFS_Prof_DigRad.zip

standardization so that radio stations can provide an advanced multimedia experience to auditors that are appropriately equipped.

The requirements on the technologies that could be used to support interactive radio services have been driven by the digital radio scenarios described in Chapter 2 and Appendix A [8][9][10][11]. It includes an enhanced tool box compared to Core2D@level1 profiles in order to provide more compact and flexible descriptions of a multimedia presentation. For instance, graphical elements such as curves or gradient enable rich-media content for live programs since their light representation (compared to an image) might be compatible with strong synchronization constraints (TR 4.2). In the same way, scene elements providing automated text layout management can be easily updated (TR 7.0) or scaled to different aspect ratio.

Since flexible and adaptable media components might require implementation costs that are not compatible with the actual performances of multimedia chipsets, the `<EnvironmentTest>` tool has been introduced in the new ExtendedCore2D scene profile (see Figure 83). Additionally, storage mechanisms have been defined to save and restore any field values of the scene to a private storage zone of the terminal. Hence, user preferences configured from the multimedia service can be automatically retrieved when switching on a radio station (TR 7.2). In the same way, MPEG-4 BIFS update commands dealing with external addresses have been specified in order to retrieve device information (TR 7.3) such as GPS coordinates for instance. All these optional functionalities can be used to define *Scalable MSTI* enhancements layers or configure scalability layers according to user preferences.

EnvironmentTest {

eventIn	SFBool	**evaluate**	
exposedField	SFBool	**enabled**	TRUE
exposedField	SFInt32	**parameter**	0
exposedField	SFString	**compareValue**	NULL
exposedField	SFBool	**evaluateOnChange**	TRUE
eventOut	SFBool	**valueLarger**	
eventOut	SFBool	**valueEqual**	
eventOut	SFBool	**valueSmaller**	
eventOut	SFString	**parameterValue**	

}

Functionality and semantics

The EnvironmentTest node enables testing a **parameter** of the terminal environment, possibly comparing their values with the **compareValue**. The evaluation of the parameter triggers different eventOuts depending on the type of the parameter:

- If the **parameter** type is Boolean, the evaluation triggers a **valueEqual** eventOut, and the **compareValue** field is ignored.
- If the **parameter** type is a number and the **compareValue** represents a number, the two values are compared and the following eventOuts are generated:
  - **valueEqual** if **parameter** and **compareValue** are equal
  - **valueLarger** if **compareValue** is strictly larger than **parameter**
  - **valueSmaller** if **compareValue** is strictly less then than **parameter**

The supported parameter types are defined in the following table.

Value	Definition	Type
0	Display region Aspect Ratio (larger dimension divided by smaller dimension, regardless of screen orientation)	Float
1	Portrait mode of the display region (TRUE if width<height)	Boolean
2	Display region width in pixels	Integer
3	Display region height in pixels	Integer
4	Horizontal DPI	Integer
5	Vertical DPI	Integer
6	Automotive Situation (terminal user drives a moving vehicle)	Boolean
7	User is Visually Challenged	Boolean
8	Touch Screen present on terminal	Boolean
9	Navigation Keypad present on terminal	Boolean
0x00000007-0xEFFFFFFF	ISO Reserved	
0xF0000000-0xFFFFFFFF	User Reserved	

In any case, the **parameterValue** eventOut is triggered after evaluation.
If **evaluateOnChange** is set to FALSE, the node only evaluates upon receiving the **evaluate** eventIn; otherwise, the node evaluates on any change of **parameter** or **compareValue**.
The node evaluates and triggers events only when its **enabled** field is true.

The display region is the area onto which the BIFS content is rendered. This region may be the entire screen, some part of the screen or an off-screen memory region.

Figure 83: EnvironmentTest tool for Scalable MSTI adaptation updates - quote from [60].

Chapter 11
APPENDIX D: Authoring of multimedia radio services

Although radio programs are often perceived as a live production, content is mostly composed of audio sequences prepared off-line (also called master tapes) that are scheduled and launched by the radio show presenters with live comments (interviews, reportages, music titles, commercials…). Since the live production of multimedia data for digital radio will involve a significant evolution of the current working methods in radio stations, a first step consists in producing multimedia data along with master tapes. Indeed, the launch of an audio item from a mixing console can be notified using General Purpose Input/Output (GPIO) components and automatically cascaded with the start of its associated multimedia content (when available). Furthermore, a scheduling based on a playlist mode allows fetching multimedia content a few seconds before starting its related audio sequence in order to fulfil the precise synchronization of rich-media presentations.

In order to ease the off-line production of multimedia digital radio content, an authoring tool has been developed. As part of these developments, we have proposed evolutions and enhancements in order to support the proposed scalability approach.

The main functions of the authoring tool, called BIFSEdit, developed for the production of multimedia digital radio services are presented in Section 11.1. Then, the high-level principles of our authoring tool based on the principles of our scalable multimedia model are described in Section 11.2. Finally, an overview of the XML structure of the input description of the BIFSEdit software is provided in Section 11.3. In particular, the strategy that was defined to map scalable layers onto T-DMB streams when exporting a presentation is explained.

11.1 Overview of the BIFSEdit authoring tool

The BIFSEdit software is an authoring tool dedicated to the off-line edition of multimedia digital radio services. Multimedia documents, described in the MPEG-4 XMT-A format, can be loaded into the authoring tool, modified and stored backed to the MPEG-4 XMT-A format. A BIFSEdit authoring project includes one or several audio references, provides editing hints and describes the organization of XMT-A documents into a multimedia presentation. This project can be loaded and stored in a proprietary project description (i.e. XML *Show* format) and exported to the XMT-A or MP4 format for playback. Such an export of the BIFSEdit project is a lossy transformation since all editing hints and the MSTI scalable structure of the digital radio project description are not exported. In the following, snapshots of the main functionality of the authoring tool are described.

The authoring of a multimedia presentation requires advanced skills in order to achieve presentations including a rich interactivity and user-friendly animations. The development of an authoring tool including these scene properties would imply significant implementation efforts. Additionally, such sophisticated software might be too complicated for users or suffer important simplifications that are not satisfactory. As a consequence, the authoring tool that was developed to edit digital radio presentation remains as simple as possible by combining a timeline-based positioning of multimedia content (Section 11.1.1), a WYSIWYG configuration of spatial properties (Section 11.1.2), and presentation templates which can be designed according to the editorial rules of the radio station (Section 11.1.3).

11.1.1 Timeline-based authoring

The BIFSEdit authoring tool decomposes a digital radio service into labelled temporal segments corresponding to audio fragments that can be associated with one or more multimedia services. In Figure 84, the ‘meteo’ temporal interval is associated with the ‘meteo_paris’ and ‘meteo_marseille’ services. Besides, the duration of these services can be visualized and modified through the ‘Ligne temporelle’ tab.

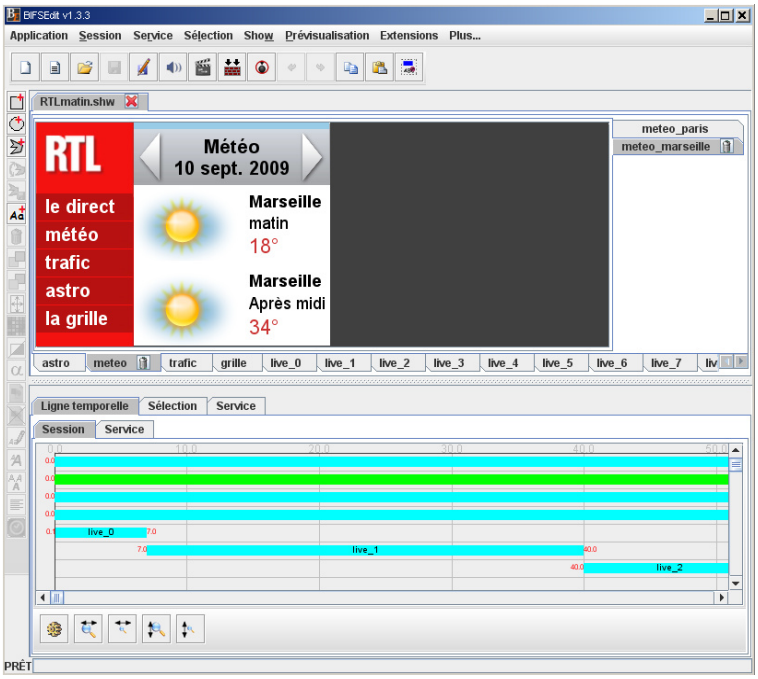


Figure 84: The timeline view of the BIFSEdit authoring tool.

11.1.2 WYSIWYG authoring

BIFSEdit is a WYSIWYG authoring tool that provides a direct access to a large set of the scene properties that are part of the MPEG-4 BIFS Core2D profiles: Rectangle, Circles, Polygons, Text and Images. These properties can be created from scratch using media primitives and further modified using the left toolbar whose items are activated according to the type of media being manipulated as illustrated in Figure 85. Additionally, scene properties of media components can be directly edited from the 'Sélection' tab as illustrated in Figure 86.

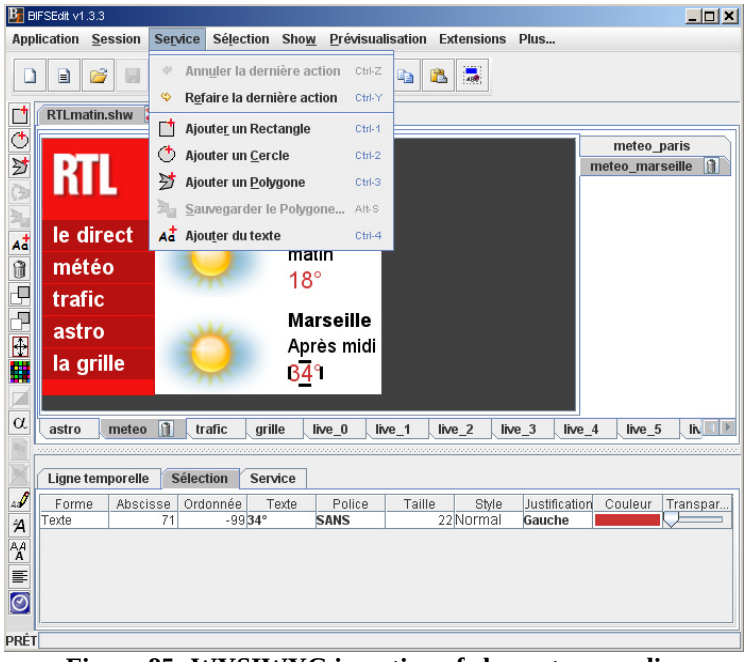


Figure 85: WYSIWYG insertion of elementary media.

