Continuité des services multimédias dans le Web

5.1 Motivation

Nous avons souhaité continuer notre étude sur les services multimédias mais en élargissant nos recherches au-delà du cadre IMS en nous intéressant en particulier au Web où les fournisseurs de contenus multimédias connaissent un très grand succès. L'architecture de télécommunications était un point de départ intéressant pour cette étude avec une cadre standardisé et de nombreux mécanismes assurant des fonctions essentielles telles que l'identification, le déclenchement (automatique notamment) ou encore la communication entre le client (IMS) et l'infrastructure. Cependant elle imposait aussi de nombreuses contraintes, du point de vue de la norme stricte mais également au niveau des terminaux mobiles à faible capacité et des clients peu flexibles d'où une réelle difficulté dans la mise en œuvre de modes manuels de transfert, plus proches de l'utilisateur.

De très nombreux sites proposant la diffusion de flux multimédias (*streaming*) ont émergé depuis le début de ces travaux, prouvant l'engouement des utilisateurs pour ce type de services. L'environnement Web offre en effet de nombreuses opportunités : ouverture totale, aucune limitation au niveau client, richesse des services, ... avec en contrepartie la coexistence d'un grand nombre de solutions propriétaires présentant des difficultés particulières d'interopérabilité pour la conception d'une solution globale. Cette approche Web nous permet cependant d'aborder

la mobilité sous un autre angle, d'étudier plus en détail le mode manuel mais aussi un autre genre de continuité offrant une meilleure expérience utilisateur.

5.2 Problématiques et approche

Le cadre de travail change radicalement par rapport aux deux chapitres précédents, nous nous dégageons totalement des contraintes du monde télécom et de l'IMS en particulier. Les limitations de performances intrinsèques aux terminaux mobiles ne sont plus applicables, nous nous intéresserons d'ailleurs uniquement à des terminaux de type « ordinateur » dotés, en comparaison, de ressources illimitées. En conséquence, nous nous focaliserons sur le cas de *terminal handover*, le changement de réseau étant peu probable dans ces conditions ; c'est ici l'utilisateur qui crée la mobilité en décidant de changer d'interface (de terminal) pour son service multimédia.

L'infrastructure IMS assure de nombreuses fonctions qui font maintenant défaut au niveau de l'identification, du déclenchement et des interfaces client-serveur ; il est nécessaire de les repenser dans le cadre Web. Cependant, le *Media Proxy* introduit dans le chapitre précédent (cf. 4.3.2) est externe à l'infrastructure télécom, nous l'avons donc naturellement réutilisé dans le cadre de cette étude. En effet, nous avons défini plusieurs fonctions de traçage, d'identification et de transfert applicables également aux services multimédias Web. Ce composant, en tant que proxy entre le lecteur média et le serveur, est capable de collecter les informations clés des messages de signalisation et des flux qui le traversent tel que l'URL, l'adresse IP ou encore la position courante du média. Il est cependant nécessaire de l'adapter au monde Web constitué de nombreuses solutions propriétaires. Les traceurs RTSP et MMS déjà définis doivent être complétés par des analyseurs spécifiques pour chaque acteur Web de services multimédias. Une gestion exhaustive étant impossible, nous avons adressé les principaux acteurs du moment, à savoir : *Youtube*, *Dailymotion* et *Google Video*. À noter que le support des protocoles MMS et RTSP permet déjà de gérer un grand nombre de sites tels que ceux des principales chaînes de télévision qui diffusent généralement en RTSP.

5.2.1 Déclenchement

Le déclenchement ne peut plus être réalisé via une communication standard entre le client et l'infrastructure comme dans les approches précédentes. De plus, l'absence de mécanismes de présence et d'identification tels qu'ils existent dans l'IMS limite les scénarios automatiques où le transfert était initié en fonction de l'état de présence (ou d'enregistrement) des terminaux. Ici les terminaux sont simplement équipés d'un client HTTP (ou navigateur Web) et d'un lecteur multimédia. Ce dernier peut prendre diverses formes : logiciel dédié, plug-in du navigateur ou encore une application Flash téléchargée à la demande.

L'infrastructure est ici le Web et le client est le navigateur, il n'existe évidemment aucun mécanisme spécifique prévu au transfert et à la continuité des services multimédias accessibles sur le Web. Il est donc nécessaire de créer une nouvelle interface entre le Web et l'utilisateur pour orchestrer le transfert de ces services. De plus, l'absence de mécanisme d'enregistrement et de présence nous oriente plus facilement vers un mode de déclenchement manuel, entière géré par l'utilisateur, ce qui soulève plusieurs interrogations.

