MASTER EN SCIENCES INFORMATIQUES

Alignement des concepts dans les domaines du contrôle d'accès et de la gestion des droits numériques

Guillaume, Gautier

Award date:
2011

Awarding institution:
Universite de Namur

Facultés Universitaires Notre-Dame de la paix

Alignement des concepts dans les domaines du contrôle d'accès et de la gestion des droits numériques

---

Année académique 2010-2011

Mémoire présenté en vue de
l'obtention du grade de master
en informatique

Gautier GUILLAUME

Sous la direction du professeur
J.-N. COLIN

Résumé

Depuis l'apparition des supports numériques, les technologies ont bien évolué, de telle sorte qu'il est aujourd'hui possible de copier un support sans perte de qualité. Il est donc devenu impératif de protéger les œuvres créées afin de faire valoir les droits d'auteurs. C'est pourquoi les propriétaires et les distributeurs cherchent des moyens de stopper cette diffusion illégale de données. Les principaux moyens utilisés sont :

- Les modèles de contrôle d'accès qui permettent de vérifier la légitimité de l'accès d'un utilisateur à un objet.
- Les technologies de gestion de droits numériques qui permettent de définir les droits des utilisateurs via les langages d'expression de droits ;

Le présent mémoire a pour objectif de définir un modèle générique permettant à la fois de contrôler les accès et d'exprimer les droits des utilisateurs.

Pour ce faire, une analyse détaillée des principaux modèles de contrôle d'accès et des principaux langages d'expression de droits existants a été réalisée.

Remerciements

En préambule à ce mémoire, je souhaite adresser mes remerciements les plus sincères aux personnes qui m'ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire.

Je tiens à remercier Monsieur Jean-Noël Colin, qui, en tant que Directeur de mémoire, s'est toujours montré à l'écoute et très disponible. Merci pour toute l'aide qu'il a pu m'apporter ainsi que pour le temps qu'il a bien voulu me consacrer.

Je n'oublie pas ma famille et mes amis pour leur soutien moral, leur encouragement et surtout leur patience tout au long de mes études.

Merci à tous et à toutes.

Table des matières

Table des figures

Index des tableaux

1 Introduction

L'objectif principal de ce mémoire consiste à définir un modèle générique dans le cadre du contrôle d'accès et de la gestion des droits numériques. Pour ce faire, le travail est subdivisé en trois parties.

La première partie est axée sur la politique de sécurité et plus particulièrement celle des modèles de contrôle d'accès. Ceux-ci fournissent un langage pour l'application de politiques d'accès destiné à gérer l'autorisation sécuritaire et à connecter des sujets autorisés à des objets par le biais de déclarations de politiques formalisées. Cette partie aborde les modèles de base tels que DAC (Discretionary Access Control), MAC (Mandatory Access Control), RBAC (Role Based Access Control), le modèle de contrôle d'accès contextuel ABAC (Attributed Based Access Control) mais également le modèle Or-BAC (Organization Based Access Control).

Différentes technologies sont utilisées pour mettre en place cette gestion des droits d'accès. Celles-ci, plus communément appelées DRM (Digital Rights Management) ou GDN (Gestion des droits Numériques) en français, sont abordées dans la deuxième partie de ce mémoire. Les DRM se basent sur un langage d'expression de droits (REL – Rights Expression Language) permettant d'exprimer les opérations qu'un utilisateur peut effectuer sur un contenu et sous quelles conditions. Dès lors, plusieurs langages d'expression de droits tels que ODRL, XrML et MPEG21 REL ont été étudiés. De plus, la politique de contrôle de gestion XACML a également été analysée dans cette deuxième partie étant donné que lorsqu'elle est combinée à un système de contrôle d'accès, elle forme un système complet d'expression de droits.

L'objectif des analyses de ces modèles de contrôle d'accès et des langages d'expression de droits est d'en dresser un comparatif afin d'extraire les principaux concepts qui permettront de définir, dans la dernière partie de ce mémoire, un modèle théorique générique appelé « domain model ».