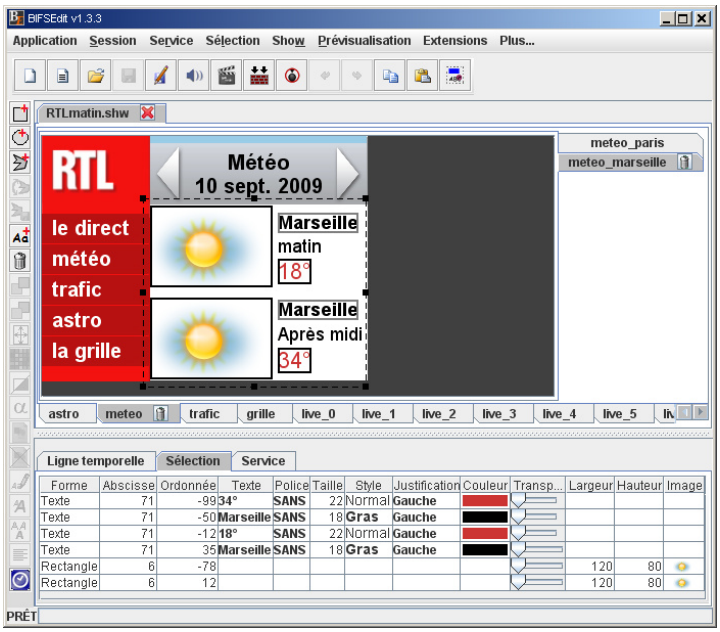


Figure 86: WYSIWYG handling of spatial scene properties.

11.1.3 Template-based authoring

All media elements and scene properties can not be handled through the WYSIWYG interface. For instance, timed animations and content interactivity cannot be modified using the BIFSEdit authoring tool. Instead, presentation templates which include configurable media and all scene logic required for the attractive playback of the presentation can be loaded during authoring and displayed as a background image during service authoring. In Figure 87, the 'meteo_marseille' service is attached to the 'MeteoTemplate' template which inherits the menu bar from the 'root' template. The principles of our template-based approach for multimedia digital radio services are presented in Section 11.2.

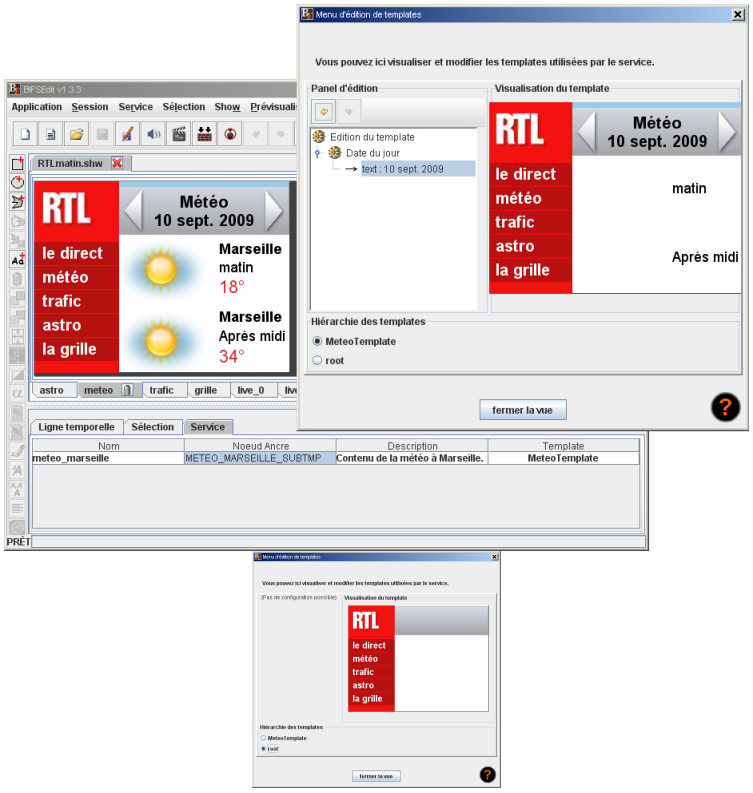


Figure 87: The template view of the BIFSEdit authoring tool.

11.2 Multimedia scene scalability and hierarchical templates

The main requirement for the BIFSEdit authoring tool is that multimedia presentations have to be prepared in order to be transmitted on a narrow-bandwidth channel. In particular, the progressive rendering of multimedia services on digital radio devices is needed. For that purpose, the scalability features of the *Scalable MSTI* approach have been transposed in the BIFSEdit authoring environment. Moreover, our approach still includes adaptation functionality that can be used to deal with in-car reception where animations need to be cancelled and to provide suitable presentation for receivers that do not have any interactive means.

In order to introduce scene scalability to the authoring process, the BIFSEdit tool handles two different concepts: template and content service. Content services are attached to templates and store all the spatial properties defined by the user through the WYSIWYG interface. Templates are scene descriptions including spatial, temporal and interactive properties than can be filled in with additional scene properties. Hence, a template can be attached to another template thereby creating a hierarchical template structure. Furthermore, each template can be sub-divided following the MSTI classification. For instance, a two-pane or three-pane interactive slideshow can be configured by selecting the appropriate *Interactive* description related to an MSTI template. Finally, the scene properties contained inside templates and content services are scene updates to be cascaded. In particular, all *Spatial*, *Temporal* and *Interactive* but also the *Media* descriptions have been described using MPEG-4 BIFS scene updates. In particular, this allows collapsing STI layers as introduced in Section 6.5.3.2 but also *Media* descriptions which are divided into several hierarchical pieces in that case.

The multimedia scene scalability has been defined by assigning layers in the document hierarchy. As illustrated in Figure 88, the authored multimedia document results from the aggregation of multiple MSTI sub-documents through scene transformations. In this example, four layers have been defined (L_0, L_1, L_2 and L_3).

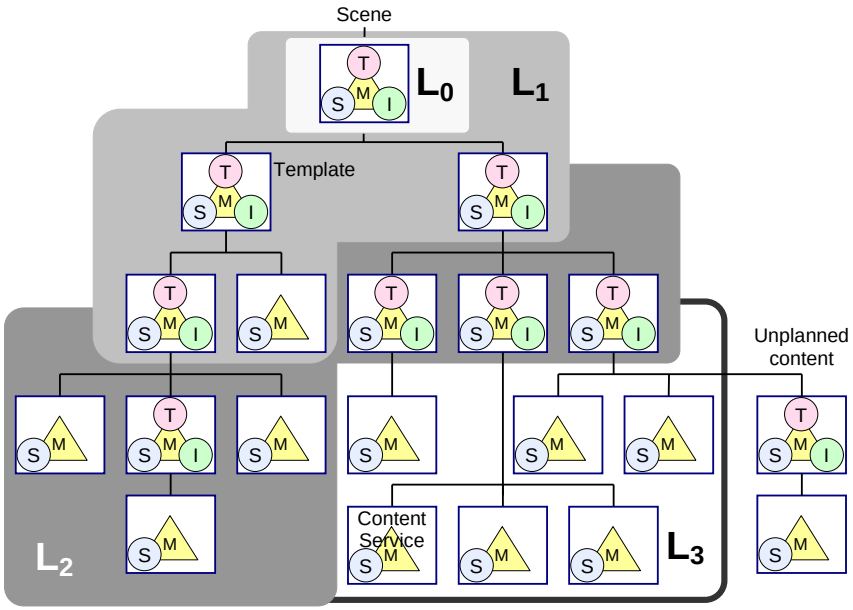


Figure 88: The layered structure of multimedia documents in the BIFSEdit authoring tool.

11.3 BIFSEdit project description

The XML description given as input to our authoring tool to load or store a digital radio content project, called a *Show*, is illustrated in Code 53 on a simplified example. Each show is related to one or several audio sequences that are referenced using `<Audio>` descriptions. The structure of a BIFSEdit project description is composed of audio chunks that indicate key timecodes of the audio sequence. As a consequence, each `<AudioChunk>` description contains `<service>` or `<template>` descriptions referencing XMT-A files. These XMT-A files contain MPEG-4 BIFS scene fragments that can be

attached to other scenes by pointing to identified `anchorNode` elements. These MPEG-4 BIFS commands are applied at the time specified by a `start` attribute of `<AudioChunk>` elements.

Each service and template indicates the scalable layer it belongs to (`layer` attribute). As shown in Code 53, some of them are grouped into a single layer and specific tagging (`type` attribute) has been introduced according to the broadcast strategy described in the following: `base` (Section 11.3.1), `live` (Section 11.3.2), `canvas` (Section 11.3.3) and `asynchronous` (Section 11.3.4).

```
<Show xmlns="ShowSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"">
 <Template id="root" type="base" layer="0" url="base.xmt"/>
 <Audio start="0.0" url="RTLMatin_audio.mp4">

 <AudioChunk id="astro" start="0.0">
  <Template type="asynchronous" layer="5" id="AstroTpl" template_ref_id="root"
            anchorNode="ASTRO_SERVICE_HEADER" url="astro_subtmp.xmt"/>
  <Template type="asynchronous" layer="5" id="BelierTpl" template_ref_id="AstroTpl"
            anchorNode="ASTRO_BELIER_SUBTMP" url="signe_belier.xmt"/>
  <Template type="asynchronous" layer="5" id="TaureauTpl" template_ref_id="AstroTpl"
            anchorNode="ASTRO_TAUREAU_SUBTMP" url="signe_taureau.xmt"/>
   <Service id="astro_belier" type="asynchronous" layer="5" template_ref_id="BelierTpl"
            anchorNode="ASTRO_BELIER_ANCHOR" url="astro_service_content_01.xmt"/>
   <Service id="astro_taureau" type="asynchronous" layer="5" template_ref_id="TaureauTpl"
            anchorNode="ASTRO_TAUREAU_ANCHOR" url="astro_service_content_02.xmt"/>
 </AudioChunk>

 <AudioChunk id="meteo" start="0.0">
  <Template id="MeteoTpl" type="asynchronous" layer="4" template_ref_id="root"
            anchorNode="METEO_SERVICE_HEADER"  url="meteo_subtmp.xmt">
    <Config field="string" node="TEXT_DATE" value="7 Octobre 2010"/>
  </Template>
  <Service id="meteo_paris" type="asynchronous" layer="4" template_ref_id="MeteoTpl"
           anchorNode="METEO_PARIS_SUBTMP" url="meteo_service_content_01.xmt"/>
  <Service id="meteo_marseille" type="asynchronous" layer="4" template_ref_id="MeteoTpl"
           anchorNode="METEO_MARSEILLE_SUBTMP" url="meteo_service_content_03.xmt"/>
 </AudioChunk>

 <AudioChunk id="live_0" start="0.1" stop="7.0">
   <Template type="canvas" layer="1" id="rtl_canvas" template_ref_id="root"
            anchorNode="BASE_CANVAS" url="template_RTL.xmt"/>
   <Template type="canvas" layer="2" id="rtl_matin_canvas" template_ref_id="rtl_canvas"
            anchorNode="LIVE_CANVAS" url="template_RTLMatin.xmt"/>
   <Service  type="live" layer="3" id="rtl_matin_into" template_ref_id="rtl_matin_canvas"
            anchorNode="LIVE_SERVICE_CONTENT" url="RTLMatin_intro.xmt"/>
 </AudioChunk>

  <AudioChunk id="live_1" start="7.0" stop="40.0">
   <Service id="live_meteo" type="live" layer="3" template_ref_id="rtl_matin_canvas"
            anchorNode="LIVE_SERVICE_CONTENT" url="meteo.xmt"/>
  </AudioChunk>
<Show>
```

Code 53: An BIFSEdit project description (Show description).