5.2.2 Identification

Comment identifier au niveau Web/proxy le propriétaire d'une session multimédia, en d'autres termes comment associer un utilisateur à un flux ? En effet, outre les mécanismes d'authentification nécessaires pour préserver l'utilisateur et ses services, il faut également identifier toutes les composantes du système pour permettre un mode manuel : terminaux, utilisateurs et services. Si le service peut être identifié par l'URL et le terminal par son adresse IP, il reste encore l'utilisateur. Enfin, le service n'est pas la même chose que la session (le flux) qui en est une instance liée à un terminal (ou à l'utilisateur). Nous verrons par la suite que nous adoptons une approche orientée Web qui ne nécessite pas d'identifier ni de désigner les sessions.

Une autre question se pose : comment l'utilisateur désigne une session, celle-ci n'existant qu'au travers d'une URL qui est généralement particulièrement incompréhensible pour l'utilisateur. Au niveau signalisation, seule l'URL est perçue, les informations contenues dans les pages Web n'étant pas exploitables de par leur incomplétude et leur volatilité ; seul l'utilisateur peut donner du sens à ces médias.

Bien qu'il ne regarde généralement qu'un média à la fois et que la désignation peut sembler inutile, nous verrons dans la section suivante qu'il devient nécessaire d'apporter du sens aux noms de sessions dans un environnement Web. Si ces problèmes existaient déjà dans les approches précédentes elles étaient résolues en partie grâce à des mécanismes de déclenchement automatiques ou semi-manuel.

5.2.3 Style Web 2.0

Cette étude porte sur un nouvel environnement et il est nécessaire de concevoir des solutions de mobilité qui s'adaptent parfaitement à ce nouveau cadre pour une expérience utilisateur optimale. Le Web s'est récemment approché d'un modèle très interactif, collaboratif, faisant de chaque consommateur de contenu un producteur potentiel, l'ensemble des technologies et l'état d'esprit même de cette approche sont usuellement qualifiés de « Web 2.0 » [86]. Nous avons cherché à insuffler aux mécanismes de continuité définis dans les deux chapitres précédents, relativement « rigides », cet esprit 2.0 en favorisant la communication, la collaboration et l'intégration dans l'environnement Web.

Nous avons recherché un mode de continuité différent, qui privilégie la continuité contextuelle à la continuité temporelle, étudier par exemple une mobilité asynchrone des services multimédias, tel un livre que l'on arrête de lire puis que l'on reprend plus tard au même endroit grâce à un marque-page. De nouvelles possibilités s'offrent alors, « marquer » les flux médias, les enrichir avec des métadonnées (*metadata*), les partager, ouvrir à d'autres applications, etc.

Ainsi est née la notion de *media bookmark*. Un *media bookmark* est un repère dans un service multimédia qui permet à l'utilisateur d'arrêter la lecture et de le reprendre au même endroit dans le flux depuis n'importe quel terminal, et pourquoi pas sur celui d'un autre utilisateur. Ce mode change radicalement de ceux plus classiques présentés jusqu'alors, il respecte pourtant la notion de continuité, de mobilité (*terminal handover*), seule la synchronisation n'est pas assurée ni recherchée d'ailleurs par l'utilisateur.

5.3 Implémentation

Nous avons implémenté un prototype composé en plusieurs parties. Le client est réalisé sur *Google desktop* [87], un environnement logiciel où peuvent être déployés des plug-ins (*Google Gadgets*) : des applications qui s'intègrent ainsi au bureau du système d'exploitation du terminal utilisateur. Cet environnement a été choisi car il présente une solution orientée Web qui offre un cadre permettant le développement et l'implémentation rapide d'applications. Le client que nous avons développé ici s'appelle *Google Gadget Media* (GGMedia), supposé toujours actif sur le terminal, le *Google desktop* étant généralement exécuté automatiquement au démarrage du SE. Le client se connecte à un serveur Web, le *Bookmark Server*, lui même associé au *Media Proxy* que nous ne présentons plus (cf. 4.3.2) très légèrement amélioré.

5.3.1 Scénario

Les scénarios sont multiples, voici un exemple d'utilisation orienté Web de notre solution, nous détaillerons l'architecture et les mécanismes par la suite.

Lionel navigue sur le Web et décide de regarder le résumé des évènements sportifs de la semaine retransmis sur le site officiel d'une grande chaîne de télévision. Il clique sur le lien du flux et le lecteur multimédia embarqué dans son navigateur joue la vidéo. Une notification provenant de son GGMedia apparaît alors, lui indiquant qu'une session a été détectée. Il doit cependant se déconnecter mais souhaite mémoriser la position de lecture courante, il affiche alors son GGMedia (localisé dans la barre des tâches *Google desktop*) et double-clique sur la session qui s'ajoute à sa liste de *bookmarks*.