2 Modèles de contrôle d'accès

2.1 Définition

Le contrôle d'accès est un service de sécurité dont les fonctionnalités permettent de vérifier la légitimité de l'accès d'un sujet à un objet, sujet ayant pour but la réalisation d'une action. Un objet peut être défini comme étant un conteneur d'informations. Sont considérés comme tel : un fichier, une imprimante, un programme, un processus, … Un objet est la cible d'une action généralement élémentaire (par exemple « lire », « écrire », …). Le sujet (par exemple un utilisateur, un programme, un processus, …), initiateur de cette action, est dit actif et l'objet auquel il souhaite accéder ou la donnée contenue dans celui-ci sont à l'inverse dits passifs.

On peut se rendre compte que la distinction sujet/objet est un concept fondamental du contrôle d'accès : un sujet effectue toujours une action sur un objet. Cependant, il parait évident avec les exemples donnés ci-dessus, qu'un sujet peut parfois prendre la place d'un objet dans une autre opération ; et inversement, qu'un objet peut également devenir le sujet d'une autre. Par exemple, dans le cas où un utilisateur, par l'intermédiaire d'un programme quelconque, désire modifier une valeur présente dans un fichier. Le programme en question est vu comme étant un objet dans la relation utilisateur-programme ; mais comme étant un sujet dans la relation qui le lie au fichier.

Le contrôle d'accès permet en définitive de limiter les actions qu'un sujet peut réaliser. Il contraint donc ce que l'utilisateur peut faire directement, mais aussi les programmes exécutés sur demande de l'utilisateur, afin que seules les entités autorisées puissent accéder aux ressources du système d'information.

Les principaux modèles de contrôle d'accès existants, à savoir les modèles DAC, MAC, RBAC, ABAC et Or-BAC, sont présentés dans la section suivante.

2.2 Description des principaux modèles d'accès existants

2.2.1 DAC

2.2.1.1 Introduction

Le contrôle d'accès discrétionnaire (DAC) est défini comme :

> Un moyen de limiter l'accès à des objets basé sur l'identité des sujets et/ou des groupes auxquels ils appartiennent. Les contrôles sont discrétionnaires dans le sens ou un sujet possédant un certain droit d'accès est capable de transmettre cette permission (peut-être de manière indirecte) à n'importe quel autre sujet (sauf si cela est restreint par un contrôle d'accès obligatoire) (Latham, 1985 p. 107). [Notre traduction]

Il repose sur un concept de base défini en 1972 par James P. Anderson selon lequel chaque sujet d'un système et chaque objet doivent être identifiés et mis en relation en accord avec leurs droits d'accès (Anderson, 1972 p. 8). Afin de représenter ces relations, Anderson préconise l'utilisation d'une matrice sujet-objet définissant, pour chaque sujet, si l'accès est autorisé à un objet et avec quels privilèges (Anderson, 1972 pp. 9,10). Un exemple d'une telle matrice est repris dans le Tableau 1.

Tableau 1 Matrice sujet-objet

	Objet 1	**Objet 2**	**Objet 3**	**…**	**…**	**Objet n**
Sujet 1	rw	rw	rw	rw	rw	rw
Sujet 2	-	-	r	-	-	rw
Sujet 3	c	r	r	c	-	-
…	-	-	-	-	-	-
Sujet n	r	rw	rw	-	cp	rw

Le type d'accès « r » représenté dans la matrice indique un accès lecture (« read ») ; le type d'accès « w » dénote quant à lui un accès écriture (« write »). Dans cet exemple, chaque ligne s de la matrice peut être vue comme une liste de capacités pour le sujet s (ex : le sujet 2 a la capacité de lire l'objet 3), tandis que chaque colonne o est similaire à une liste d'accès pour l'objet o (ex : l'objet 2 est accessible en lecture par le sujet 3). Diverses combinaisons de ces types d'accès existants sont également réalisables (ex : le sujet 2 a la capacité de lire/écrire l'objet n ; l'objet 2 est accessible en lecture/écriture par le sujet 1 et n). Il existe de plus deux autres types d'accès, mais considérés plus précisément comme des autorisations de contrôle. Le type d'accès « c » signifie l'autorisation de contrôle (« control permission ») et le type d'accès « cp » signifie l'autorisation de contrôle avec délégation (« control with passing ability »).