11.3.1 Base service

The base service of a digital radio program aims at offering a simple multimedia content to a user switching to a radio station within a very short delay (from 500ms to 2s). Therefore, the description of a base service contains the main audio object along with some limited data in a text format (e.g. the name of the station, the title of the program) as illustrated in Figure 89.

Figure 89: Exemple of a Base service (repeated every 500 ms).

The main objective when authoring a base service is to optimize, as much as possible, the payload of the MPEG-2 TS packets containing this initial scene description which is repeated every 500 ms and constitute an entry point for the MPEG-4 presentation (Random Access Point or RAP). As a consequence, this scene description must not exceed 172 bytes[56] to be contained in a single MPEG-2 TS packet which will consume about 3kbps.

The updates applied to the base service over time could be signalled by incrementing the counter (version_number) of MPEG-2 sections header containing an MPEG-4 BIFS `Replace Scene` command. However, the scene description of the base service always contains the definition of the audio object and the signaling of a modification in the audio stream reference might be misinterpreted by some receivers. In particular, it could lead to the reloading of the audio object inducing a short interruption in the audio stream which is not acceptable. As a consequence, updates applied to the *Base* service might not be signaled in order to avoid disturbance for receivers connected to the DMB service. Such an approach still guarantees an up-to-date welcome page for receivers when connecting to the service as illustrated in Figure 90. If necessary, suitable updates of the base service targeting connected devices might be performed through the live stream described in Section 11.3.2.

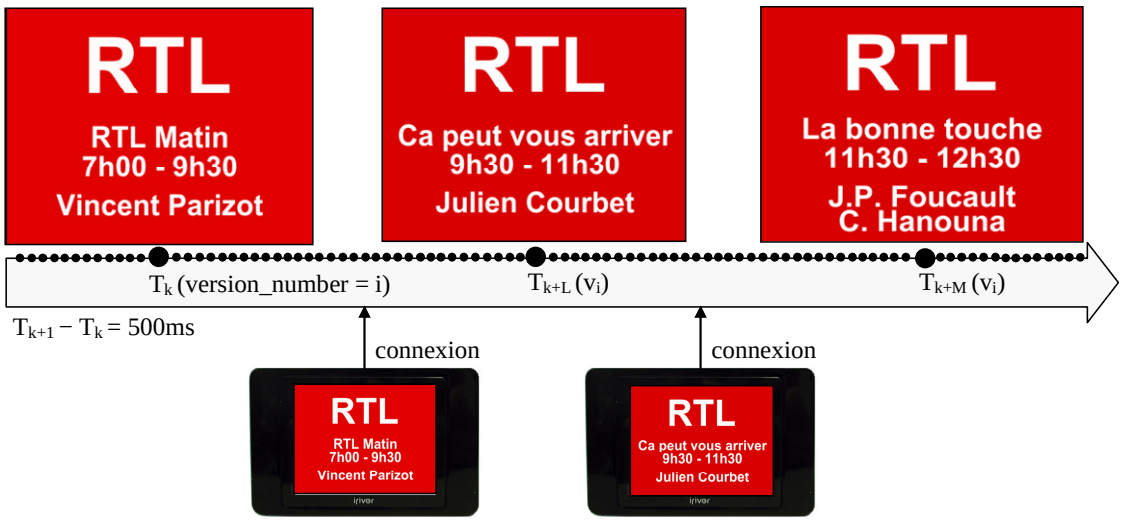


Figure 90: Update of the Base service.

11.3.2 Live stream

Scene update commands may have short duration and limited consequences on the scene structure. Indeed, some updates are only valuable at a precise time and quickly becomes useless when this moment has passed. An extreme example of these *ephemeral* updates is the height of the volume bars of a VU-meter display. Additionally, all updates do not introduce scene grouping elements or new node identifiers but simply modify some node attributes (e.g. replacing some text, the color of a rectangle) or remove some parts of the scene tree (e.g. cleaning out some unused materials). As a consequence, updates commands that will come after these *ephemeral* updates will not depend upon them. Hence, these updates do not have to be repeated over time through carrouseling since they are only useful to receivers connected to the radio program when events occurred. However, they might be repeated in a short period of time in order to maximize the reception success in noisy transmission environments. Since this scene resending will not be signalled using the MPEG-2 carrousel mode, the same scene transformations might be repeated on some receivers. For this reason, the use of insert commands is not recommended in that case.

As illustrated in Figure 91, the *ephemeral* updates are transmitted to digital radio receivers on an MPEG-4 Elementary Stream which depends on the *Base* ES which conveys the base service. This MPEG-4 ES

[56] 188 (payload TS) – 4 (header TS MPEG-2) – 8 (header Section MPEG-2) – 4 (CRC).

is called the *live* stream in the following. Whatever the scene elements contained in the MPEG-2 section, the version number is always incremented so that the BIFS commands of the *live* stream, possibly grouped into a single MPEG-2 TS packet, are always executed by the digital radio receiver.

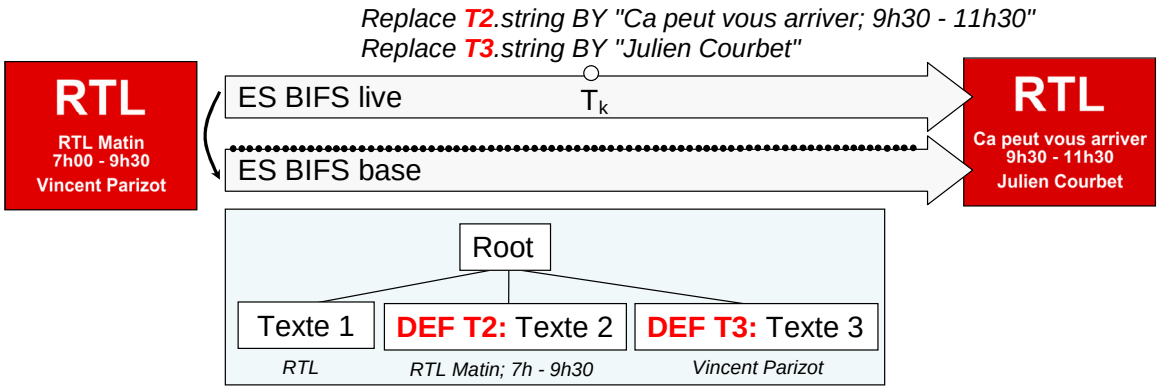


Figure 91: Update of the Base service through the live BIFS stream.

11.3.3 Canvas stream

Contrary to *ephemeral* updates, BIFS commands might have long-term consequences on the presentation and can define node identifiers in the scene tree. For instance, the name of an artist and of his song may last for three or four minutes. Other updates introduce scene structures that will be referenced by *live* updates all along a one-hour program. These *persistent* updates need to be transmitted to connected receivers though a *live* BIFS stream but must also be repeated regularly (in carrousel mode) so that receivers switching to the radio station can also have access to a multimedia service that includes the latest updates.

As illustrated in Figure 92, these *persistent* updates are aggregated in MPEG-2 sections whose version number is not always incremented and transmitted on an MPEG-4 ES which depends on the *base* BIFS ES. Indeed, these BIFS commands are also transmitted through the *live* stream to digital radio receivers connected to the program. In that case, receivers can simply ignore these repeated BIFS commands. Since the *base* service of our scalable approach only contains a simplistic multimedia presentation due to bandwidth constraints, an MPEG-4 ES in carrousel mode has been reserved to convey the main scene which defines the presentation layout and interactivity for the digital radio program. This MPEG-4 ES is called the *canvas* stream in the following.

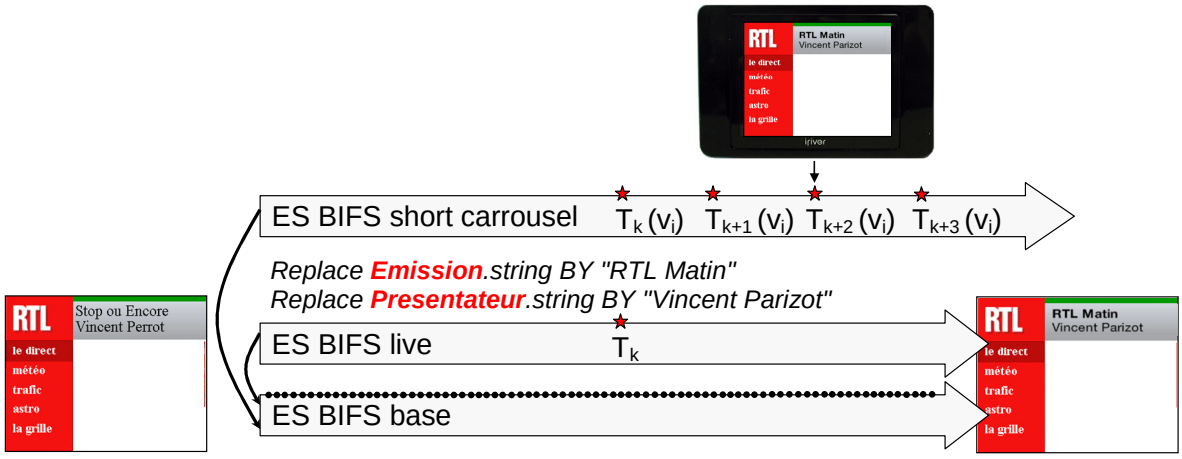


Figure 92: Carrouseling of a persistent scene update.