Plus tard, il rallume son ordinateur, affiche le GGMedia qui liste l'ensemble des ses bookmarks multimédias dont celui du résumé sportif. Il double-clique dessus et son lecteur multimédia démarre, affichant le flux vidéo à l'endroit même où il l'avait laissé.

Enfin, tellement enthousiasmé par l'exploit d'un coureur français, il décide de donner au bookmark un nom plus explicite (« Djhone qualifié ! ») et y ajoute un petit commentaire « *La course fabuleuse.* », enfin il rend le bookmark public (via son GGMedia) afin de le partager avec ses amis. Ces derniers verront alors le nou-

veau bookmark de Lionel et pourront le contacter d'un simple clic, les informations de contact étant incluse dans le bookmark partagé.

La figure 5.1 est une capture d'écran du bureau de Lionel pendant qu'il regarde une vidéo sur le Web. Au milieu, la fenêtre du navigateur est ouverte sur un flux média, on peut remarquer sur la droite la barre *Google desktop* contenant le client GGMedia. Le client affiche la session en cours mais également deux bookmarks que Lionel a précédemment créés et déjà enrichis avec de métadonnées.

Les quelques actions présentées dans ce scénario ne sont qu'un aperçu des possibilités d'une telle solution, qui, grâce aux fonctions offertes par le Web, sont illimitées.

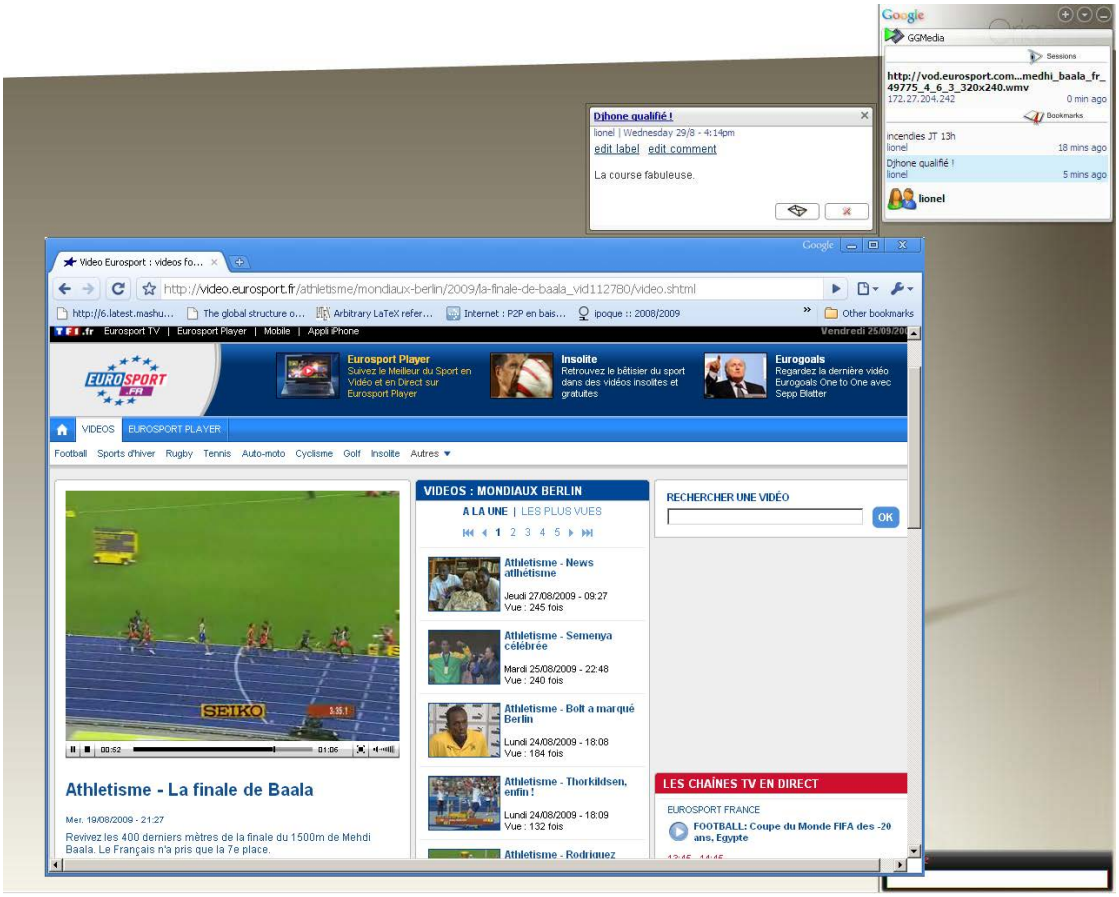

FIGURE 5.1 – Illustration du scénario.