L'autorisation de contrôle (avec ou sans délégation) diffère des types d'accès classiques (ex : lecture / écriture / …) dans la mesure où elle indique quels sont les sujets ayant la possibilité d'attribuer, de retirer ou de modifier ces types d'accès.

Patrick R. Gallagher nous explique la différence entre les deux types d'autorisation de contrôle (Gallagher, 1987 p. 17) :

- Les sujets disposant d'une autorisation de contrôle peuvent uniquement définir les types d'accès dont les autres sujets peuvent effectuer sur les objets ;
- Les sujets disposant d'une autorisation de contrôle avec délégation ont en plus la capacité d'attribuer des autorisations de contrôle à d'autres sujets.

Quatre modèles de contrôle ont été définis par Patrick R. Gallagher, autorisant à un sujet ou groupe de sujets ce type de modification (Gallagher, 1987 pp. 17-19) :

- *Hiérarchique* : la représentation du système est faite de manière hiérarchique ; l'ensemble des objets étant structuré de manière à ce que chaque nœud soit représenté par un répertoire/dossier dans la hiérarchie de stockage. Le type d'accès et l'autorisation de contrôle sur ce dossier permettraient au sujet ayant ce droit de pouvoir modifier les modes d'accès pour tous les objets en faisant partie. Cette solution permet à l'administrateur du système de déléguer les droits/permissions de contrôle à différents sujets de l'organisation. Un exemple de ce modèle est donné à la Figure 1.

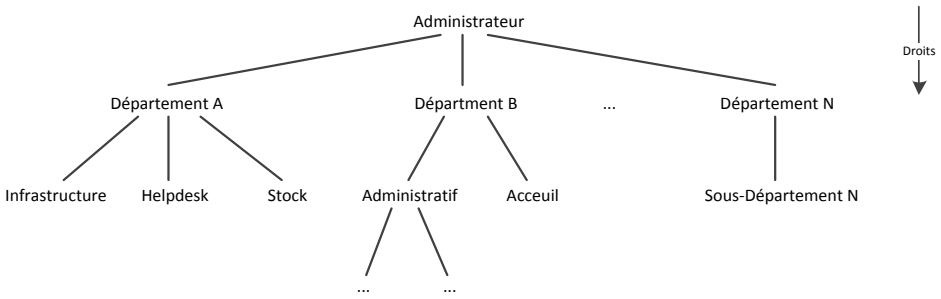

Figure 1 Modèle de contrôle d'accès hiérarchique

- Le modèle basé sur le *concept de propriété* associe pour chaque objet un propriétaire. Ce dernier, en plus d'être généralement le créateur, est le seul sujet autorisé à effectuer des modifications sur les droits d'accès à cet objet. Par défaut, le propriétaire possède toutes les permissions mais n'a pas la capacité de transmettre ce contrôle à d'autres sujets. En définitive, tout propriétaire d'un objet a la possibilité d'ajouter et/ou de retirer des droits d'accès à tout autre sujet sur cet objet en question. On peut donc comparer ce modèle au modèle hiérarchique, mais seulement sur deux niveaux. Un exemple de ce modèle est donné à la Figure 2.