Persistent updates require the management of the scene tree in order to be able to aggregate data to the current scene state that is repeated in carrousel mode. Additionally, regular (but not too frequent) refreshes are needed to guarantee the access of the multimedia presentation to connecting auditors at the cost of an acceptable bandwidth. Hence, each significant transformation of the presentation is first

applied to the main scene and then transmitted on the *canvas* stream with the appropriate signaling at the MPEG-2 level as illustrated in Figure 93. In this example, it can be noticed that the beginning of a program triggers a scene update signaling (version number increment) since scene elements might be very different from a program to another. The progression bar of the current song is not aggregated on the *canvas* since its current status is considered as an *ephemeral* update and will be updated shortly after.

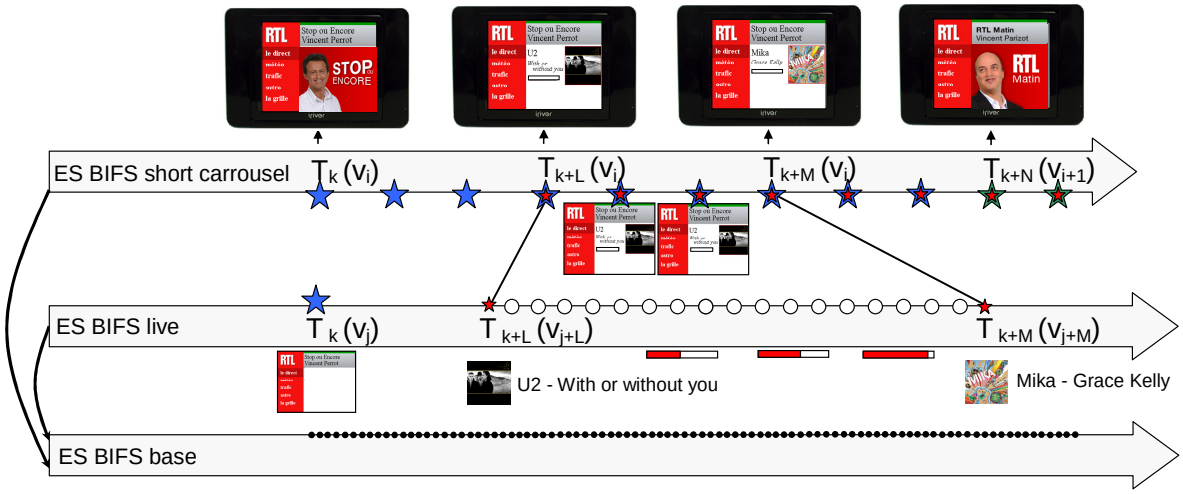


Figure 93: Aggregation of the persistent live BIFS updates on a short carrousel.

11.3.4 Asynchronous services

All digital radio services do not mandate synchronization with the audio stream. For instance, a weather forecast service accessed through interactivity is not related to the audio stream when continuously broadcast. From a technical point of view, the delivery of *asynchronous* scene elements relies on the same mechanisms as synchronized data. However, two essential aspects enable important bandwidth optimizations in that case:

- The amount of data contained in an asynchronous service can be much more important than in the case of synchronized data. Indeed, the 'downloading' time for these supplemental scene elements is less critical in that case. For this reason, asynchronous services can be conveyed on one (and more) MPEG-4 ES with a long carrousel period.
- Asynchronous services have a lower priority than synchronized services. Indeed, it is important that the scene elements and their related images are received on the digital radio device before the time code used for the synchronization of the audiovisual presentation is reached. As a consequence, it might be acceptable that the scheduling of the some asynchronous data is cancelled or delayed in order to release some additional bandwidth for synchronized data.

The update of asynchronous services is signalled by incrementing the version number of MPEG-2 section carrying these supplemental digital radio services. It should be noticed that such an update implies the reloading of all asynchronous data if these updates are conveyed in a single *asynchronous* stream as illustrated in Figure 94. So, multiple *asynchronous* streams could be used in order to avoid such artefacts. However, the number of elementary streams must remain limited since each stream requires an appropriate signalling at the MPEG-2 (PMT) and MPEG-4 (OD) level which reduce data payload. As a consequence, the structure of asynchronous services is seldom modified so that connected receivers keep their downloaded scenes as long as possible. Instead, updates of asynchronous services are signalled on the *live* BIFS stream and possibility aggregated as for the main scene *canvas*. Besides, it can be noticed on Figure 94 that the *asynchronous* stream does necessarily depend on the *canvas* stream although it is implicitly attached to it. In our approach, empty node references are specified in the *base* service and constitute a multimedia dictionary that is fed with their *asynchronous* data. Additionally, dictionary identifiers are referenced by the *canvas* scene so that asynchronous services are maintained while updating the main scene.

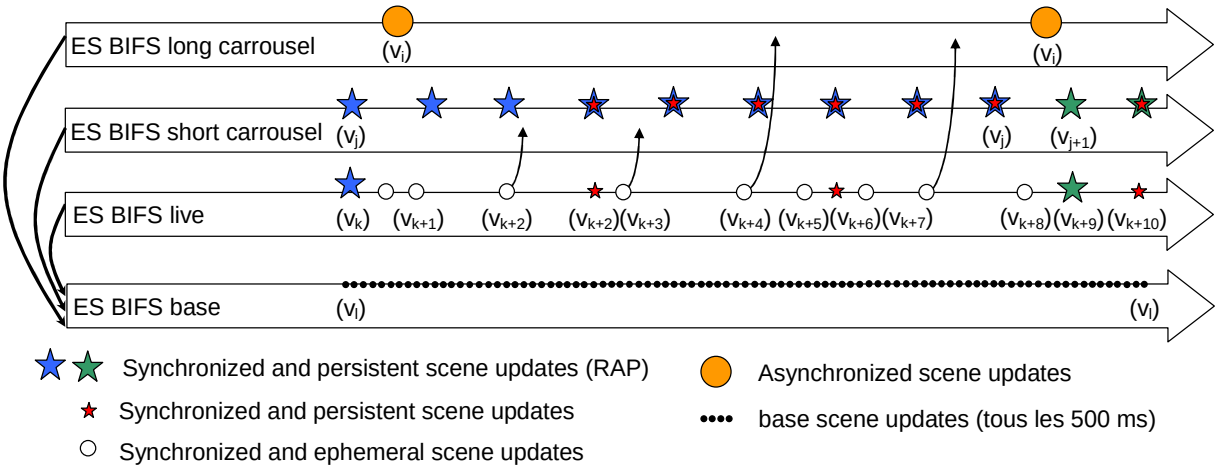


Figure 94: Overview of multimedia scene scalability in a digital radio service.

Chapter 12
APPENDIX E: Broadcasting of multimedia radio services

In this chapter, all experimentations have been made in the context of the T-DMB standard [38]. Digital Multimedia Broadcasting (DMB) for radio, also called DMB Radio, is part of Eureka 147 digital family of standards and therefore relies on the same network layer (e.g. packet mode statistical multiplexing) and the same physical layer (e.g. error-correction coding and OFDM modulation) as the Digital Audio Broadcasting (DAB/DAB+) standards [43][41]. In a few words, DMB Radio differs from DAB/DAB+ in its presentation layer that is based on the MPEG-4 multimedia framework designed for the synchronized playback of audio and advanced multimedia content. Within the Terrestrial (T)-DMB specification, two application fields are identified: mobile television and interactive radio. For more information about the digital radio broadcasting, please refer to Appendix B [13].

12.1 The Radio+ project

The Radio+ project is a two-year research project managed by RTL in order to develop an enhanced multimedia generation chain for visual and interactive digital radio programs based on the T-DMB standard. As a consequence, the Radio+ consortium gathers four companies and Telecom ParisTech that represent the different actors of the multimedia production chain: RTL (audio and multimedia production), Telecom ParisTech (multimedia encoding, decoding and packaging), Allegro DVT (audio encoding and DMB multiplexing), TDF (DAB multiplexing and RF broadcasting) and Cameon (RF reception and interactive audiovisual playback on an handheld receiver called ‘Diabolo’).

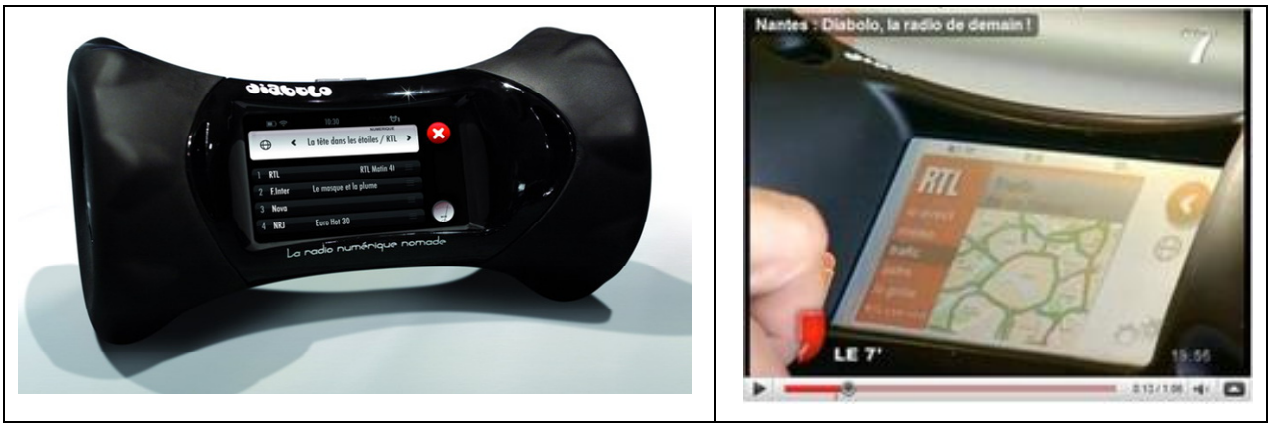


Figure 95: The Diabolo digital radio device.

The main objective of the Radio+ project is to optimize the methods and tools that contribute to the creation, the production and the broadcasting of interactive radio programs. In the scope of this project, a significant effort was spent on the interface, called DMB-ML, between the output of the multimedia production chain of a radio station and the input parameters of a DMB multiplexer which is in charge of content encoding and DMB multiplexing.

12.2 The DMB markup language

As explained in Appendix D, the scalable digital radio documents produced by the BIFSEdit authoring tools are standalone content that have been designed for a hierarchical broadcasting of its scene elements over dependent MPEG-4 Elementary Streams. During broadcasting, scene descriptions contained in each scalable layer need to be transmitted over time with an accurate synchronization with the audio stream and according to the available bandwidth that is not consumed by the audio component of the digital radio program.

The DMB-ML interface aims at configuring the DMB multiplexer from the multimedia production chain. It provides:

- the configuration of the MPEG-2 multiplex (the number of steams, their types and also information about MPEG-2 signalling)
- the multimedia data to be encoded (i.e. MPEG-4 XMT-A scene commands)
- the precise insertion time of multimedia data into the multiplex. It should be noticed here that multiplexing time might differ from presentation time since elementary media might be pre-fetched to be dynamically activated upon synchronization events.
- the multiplexing directives driving the carrouseling of data being broadcast so as to guarantee to the user an acceptable access time to the multimedia service while maintaining a high-quality program sound.