5.3.2 Architecture

Les différents acteurs de ce scénario sont les suivants :

- le terminal de Lionel doté de :
  - un navigateur Web,
  - un lecteur multimédia (application dédiée ou plug-ins),
  - le client GGMedia (dans son environnement *Google desktop*) ;
- le site du fournisseur Web de contenu multimédia (joue le rôle de serveur média),
- le *Media Proxy* positionné entre le client et le site Web,
- le *Bookmark Server*, serveur Web associé au *Media Proxy*.

Nous allons nous intéresser à présent aux mécanismes clés qui entrent en jeu parmi ces acteurs représentés de manière schématique en Figure 5.2.

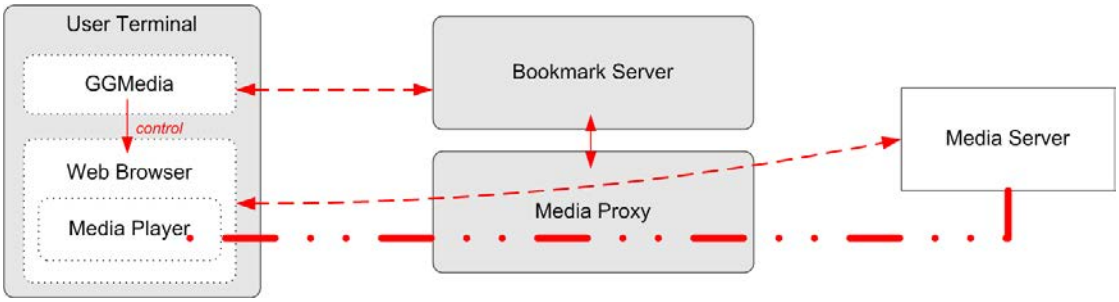

FIGURE 5.2 – Vue globale des différentes composantes du système.

Identification.

Le service de continuité est ici aussi uniquement fourni aux utilisateurs qui y ont souscrit, il est donc nécessaire d'identifier les clients pour assurer un contrôle d'accès permettant la commercialisation du service, la protection de la vie privée ou encore la mise en œuvre de mécanismes adéquats. L'identification est réalisée entre le client GGMedia et le *Bookmark Server* via un secret partagé délivré à la souscription. Des échanges via le protocole HTTP transportant des messages de signalisation propriétaires (nous contrôlons le client et le serveur) permettent au client de communiquer l'adresse IP du terminal qui est alors stockée par le *User Manager* (dans la base de données utilisateurs *User data*) et autorisée au niveau

du proxy, cf. Figure 5.3. Le *Media Proxy* pourra par la suite associer les sessions multimédias destinées à cette adresse IP à l'utilisateur en question.

Une fois le client identifié, l'interface GGMedia-*Bookmark Server* permet le transfert de différentes informations concernant les sessions ou les bookmarks.

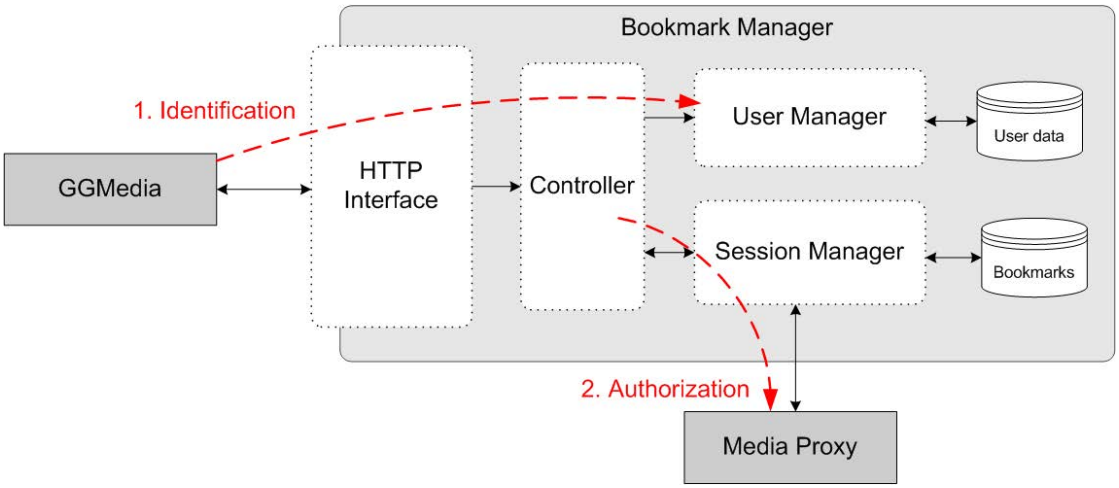

FIGURE 5.3 – Identification du GGMedia dans le *Bookmark Manager*.