12.2.1 DMB-ML syntax overview

The configuration parameters of the MPEG-4 streams and their carried multimedia data are described in a broadcast configuration using `<Scene>` or `<Image>` descriptions for scene elements and images. These stream configurations can also be modified over time through broadcast updates using `<SceneDataUpdate>`, `<ImageDataUpdate>` or `<StreamParamUpdate>` description as illustrated in Figure 96. All these descriptions rely on a limited set of attributes which drive the DMB multiplexer. For instance, `<SceneDataUpdate>` and `<ImageDataUpdate>` allows providing multimedia data along with the MPEG-2 signalling hints (`refresh` attribute) or aggregation directives (`aggregation` attribute) which allows the automatic concatenation of MPEG-4 BIFS updates on data currently broadcast in carrousel mode.

The `<StreamParamUpdate>` description is a key element of the dynamic multiplexing strategy since it provides updates on the multiplexing directives for a specific stream identified using the `target-es-id` attribute. Each `<StreamParamUpdate>` description is applied at a precise timestamp defined by its `at` attribute. In practice, two main parameters can be configured to define a multiplexing strategy: the duration of the carrousel and stream priority:

- The `carrousel-period` attribute is a positive integer (32 bits) that determines the expected time for a complete carrousel of the data on the multimedia stream. A attribute value of '1000' indicated that the time interval between two successive repetitions of the same data on the multimedia stream should be shorted than 1000 ms.
- The `priority` attribute is a positive integer between 1 and 100 that indicates the importance of the data conveyed on the multimedia stream with respect to the other multimedia streams of the DMB session. An attribute value of '1' gives the highest priority while an attribute value of '100' gives the lowest priority to the stream.

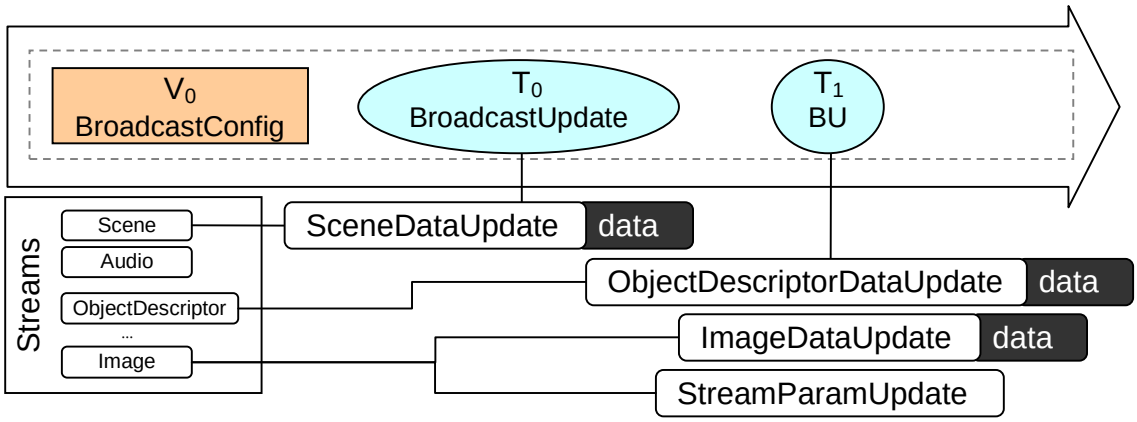


Figure 96: Overview of the DML-ML multiplexing protocol.

12.2.2 Example of a DMB-ML description

The DMB-ML example provided in Code 54 is an output from the BIFSEdit authoring tool. It includes three types of Elementary Streams: BIFS, OD and Image streams. The four scalable layers generated during presentation export can be visualized from the `radioplus:type` attributes: `BIFS_BASE`, `BIFS_CANVAS`, `BIFS_LIVE` and `BIFS_ASYNCH`. These attributes are extensions of the DMB-ML XML schema since the DMB multiplexer only have to deal with streams `priority` attribute. Several image streams are associated with these hierarchical BIFS streams. Some of them (`es-id="201"`, `es-id="202"` and `es-id="203"`) are related to the scene canvas, some others (`es-id="301"`, `es-id="302"`, `es-id="305"` and `es-id="306"`) belong to the asynchronous services and a single stream (`es-id="401"`) is dedicated to live content. The priority of each scene and image stream has been configured according to authors'wishes. For instance, the weather forecast has a higher priority than traffic information or the EPG. The DMB multiplexer can rely on these hints during the multiplexing of the multimedia service.

```
<DMBML">
  <BroadcastConfig refresh="false">
  <Stream carrousel-duration="500" enable="true" es-id="1" priority="1"
          radioplus:type="BIFS_BASE" type="BIFS" url="bifs/BIFS_base.xmt"/>
  <Stream carrousel-duration="5000" enable="true" es-id="2" priority="2"
          radioplus:type="BIFS_CANVAS" type="BIFS" url="bifs/BIFS_template.xmt"/>
  <Stream carrousel-duration="60000" enable="true" es-id="3" priority="3"
          radioplus:type="BIFS_ASYNCH" type="BIFS" url="bifs/BIFS_asynchronous.xmt"/>
  <Stream carrousel-duration="500" enable="true" es-id="4" priority="1"
          radioplus:type="BIFS_LIVE" type="BIFS"/>
  <Stream carrousel-duration="500" enable="true" es-id="11" priority="2"
          radioplus:type="OD_BASE" type="OD" url="OD_init.xmt"/>
  <Stream carrousel-duration="500" enable="true" es-id="12" priority="2"
          radioplus:type="OD_LIVE type="OD"/>
  <Stream carrousel-duration="5000" enable="true" es-id="201" priority="3"
          type="IMAGE" url="images/horizontal_tab.png"/>
  <Stream carrousel-duration="5000" enable="true" es-id="202" priority="3"
          type="IMAGE" url="images/vertical_tab.jpg"/>
  <Stream carrousel-duration="5000" enable="true" es-id="203" priority="3"
          type="IMAGE" url="images/nav_tab.PNG"/>
  <Stream carrousel-duration="60000" enable="true" es-id="301" priority="4"
          type="IMAGE" url="images/meteo_cloudy.PNG"/>
  <Stream carrousel-duration="60000" enable="true" es-id="302" priority="4"
          type="IMAGE" url="images/meteo_sunny.PNG"/>
  <Stream carrousel-duration="100000" enable="true" es-id="305" priority="5"
          type="IMAGE" url="images/trafic_paris.PNG"/>
  <Stream carrousel-duration="60000" enable="true" es-id="306" priority="6"
          type="IMAGE" url="images/grille.png"/>
  <Stream carrousel-duration="5000" enable="true" es-id="401" priority="3"
          type="IMAGE"/>
 </BroadcastConfig>

 <BroadcastUpdate radioplus:at="0.0">
   <StreamDataUpdate at="0.1" carrousel="false" target-es-id="4"  url="BIFS_live.xmt"/>
  <StreamDataUpdate at="0.1" carrousel="false" target-es-id="12" url="OD_live_0.1.xmt"/>
  <StreamDataUpdate at="7.0" carrousel="false" target-es-id="401" url="meteo_sunny.PNG"/>
 </BroadcastUpdate>

 <BroadcastUpdate radioplus:at="6.5">
  <StreamDataUpdate at="7.0" carrousel="false" target-es-id="4" url="BIFS_live_7.0.xmt"/>
  <StreamDataUpdate at="7.0" carrousel="false" target-es-id="12" url="OD_live_7.0.xmt"/>
 </BroadcastUpdate>

 <BroadcastUpdate radioplus:at="30">
  <StreamDataUpdate at="40.0" carrousel="false" target-es-id="401" url="srv_cont.jpg"/>
 </BroadcastUpdate>
</DMBML">
```

Code 54: A DMB-ML description.

12.3 MPEG-2 TS carrousel simulator

The DMB-ML interface was developed so that the DMB service generated by a radio station and broadcast to digital radio receivers leverage the production optimizations that are enabled by the authoring of scalable multimedia presentations. However, some broadcast configuration parameters transmitted to the DMB multiplexer along with images and scene elements are targeted value that might not be reached in practice. In particular, the carrousel duration of a stream might not be fulfilled if the available bandwidth is not sufficient during the live broadcast. For this reason, the DMB-ML interface would take advantage from a return channel from the DMB multiplexer to the multimedia production module. Such a dynamic feedback would indicate if targeted carrousel durations are respected and if the synchronization of presentation updates with the audio stream is accurately fulfilled.

In order to improve the scheduling of the scalable layers of digital radio presentations, an MPEG-2 TS carrousel simulator was developed. This software is in charge of simulating the multiplexing of the different streams defined by the DMB-ML interface into a DMB service.

12.3.1 Overview of the simulator

The MPEG-2 TS carousel simulator multiplexes several MPEG-4 Elementary Streams into a single MPEG-2 TS stream. However, this software cannot be considered as a complete DMB multiplexer since MPEG-2 and MPEG-4 signaling are not taken into account. Hence, MPEG-2 PAT and PMT tables are not considered and all MPEG-2 TS packets assume the same payload size although different MPEG-2 adaptation field are used in T-DMB over time (MPEG-2 Program Clock References are regularly transmitted for instance). Furthermore, since no audio coding is performed, a fixed bandwidth is allocated to all multimedia data whereas it is not realistic due to the MPEG-4 AAC variable-length coding. Despite all of these limitations, the results from the experiments conducted with the MPEG-2 TS carrousel simulator can give an overview of the scalable multiplexing of the digital radio content.

The MPEG-2 TS carrousel simulator produces detailed logs but also offer a visual interface that provide some key outputs:

	Data broadcasting is activated
	Data broadcasting is not activated
	New data is being broadcast (number of MPEG-2 TS packet to be transmitted)
	New data delivered cannot be delivered on time
	Data was delivered (number of carousel loop)
	Data was delivered late (delay)
	Data carrousel was repeated
	The carousel duration was not respected and is extended
	The carrousel loop is finished and carrousel duration was extended (extended duation)
	Data delivery has been cancelled
	Current data buffer is flushed (number of remaining MPEG-2 TS packet)

Figure 97: Overiew of the MuxSimulator GUI

12.3.2 DMB simulations

The MPEG-2 TS carrousel simulator can be used to define a scheduling strategy of the scalable layers of the digital radio presentation. Indeed, the bandwidth consumed by the *live* BIFS stream and its associated image Elementary Streams might have an important impact over carrousel durations. Such extended carrousel durations can prevent accessing the multimedia service (due to a higher probability of transmission errors) or can even be cancelled before it finishes (due to up-of-date data carrouseling). For this reason, the control of carrousel durations is a key parameter for multimedia digital radio broadcasting. In the following, simulations are provided and illustrate some of the parameters that can be configured to manage T-DMB broadcasting.

Figure 98 is a 60-seconds simulation of the broadcasting of the digital radio content illustrated in Code 54 based on a 32kbits bandwidth. The mapping between DMB-ML input data and the simulation snapshot can be done by looking at `es-id` identifiers reported on the left side of the simulation graph. Thus, it can be noticed that `live` updates (on stream `es-id="4"`) have always been delivered on time thanks to a 500ms prefetching. Additionally, asynchronous services have all been delivered on time. However, quite an important time interval can be noticed between the end of first transmission of asynchronous images and their second carrouseling loop. A DMB multiplexer would automatically reduce this carrouseling period but additional enhancement layers from the asynchronous service could also be introduced to fill in these gaps. These possible enhancements are referenced in Figure 88 as unplanned content.

In Figure 99, the carrousel duration of asynchronous images (except for the stream `es-id="305"`) has been reduced to 30s. It can be noticed that this increase of the amount of asynchronous data has consequences for the traffic information service (on stream `es-id="305"`) which exceed its initial 60s carrousel time. In the same way, one of the images of the canvas scene (on stream `es-id="202"`) is regularly delayed. Such slight delays (few seconds) are still acceptable since the MPEG-2 TS simulator does not exactly mirror a real DMB multiplexer.

The two simulations displayed in Figure 98 and Figure 99 show a *live* BIFS service that does not include any image (no image data is transmitted on `es-id="401"`). In practice, images consume a large portion of the multimedia bandwidth since an efficient compression of scene elements can be performed. In the following, the three *live* BIFS updates (at 7s, 40s and 53s) on stream `es-id="4"` are all referencing a single image which is carried on a stream reserved to live images (on stream `es-id="401"`). In order to take into account image transmission delays, a prefetch time of 10s has been defined. In Figure 100, it can be seen that the simplistic linear multiplexing algorithm which consists in regularly broadcasting an MPEG-2 TS packet for each stream do not provide satisfactory hints for *live* services. Indeed, all images are late and the broadcasting of the second image was even cancelled in order to be able to start the broadcasting the third image whereas all asynchronous images (with a lower priority) were transmitted on time. In practice, the multiplexing strategy of a DMB multiplexer is a confidential algorithm that cannot be controlled by the production chain. The MPEG-2 TS simulator implements an optimized algorithm which combines carrousel duration and stream priorities. The output of this multiplexing algorithm for live images can be visualized in Figure 101. In that case, all carrousel durations have been respected thanks to stream priorities; the scalable multimedia service has been delivered to the user according to broadcaster's wishes.

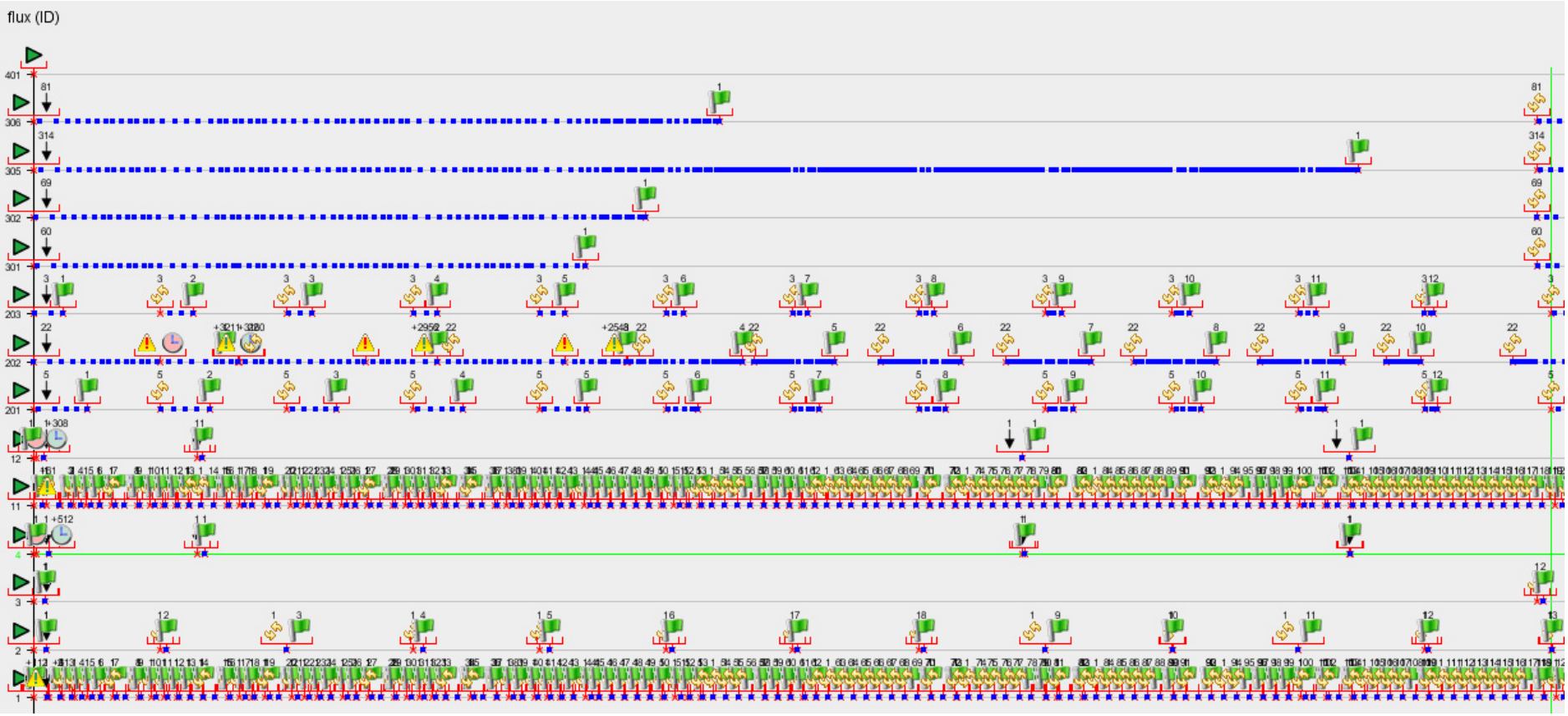


Figure 98: Capture of a sub-optimal carrouseling of asynchronous services (60s carourel).

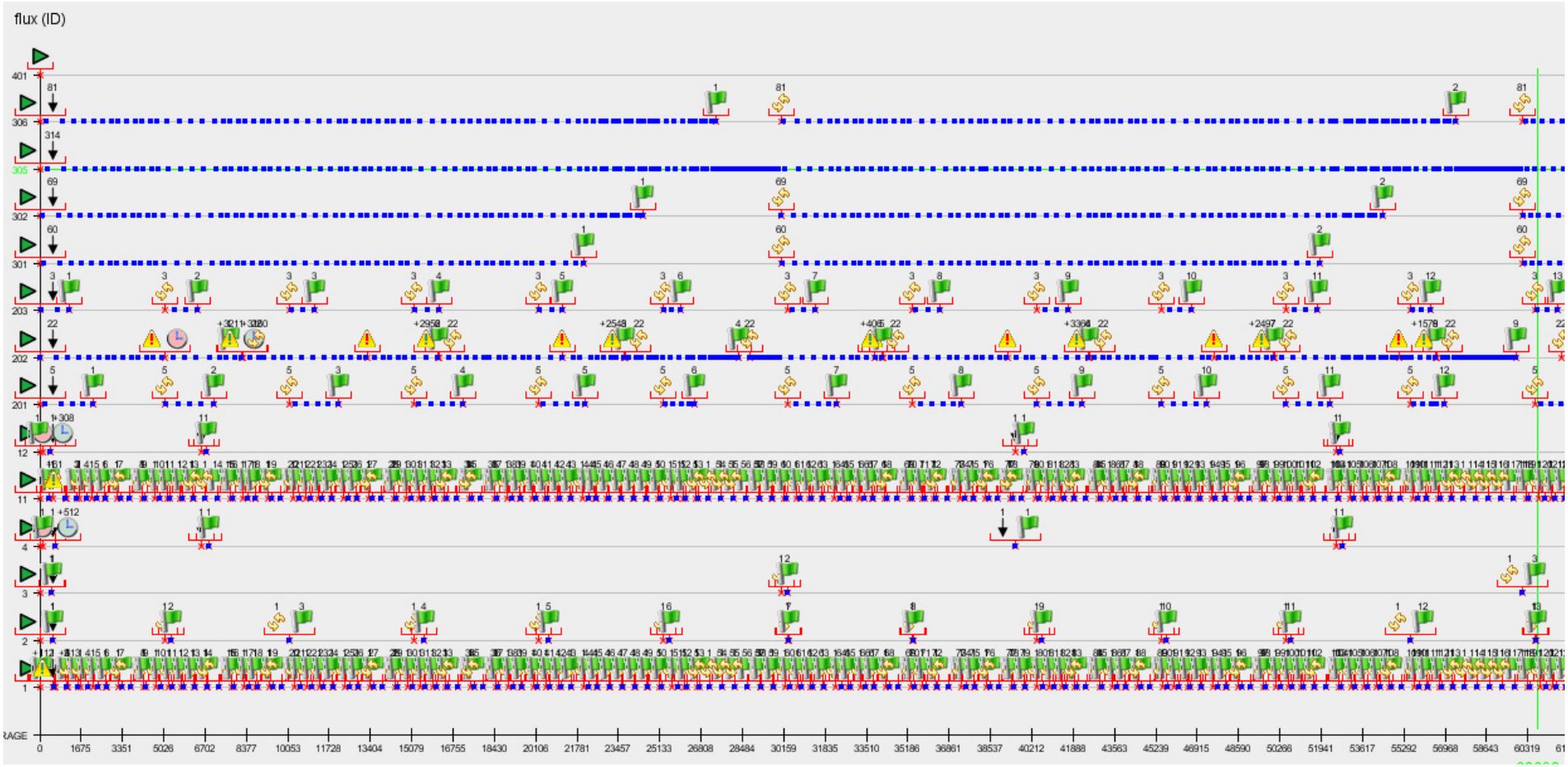


Figure 99: Capture of a short carrousel of asynchronous services (30 s carousel).

Figure 100: Capture of a linear multiplexing algorithm with live images.

Figure 101: Capture of an optimized multiplexing algorithm based on stream priorities.