Traçage des sessions.

Le *Media Proxy* joue toujours son rôle de proxy transparent entre le client et le serveur de médias. Or ici le client est double, il est à la fois le navigateur Web mais aussi le lecteur multimédia. Il doit donc savoir analyser plusieurs protocoles : HTTP, mais aussi RTSP, MMS, les messages de signalisation propriétaires (lecteurs basés sur la technologie Flash) et enfin les flux médias (RTP). Lorsqu'un flux média est requis par le navigateur Web, les requêtes correspondantes doivent passer par le proxy, il est donc nécessaire que le navigateur soit configuré de manière appropriée (disponible sur toutes les applications de ce type).

Le *Stream Logger* et le *Switch Processor* présentés précédemment (cf. 4.3.3) doivent assurer différentes fonctions spécifiques à chaque protocole afin de capturer les informations permettant un transfert futur. Devant la multitude de solutions multimédias propriétaires présentes dans le Web, la continuité ne pourra être réalisée que pour un nombre limité de protocoles dont les mécanismes sont connus a priori. Nous avons étudié et implémenté les mécanismes de transfert pour les prin-

cipales solutions RTSP, MMS, *Youtube*, *Google Video* et *Dailymotion* afin de couvrir la large majorité des services existants, cependant l'implémentation du proxy a requis une attention particulière afin qu'il se comporte ne manière transparente vis à vis des protocoles non supportés. En effet, si la continuité n'est pas réalisable, il est indispensable que les mécanismes de mobilité restent imperceptibles pour l'expérience utilisateur.

L'étude des protocoles propriétaires de Youtube, Dailymotion et Google Video a requis l'analyse des échanges HTTP entre le client et le serveur. Ainsi c'est par un premier travail de *reverse engineering* qu'il a été possible de retrouver dans les échanges protocolaires la position courante d'un flux ou la manière de démarrer la lecture à une position précise. Maîtriser ces paramètres identifiés dans le chapitre précédent (URL, position, etc) est la condition d'un transfert de session réalisable dans le cas des services multimédias. L'objectif final étant l'évaluation de la faisabilité de la continuité dans un environnement où coexistent de nombreux acteurs hétérogènes (en terme de mécanismes et de signalisation).

Déclenchement.

Le déclenchement du transfert est manuel, asynchrone et réalisé en deux temps. Tout d'abord l'utilisateur marque une session à une position spécifique en créant un bookmark depuis l'interface graphique du GGMedia. Il a alors la possibilité d'ajouter des informations (métadonnées) qu'il considère pertinentes : un titre, des commentaires, des tags, une image ... dans un pur style Web 2.0. À noter que les métadonnées peuvent également être ajoutées par le *Bookmark Manager*, cf. 5.3.2 ; la Figure 5.4 présente un exemple de bookmark enrichi.

La demande de création du marque-page média et les métadonnées sont envoyées vers le *Bookmark Manager* qui récupère l'URL de la session et le *timestamp* correspondant à la position actuelle via le *Media Proxy*, cf. Figure 5.6. Le *Session Manager* stocke ensuite le bookmark ainsi créé et le communique à tous les clients GGMedia afin qu'il soit affiché à l'utilisateur, la Figure 5.5 présente un exemple de message envoyé aux clients décrivant les sessions et les bookmarks de l'utilisateur. Cette première étape marque la session comme « continuable » un peu comme la commande *Switch* dans le chapitre 3, mais cette fois-ci la position de reprise du service est définie a priori. La seconde partie du déclenchement n'est pas néces-

sairement immédiate, c'est l'utilisateur qui décide de l'initier quand il le souhaite. Il peut alors réaliser d'autres actions : marquer de nouveau la session et créer un bookmark supplémentaire, partager le bookmark avec des amis via divers canaux de communication (courrier électronique, réseau social, blog, *Bookmark Manager*, etc). Le bookmark permet ainsi de préparer le transfert mais est également un prétexte d'échange entre utilisateurs.