List of Figures

List of Codes

List of Tables

Bibliography

[16] Allen J., Maintaining Knowledge about Temporal Intervals, In Communications of the ACM, vol. 26, no. 11, 1983, pp. 832-843,.

[17] Badros G.J. et al., A Constraint Extension to Scalable Vector Graphics, In proc. of the International World Wide Web Conference, Hong Kong, 2001, pp. 489-498.

[18] Benbernou S., Makhoul A., Hacid M.S. and Mostefaoui A., A Spatio-Temporal Adaptation Model for Multimedia Presentations, In proc. of the IEEE International Symposium on Multimedia (ISM), Irvine, California, December 2005, pp. 143-150.

[19] Bes F., Jourdan M. and Khantache F., A Generic Architecture for Automated Construction of Multimedia Presentation, In proc. of the conference on Multimedia Modeling (MMM), Amsterdam, Holland, 2001, pp. 229-246.

[20] Bertino E., Ferrari E. and Stolf M., MPGS: An Interactive Tool for the Specification and Generation of Multimedia Presentations, In IEEE Transactions On Knowledge And Data Engineering vol. 12, no. 1, 2001, pp. 102-125.

[21] Bilasco I.M., Modèles de presentations multimédias adaptables, DEA, 2003.

[22] Bilasco I.M., Gensel J. and Villanova-Oliver M., STAMP: a Model for Generating Adaptable Multimedia Presentations, In Multimedia Tools and Applications (MTAP), vol. 25, no. 3, 2005, Special Issue on Metadata and Adaptatbility in Web-based Information Systems, pp. 361-375.

[23] Boll S. and Klas W., ZYX – A Multimedia Document Model for Reuse and Adaptation, In IEEE Transactions on Knowledge and Data Engineering, vol. 13, no. 3, 2001, pp. 361-382.

[24] Boll S., Towards Flexible Multimedia Document Models for Reuse and Adaptation, PhD Thesis, 2001.

[25] Boll S., Klas W. and Westermann U., Multimedia Document Models: Sealed Fate or Setting Out for New Shores?, In Multimedia Tools and Applications, vol. 11, no. 3, 2000, pp. 267-279.

[26] Borade S., Nakiboglu B., Lizhong Z., Unequal Error Protection: some fundamental limits and optimal strategies, In proc. of the International Symposium on Information Theory, Toronto, July 2008, pp. 2222-2226.

[27] Buchanan M.C. and Zellweger P.T., Automatic Temporal Layout Mechanisms Revisited, In ACM Transactions on Multimedia Computing, Communications and Applications, vol. 1, no. 1, 2005, pp. 60-88.

[28] Chen G. and Kotz D., A Survey of Context-Aware Mobile Computing Research, Technical Report, UMI Order Number: TR2000-381, Dartmouth College, 2000.

[29] Concolato C., Description de Scène Multimedia: Representations et Optimizations, PhD Thesis, 2007.

[30] Crease M, Gray P.D. and Brewster S.A., A Toolkit of Mechanism and Context Independent Widgets, In proc. of the workshop design, specification and verification of interactive systems, (VSVIS), 2000, pp. 121-133.

[31] Randell D.A., Cui Z. and Cohn A., A Spatial Logic based on Regions and Connection, In proc. of the international conference on knowledge representation and reasoning, Morgan Kayfmann, 1992, pp. 165-176.

[32] Dey A., Providing architectural support for building context-aware applications, PhD Thesis, 2000.

[33] Dey A.K. and Abowd G.D., Towards a better understanding of context and context-awareness, In proc. of the international symposium on Handheld and Ubiquitous Computing, 1999, pp. 304-307.

[34] De Cock J., Notebaert S., and Van de Walle R., Transcoding from H.264/AVC to SVC with CGS layers, in proc. of the IEEE International Conference on Image Processing, September 2007, pp. 73-76.

[35] Devillers S., Timmerer C., Heuer J. and Hellwagner H., Bitstream Syntax Description-based Adaptation in Streaming and Constrained Environments, In IEEE Transactions on Multimedia, vol. 7, no.3, 2005, pp. 463-470.
[36] Dragicevic P., Chatty S., Thevenin D. and Vinot J-L., Artistic Resizing: A Technique for Rich Scale-Sensitive Vector Graphics, In proc. of the annual ACM symposium on user interface software and technology, 2005, pp. 201-210
[37] EBU – Recommendation R 126, Digital Radio broadcasting: Common European Digital Radio Profiles.
[38] ETSI TS 102 428 V1.2.1 (2009-04), Digital Audio Broadcasting (DAB); DMB video service; User application specification.
[39] ETSI TS 102 563 V1.1.1 (2007-02), Digital Audio Broadcasting (DAB); Transport of Advanced Audio Coding (AAC) audio.
[40] ETSI TS 102 796 V1.1.1 (2010-06), Hybrid Broadcast Broadband TV.
[41] ETSI EN 300 401 V1.4.1 (2006-06), Radio Broadcasting Systems; Digital Audio Broadcasting (DAB) to mobile, portable and fixed receivers.
[42] ETSI EN 300 744 V1.6.1 (2009-01), Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television.
[43] ETSI EN 301 234 V2.1.1 (2006-06), Digital Audio Broadcasting (DAB); Multimedia Object Transfer (MOT) protocol.
[44] ETSI EN 302 304 V1.1.1 (2004-11), Digital Video Broadcasting (DVB); Transmission System for Handheld Terminals (DVB-H).
[45] Euzenat J., Layaïda N. and Diaz V., A semantic framework for multimedia document adaptation, In proc. of the International Joint Conference on Artificial Intelligence (IJCAI), August 2003, Acapulco, Mexico, pp. 9-16.
[46] Freeman-Benson B., Maloney J. and Borning A., An incremental Constraint Solver, In Communications of the ACM, vol. 33, no. 1, January 1990, pp. 54-63
[47] Garlatti S. and Iksal S., Declarative Specifications for Adaptive Hypermedia Based on a Semantic Web Approach, In Lecture Notes in Computer Science, 2003, Volume 2702/2003.
[48] Hardman L., Modelling and Authoring Hypermedia Documents, PhD Thesis, 1998.
[49] Hardman L, Van Rossum G., Jansen J. and Mullender S., CMIFed: A Transportable Hypermedia Authoring System, In proc. of the ACM International conference on Multimedia, New-York, USA, 1994, pp. 471-472.
[50] Hu J., Shuo W., Nian-Wei C. and Zhen Y., Finite State Machine for Automatic GUI Testing, In the International Conference on Computational Intelligence and Software Engineering, (CiSE), December 2009, Wuhan, China, pp.1-4
[51] IETF RFC 3550 (2003), RTP: A Transport Protocol for Real-Time Applications, http://www.ietf.org/rfc/rfc3550.txt.
[52] IETF RFC 4566 (2006), SDP: Session Description Protocol, http://www.ietf.org/rfc/rfc4566.txt.
[53] ISO/IEC 8859-1:1998, Information technology -- 8-bit single-byte coded graphic character sets -- Part 1: Latin alphabet No. 1.
[54] ISO/IEC 13818-1:2007, Information technology -- Generic coding of moving pictures and associated audio information: Systems.
[55] ISO/IEC 14772-1:1997, Information technology -- Computer graphics and image processing - The Virtual Reality Modeling Language (VRML) -- Part 1: Functional specification and UTF-8 encoding.
[56] ISO/IEC 14496-1:2004, Information technology -- Coding of audio-visual objects -- Part 1: Systems.
[57] ISO/IEC 14496-3:2005, Information technology -- Coding of audio-visual objects -- Part 3: Audio.
[58] ISO/IEC 14496-10:2008, Information technology -- Coding of audio-visual objects -- Part 10: Advanced Video Coding.
[59] ISO/IEC 14496-11:2005, Information technology -- Coding of audio-visual objects -- Part 11: Scene description and application engine.
[60] ISO/IEC 14496-11:2005/Amd 7:2010, Information technology -- Coding of audio-visual objects -- Part 11: Scene description and application engine, AMENDMENT 7: ExtendedCore2D profile.

[61] ISO/IEC 14496-14:2003, Information technology -- Coding of audio-visual objects -- Part 14: MP4 file format.
[62] ISO/IEC 14496-20:2008, Information technology -- Coding of audio-visual objects -- Part 20: Lightweigh Application Scene Representation (LASeR) and Simple Aggregation Format (SAF).
[63] ISO/IEC 14496-22:2009, Information technology -- Coding of audio-visual objects -- Part 22: Open Font Format.
[64] ISO/IEC DIS 16262:2010, Information technology -- ECMAScript language specification (Standard ECMA-262 5th edition).
[65] ISO/IEC 21000-3:2003, Information technology -- Multimedia framework (MPEG-21) -- Part 3: Digital Item Identification.
[66] ISO/IEC 21000-7:2007, Information technology -- Multimedia framework (MPEG-21) -- Part 7: Digital Item Adaptation.
[67] ISO/IEC 21000:7:2004/Amd1:2006, Information technology -- Multimedia framework (MPEG-21) -- Part 7: Digital Item Adaptation, AMENDMENT 1: DIA Conversions and Permissions.
[68] ISO/IEC 21000-10:2006, Information technology -- Multimedia framework (MPEG-21) -- Part 10: Digital Item Processing.
[69] ISO/IEC 23003-1:2007, Information technology -- MPEG audio technologies -- Part 1: MPEG Surround.
[70] Jacobs C., Li W., Schrier E., Bargeron B. and Salesin D., Adaptive Grid-based Document Layout, In Communications of the ACM, vol. 47, no. 8, 2004, pp. 838-847.
[71] Jourdan M., Roisin C. and Tardif L., Constraints Techniques for Authoring Multimedia Documents, Constraints Journal, Kluwer Academic Publishers, vol. 6, no. 1, January 2001, p.115-132
[72] Jourdan M. and Bes F., A new Step Towards Multimedia Documents Generation, In proc. of the International Conference on Media Futures, Florence, Italy, May 2001, pp. 25-28.
[73] Karczewicz M. and Kurceren R., The SP- and SI-frames design for H.264/AVC, In IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, 2003, pp. 637-644,.
[74] Kimiaei-Asadi M. and Dufourd J.C., Multimedia Adaptation By Transmoding in MPEG-21, in proc. of the International Workshop On Image Analysis for Multimedia Interactive Services, Lisbon, Portugal, April 2004.
[75] Kimiaei Asadi M., Adaptation de contenu multimedia avec MPEG-21: conversion de resources et adaptation sémantique de scenes, PhD Thesis, 2005.
[76] Kimiaei-Asadi M. and Dufourd J.C., Context-Aware Semantic Adaptation of Multimedia Presentations, In proc. of the International Conférence on Multimedia & Expo, Amsterdam, Holland, July 2005.
[77] Kofler I., Seidl J., Timmerer C., Hellwagner H., Djama I. and Ahmed T., Using MPEG-21 for cross-layer multimedia content adaptation, In Signal, Image and Video Processing, vol. 2, no. 4, 2008, pp.355-370
[78] Laborie S., Adaptation Sémantique de Documents Multimedia, PhD Thesis, 2008.
[79] Layaïda N., Madeus: Système d'Edition et de Présentation de Documents Structurés Multimedia, PhD Thesis, 1997.
[80] Lee Y.-K., Lee S.-S. and Lee Y.-L., MPEG-4 to H.264 Transcoding using Macroblock Statistics, In the IEEE International Conference on Multimedia and Expo, Toronto, Ont., 2006, pp. 57 - 60
[81] Le Feuvre J., Concolato C. and Moissinac J.C., GPAC, Open Source Multimedia Framework, ACM Multimedia, Augsburg, Germany, September 2007.
[82] Lemlouma T., Architecture de Négocation et d'Adaptation de Services Multimedia dans des environnements hétérogènes, PhD Thesis, 2004.
[83] Li C-S., Mohan R., Smith J.R., Multimedia Content Description in the InfoPyramid, In proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing, vol.6, Seattle, USA, May 1998, pp. 3789 - 3792.
[84] McCormack C., Marriott K. and Meyer B., Authoring Adaptive Diagrams. In proc. of the ACM symposium on Document engineering, Sao Paulo, Brazil, 2009, pp. 154-163.
[85] Marcellin M.W., Bilgin A., Gormish M.J. and Boliek M.P, An Overview of JPEG-2000, In proc. of the Conference on Data Compression. DCC. IEEE Computer Society, Washington, DC, 523, March 2000.