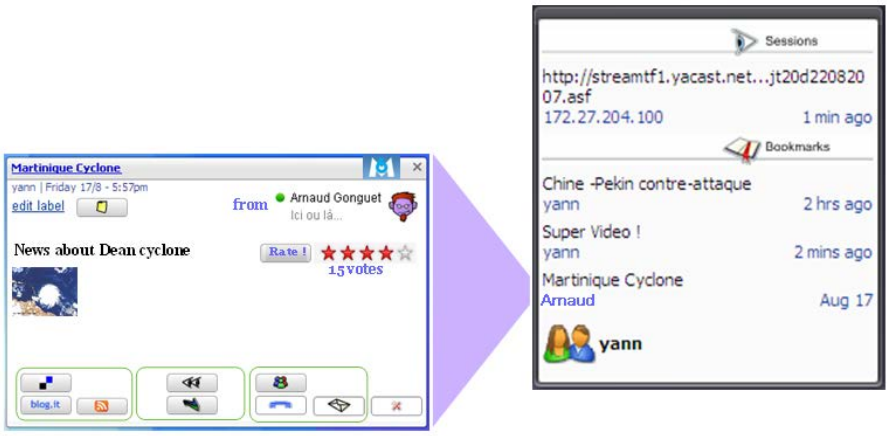

FIGURE 5.4 – Exemple de *media bookmark*

L'utilisateur active la deuxième phase du déclenchement en exécutant un bookmark depuis l'interface graphique de son client GGMedia (cf. Figure 5.7). Le bookmark peut lui appartenir ou avoir été créé et partagé par un tiers. Quoi qu'il en soit, le *Session Controller* exécute l'application correspondant au type d'URL (connu grâce à son préfixe associé par le SE à une application spécifique). Le lecteur multimédia ou le navigateur Web charge directement l'URL pour rétablir le média à la position indiquée dans le bookmark.

Désignation.

Cependant le déclenchement en mode manuel pose ici un problème de désignation des sessions. en effet, celles-ci sont identifiées au niveau du système par leur URL, unique pour un média donné. Mais cette information n'est pas satisfaisante pour l'utilisateur : trop longue et inintelligible. La nomination des bookmarks par l'utilisateur lors de la première phase du déclenchement est une action contraignante mais acceptable dans le cadre des bookmarks qui sont créés occasionnelle-

```
<?xml version="1.0" encoding="utf-8"?>
<userSessions username="Toto">
     <host ip="192.168.1.1">
          <session sid="452EB9856">
               <label>foo-bar</label>
               <description>TF1;Journal Télévisé;20h</description>
               <url proto="none">http://www.foo-bar.com/stream1</url>
               <created>5</created>
          </session>
          <session sid="45FG369">
               <label>foo-bar</label>
               <description>France 2;Stade 2</description>
               <url proto="http">http://www.foo-bar.com/stream2</url>
               <created>20</created
          </session>
     </host>
     <host ip="192.168.1.2">
          <session sid="89T54I">
               <label>foo-bar</label>
               <description>M6;Popstar</description>
               <url proto="mms">http://www.foo-bar.com/stream3</url>
               <created>2</created
          </session>
     </host>
     <bookmark bid="454H85A">
          <author>Bob</author>
          <label>foo-bar</label>
          <url proto="rtsp">http://www.foo-bar.com/stream1</url>
          <begin>458</begin>
          <created>85</created>
```

FIGURE 5.5 – Bookmarks (et sessions) dans un message de signalisation.

ment et ont un intérêt particulier pour l'utilisateur. Les sessions ne peuvent être gérées de la même manière, l'utilisateur serait sollicité à chaque établissement d'un flux, nuisant considérablement à son expérience du service offrant tout sauf de la continuité...

Nous avons proposé un mécanisme qui permet de résoudre ce problème grâce à l'aspect collaboratif de la nomination des bookmarks. Tous les utilisateurs peuvent enrichir leurs bookmarks avec des métadonnées, une manière de les reconnaître et de leur apporter du sens dans la liste qu'ils constituent sur leur GGMedia. Ces informations sont spécifiques à un flux, lui-même identifié par une URL. L'URL correspond donc à un (ou plusieurs) sujet/thème qui apparaîtra naturellement dans les métadonnées des utilisateurs qui marqueront ce flux. Or il se trouve que les URL des médias d'un fournisseur de contenus présentent des parties similaires, correspondant par exemple au diffuseur ou au type de programme. Ainsi, il est possible de donner du sens à ces parties d'URL en détectant les zones communes qui génèrent les mêmes métédonnées, cf. Figure 5.1.

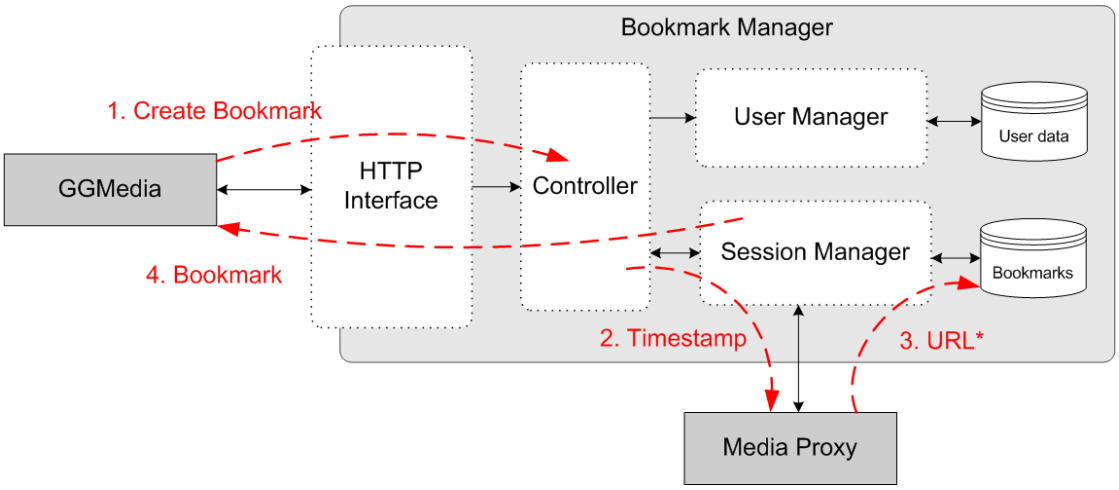

FIGURE 5.6 – Création d'un bookmark.

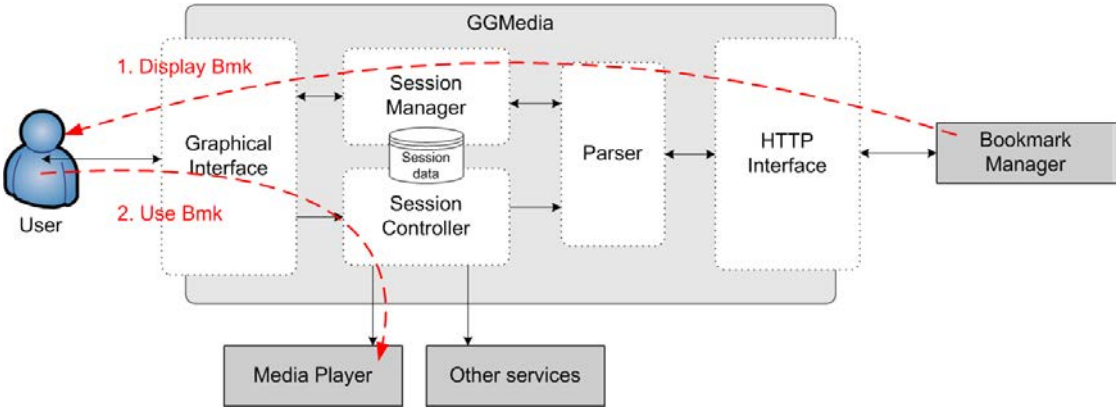

FIGURE 5.7 – Affichage et exécution d'un bookmark.

Ce mécanisme nécessite évidemment une collaboration importante afin de posséder d'une base de données significative qui permette l'identification d'un grand nombre de « blocs » (*patterns*) dans les URL et générer ainsi automatiquement des métadonnées pour les nouvelles sessions.

5.3.3 Mécanisme de transfert

Le mécanisme de handover est très similaire à celui présenté dans le chapitre précédent, le *Media Proxy* qui l'assurait étant également présent ici. Une différence existe cependant pour les nouveaux protocoles propriétaires supportés (Youtube,

TABLE 5.1 – Inférence de métadonnées à partir d'une URL.

URL	Most relevant keywords
rtsp://sdmefr.media.com/jvh/23112006.wmv	TF1, journal, 20h, innondations
rtsp://sdmefr2.media.com/clm/21062007.wmv	TF1, clip, Gorillaz, music
rtsp://sdmefr.media.com/tla/matin0607.wmv	TF1, télé-achat, matin
rtsp://sdmefr3.media.com/jvh/11092007.wmv	TF1, journal, new york
=> New URL	**Inferred keywords**
rtsp://sdmefr3.media.com/jvh/02092007.wmv	TF1, journal

Dailymotion, Google Video, etc). En effet, dans ces cas le *Media Proxy* ne modifie plus les messages de signalisation à la volée car le contrôle de ces flux se fait directement via HTTP, l'URL d'initialisation de la session (contenue dans le bookmark) peut inclure les informations de reprise, positionnant la lecture directement au bon endroit dans le flux.