[86] Marriott K., Meyer B. and Tardif L, Fast and Efficient Client-side Adaptivity for SVG, In proc. of the international conference on World Wide Web, Honolulu, Hawaii, USA, 2002, pp. 496-507
[87] Meltzer S. and Moser G., MPEG-4 HE-AAC v2 - Audio coding for today's digital media world, EBU Technical Review, January 2006.
[88] Merkel K., HbbTV— A Hybrid Broadcast-Broadband System for the living room, EBU technical review - 2010 Q1.
[89] Metso M., Koivisto A. and Sauvola J., A Content Model for the Mobile Adaptation of Multimedia Information, Journal of VLSI Signal Processing, vol 29, issue 1-2, 2001, pp. 115-128
[90] Mickley Gillenwater Z., Flexible Web Design: Creating Liquid and Elastic Layouts with CSS, New riders, ISBN-10: 0321553845, 2008.
[91] Mukherjee D., Delfosse E., Kim J.-G. and Wang Y., Optimal Adaptation Decision-taking for Terminal and Network Quality of Service, In IEEE transactions on multimedia, vol. 7, no. 3, June 2005.
[92] Newcomb S.R., Multimedia Interchange using SGML/HyTime, Part 1: Structures. In IEEE MultiMedia, vol. 2, no. 3, June 1995, pp. 86-89.
[93] Nigay L., Conception et Modélisation Logicielles des Systèmes Interactifs: application aux interface multimodales, PhD Thesis, 1994.
[94] Pimentel M., Bulterman D. and Soares L., Document Engineering Approaches toward Scalable and Structured multimedia, web and printable documents, Multimedia Tools and Applications, vol. 43, issue 3, 2009, pp 195-202.
[95] Plesca C., Charvillat V., Grigoras R., Adapting Content Delivery to Limited Resources and Inferred User Interest, In International Journal of Digital Multimedia Broadcasting, 2008.
[96] Plesca C., Supervision de Contenus Multimedia: adaptation de contenu, politiques optimales de préchargement et coordination causale de flux, PhD Thesis, 2007.
[97] Ransburg M., Codec-agnostic Dynamic and Distributed Adaptation of Scalable Media, VDM Verlag, Dr. Müller, March 2009, pp. 11-12.
[98] Ransburg M., Timmerer C., and Hellwagner H., Dynamic and Distributed Multimedia Content Adaptation based on the MPEG-21 Multimedia Framework. In Multimedia Semantics - The Role of Metadata, vol.101, 2008, pp. 3-23.
[99] Roisin C., Documents structurés multimédia, Habilitation à diriger des recherches, 1999.
[100] Rönnau S., Philipp G., and Borghoff U.M.. Efficient change control of XML documents. In proc. of the ACM Symposium on Document Engineering, Munich, Germany, September 2009, pp. 3-12.
[101] Scherp A., A Component Framework for Personalized Multimedia Applications, PhD Thesis, 2006.
[102] Schierl T., Stockhammer T. and Wiegand T., Mobile Video Transmission using Scalable Video Coding, In IEEE transactions on circuits and systems for video technology, vol. 17, no. 9, September 2007, pp. 1204-1217.
[103] Schneider K. and Cordy J., Abstract User Interfaces: A Model and Notation to Support Plasticity in Interactive Systems, Interactive Systems: Design, Specification, and Verification, vol. 2220, Springer Berlin / Heidelberg, 2001, pp. 28-48
[104] Schwarz, H., Marpe, D., and Wiegand, T., Overview of the Scalable Video Coding Extension of the H.264/AVC Standard, In IEEE Transactions on Circuits and Systems for Video Technology, vol.17, no.9, September 2007, pp.1103-1120.
[105] Soares L.F.G. and Rodrigues R.F., Nested Context Language (NCL 3.0) - Part 11: Declarative objects in NCL, Technical Report, Departamento de Informática da PUC-Rio, MCC 12/09, http://www.ncl.org.br/documentos/NCL3.0-DO.pdf.
[106] Stockhammer T., Hannuksela M.M. and Wenger S., H.26L/JVT Coding Network Abstraction Layer and IP-based Transport, In proc. of the International Conference on Image Processing, vol. 2, 2002, pp. 485-488.
[107] Sutton R.S. and Barto A.G., Reinforcement Learning: An Introduction, MIT Press, ISBN 0-262-19398-1, 1999.
[108] Thang T.C., Jung Y.J. and Ro Y.M., Modality Conversion for QoS Management in Universal Multimedia Access, In IEEE Proceedings: Vision, Image & Signal Processing, vol. 152, no. 3, June 2005, pp.374-384.
[109] Thevenin D., Adaptation en Interaction Homme-Machine: le cas de la plasticité, PhD Thesis, 2001.

[110] Vander Zanden B., Halterman R., Myers B.A., McDaniel R., Miller R., Szekely, Giuse D.A. and Kosbie D., In Lessons learned about one-way, dataflow constraints in the garnet and amulet graphical toolkits, In ACM Transactions On Programming Languages and Systems., vol. 23, no. 6, 2001, pp.776-796.
[111] Vaudry C. et al., Initiative Mixte dans les DVP: de la Pertinence à l'Adaptation, Actes de DVP 2002, Brest, July 2002, pp. 141-155.
[112] Villard L. and Layaïda N, An Incremental XSLT Transformation Processor for XML Document Manipulation, In proc. of the Internation World Wide Web Conference, Honolulu, Hawaii, USA, May 2002, pp. 474-485.
[113] Villard L., Modèles de Documents pour l'Edition et l'Adaptation de Presentations Multimédias, PhD Thesis, 2002.
[114] Vitali F. and Durand D., Using Versioning to Support Collaboration on the WWW, World Wide Web Journal, 1(1), O'Reilly, 1994, pp. 37-50.
[115] Wahl T., Wirag S. and Rothermel K., TIEMPO: Temporal Modeling and Authoring of Interactive Multimedia, In proc. of the International Conference on Multimedia Computing and System, 1995, pp. 274-277.
[116] W3C Recommendation (2008), Extensible Markup Language (XML) 1.0 (Fifth Edition) http://www.w3.org/TR/xml/.
[117] W3C Recommandation (2004), Resource Description Framework (RDF): Concepts and Abstract Syntax, http://www.w3.org/TR/rdf-concepts/.
[118] W3C Recommendation (2008), Scalable Vector Graphics (SVG) Tiny 1.2 Specification http://www.w3.org/TR/SVGMobile12/.
[119] W3C Recommendation (2008), Synchronized Multimedia Integration Language (SMIL 3.0) http://www.w3.org/TR/SMIL3/.
[120] W3C Working Draft (2010), the Hypertext Markup Language (HTML 5.0) http://www.w3.org/TR/html5/.
[121] W3C Recommendation (2010), Media Queries, http://www.w3.org/TR/css3-mediaqueries/.
[122] W3C Recommendation (1999), XSL Transformations (XSLT) Version 1.0 http://www.w3.org/TR/xslt/.
[123] W3C Candidate Recommendation (2009), Cascading Style Sheets (CCS 2.1) Specification http://www.w3.org/TR/CSS2/.
[124] W3C Working Draft (2006), Remote Events for XML (REX) 1.0, http://www.w3.org/TR/rex/.
[125] W3C Recommendation (2004), Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0, http://www.w3.org/TR/CCPP-struct-vocab/.
[126] W3C Recommendation (2008), Web Content Accessibility Guidelines (WCAG) 2.0 http://www.w3.org/TR/WCAG20/.
[127] W3C Working Draft (2009), SMIL Timesheet 1.0, http://www.w3.org/TR/timesheets/.
[128] W3C Recommendation (2005), xml:id Version 1.0, http://www.w3.org/TR/xml-id/.
[129] W3C Recommendation (2007), XML Path Language (XPath) 2.0 http://www.w3.org/TR/xpath20/.
[130] W3C Recommandation (2000), Document Object Model (DOM) Level 2 Events Specification http://www.w3.org/TR/DOM-Level-2-Events/.
[131] W3C Candidate Recommendation (2010), XMLHttpRequest http://www.w3.org/TR/XMLHttpRequest/.
[132] Watters C. and Shepherd M., Research Issues for Virtual Documents, In proc. of the Workshop on Virtual Documents Hypertext Functionality and the Web at the 8th International World Wide Web Conference, May 1999, Toronto, Canada, pp. 109-128.
[133] Wenger S., Wang Y-K. and Schierl T., Transport and Signaling of SVC in IP Networks, In IEEE transactions on circuits and systems for video technology, vol. 17, no. 9, September 2007, pp. 1164-1173.
[134] Vetro A., Christopoulos C., and Ebrahimi T., From the guest editors – Universal multimedia access, IEEE Signal Processing Magazine, vol. 20, no. 2, March 2003, pp. 16-16.
[135] Yang S., Zhang J., Chen R., and Shao N., A unit of information-based content adaptation method for improving web content accessibility in the mobile Internet, In ETRI Journal, vol. 29, no. 6, Secember 2007, pp. 794-807.