Le transfert est donc préparé en amont, lors de la création du bookmark, le *Media Proxy* construit alors l'URL de reprise avec la position courante du flux (cf. 5.3.2) et l'URL de base. Le bookmark ensuite communiqué à tous les clients de l'utilisateur permet à ce dernier de reprendre le flux à tout moment simplement en chargeant l'URL du bookmark dans son lecteur multimédia sur le terminal désiré.

Cette solution est préférée à l'approche « proxy » dans la mesure où l'on contrôle le client. En effet, la modification à la volée des messages de signalisation peut avoir des effets néfastes sur le client (affichage erroné de la barre de défilement, incohérence d'état du lecteur, désynchronisation), celui-ci s'attendant à recevoir le flux de données correspondant au début du média tel qu'il l'a demandé. Cette approche n'est cependant envisageable que pour les protocoles basés sur HTTP qui permettent une lecture immédiate et directe de l'URL, seule information transmise au lecteur. Dans le cas des protocoles étudiés au chapitre précédent, l'URL ne fait que déclencher une séquence de messages de signalisation, la requête effective de lecture étant différente de l'URL initiale, rendant la transmission des informations de reprise par cette URL impossible.

5.4 Conclusion

Cette troisième approche dans un cadre très différent, le Web, change beaucoup des précédentes de par la gestion asynchrone de la continuité temporelle. La

synchronisation qui existait jusqu'alors entre la fin de l'instance du service sur le terminal origine et le début de la nouvelle instance sur le terminal destination est ici inexistante. Cependant la continuité existe bel et bien.

La continuité temporelle est assurée dans l'exécution du service et non dans sa fourniture, ici le flux multimédia sera repris par l'utilisateur à l'endroit exact où il l'aura laissé. L'utilisateur est l'orchestrateur du transfert, il n'existe donc plus de contrainte temporelle de cet ordre au niveau infrastructure, il déclenchera le transfert quand il le souhaitera via l'utilisation de ses bookmarks, offrant de nombreuses opportunités. On peut noter que la solution réutilisant le *Media Proxy* pour estimer la position du lecteur dans le flux entraînera les mêmes approximations que dans le chapitre précédent, lié à la mauvaise perception du contrôle utilisateur de la lecture au niveau proxy.

L'approche choisie ici se rapproche du mode Web 2.0 avec une participation et une collaboration des utilisateurs qui enrichissent le modèle de continuité matérialisé par les bookmarks. Le mode déclenchement est alors purement manuel, basé sur l'utilisateur contrairement aux deux approches précédentes, le service multimédia se prêtant particulièrement à cet exercice par l'absence de tout correspondant ou de communication nécessitant une interaction forte. Les opportunités offertes par les bookmarks et le mode Web sont illimitées et le rapprochement avec des services de communications se fait naturellement, le modèle de continuité devenant une motivation d'échange.

La continuité contextuelle est beaucoup plus marquée ici que dans les approches précédentes dans la mesure où nous nous sommes libérés totalement des contraintes de l'IMS. L'utilisateur collabore dans un mode Web 2.0 et c'est lui-même qui enrichit le service via le client GGMedia. Les commentaires, tags, titres, images, etc qui sont générés sont omniprésents d'un terminal à l'autre sans nécessiter de transfert (mêmes informations sur toutes les interfaces), on ne parlera pas de mobilité mais le contexte reste le même, à condition que l'utilisateur utilise un GGMedia (principe du profil de service cf. 2.3.1).

Si le lecteur et le client GGMedia sont relativement indépendants du protocole sous-jacent des services multimédias, il en est tout autre du *Media Proxy* qui doit implémenter des mécanismes de transfert et de traçage spécifiques pour chaque protocole supporté ; encore une fois, cette contrainte limite la mobilité des services

multimédias à quelques implémentations connues a priori. L'utilisateur ne pourra avoir une expérience uniforme pour un même service en fonction des caractéristiques du fournisseur de contenus choisi.

Enfin, quel que soit le service étudié, le mécanisme de transfert ne représente qu'un maillon du processus de mobilité, de nombreuses étapes sont indispensables pour offrir une solution complète de continuité tel que nous l'avons expérimenté via les trois prototypes présentés ici.