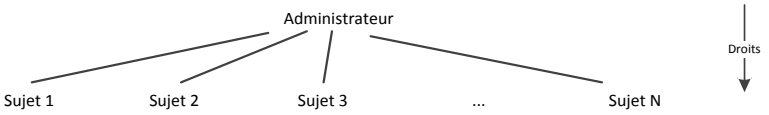

Figure 2 Modèle de contrôle d'accès concept de propriété

- Le modèle suivant, appelé « *laissez faire* » permet à quiconque possédant l'autorisation de contrôle avec délégation de transmettre cette permission à d'autres personnes. Le concept de propriété n'est pas présent dans ce modèle ; ce qui implique qu'il doit être possible de transmettre ce contrôle sans le consentement du propriétaire/créateur de l'objet.
- Le dernier modèle est appelé « *centralisé* » dans la mesure où seule une personne – généralement l'administrateur système – possède l'autorisation de contrôle sur l'ensemble des objets, sans bénéficier de l'option de délégation. Par conséquent, il est le seul à pouvoir effectuer des modifications de droits d'accès sur l'ensemble des objets présents dans le système.

2.2.1.2 Le modèle HRU

En 1976, Harrison, Ruzzo et Ullman présentent ce qui deviendra le modèle discrétionnaire le plus connu, le modèle HRU (Harrison, et al., 1976).

Comme tout modèle DAC, le modèle HRU repose sur une matrice d'accès avec des sujets et des objets en tant qu'indices de lignes et de colonnes. Comme indiqué par Harrison et al. dans leur publication (Harrison, et al., 1976 p. 2), la configuration de ce système de protection est un triple (S, O, P) où :

$S = l'ensemble\ des\ sujets\ actuels$

$O = l'ensemble\ des\ objets\ actuels\ avec\ S \subseteq O$

$P = la\ matrice\ d'accès$

De fait, $P(s, o)$ où $s \in S$ et $o \in O$ renvoie comme information les droits que possède le sujet s sur l'objet o.

Ce système de protection est composé des deux parties suivantes :

- Un ensemble fini R de droits génériques ;
- Un ensemble fini C de commandes ayant la forme suivante :

commande $\alpha(x_1,x_2,\ldots,x_k)$
si r_1 *in* (x_{s_1}, x_{o_1}) *et*
r_2 *in* (x_{s_2}, x_{o_2}) *et*
…
r_m *in* (x_{s_m}, x_{o_m})
alors op_1
op_2
…
op_n
fin.

Ou si $m = 0$, simplement :

commande $\alpha(x_1,x_2,\ldots,x_k)$
op_1
…
op_n
fin.

Avec α comme nom de commande, $r_1, \ldots, r_m$ des droits génériques, $s_1, \ldots, s_m$ et $o_1, \ldots, o_m$ des entiers compris entre 0 et k, $x_n, \ldots, x_k$ des paramètres formels liés soit aux sujets, soit aux objets, le couple (x_{si}, x_{oi}) un élément de la matrice d'accès P et op_i une des opérations primitives suivantes :

ajouter r dans (x_s, x_o)
supprimer r dans (x_s, x_o)
créer le sujet x_s
créer l'objet x_o
supprimer le sujet x_s
supprimer l'objet x_o

où r ($\in R$) représente un droit générique.

2.2.1.3 Conclusion

On peut conclure la description du modèle discrétionnaire en précisant que celui-ci peut être lourd à l'utilisation (dans le sens où il demande beaucoup de ressources) dans le cas où le nombre de sujets et/ou objets serait important car tous ceux-ci sont représentés dans la matrice d'accès. Une solution pour palier à ce grand nombre d'informations consisterait à utiliser la notion de groupe et, en fonction de l'appartenance à un groupe et des droits dont celui-ci dispose, un sujet bénéficierait des mêmes droits. La difficulté de cette solution est de bien définir les groupes afin de ne pas donner plus de droits que nécessaire à certains sujets ; de même lors de la suppression d'appartenance d'un sujet à un groupe, celui-ci perdrait tous les droits dont il avait hérité alors que l'on souhaiterait peut-être simplement retirer les droits de ce sujet sur un objet bien précis.

De plus, les modèles DAC ne sont pas capables de protéger les données (objets) contre les chevaux de Troie contenus dans les programmes. Dès lors que l'exécution du programme avec les droits de l'utilisateur est autorisée, tout le contenu de ce programme sera traité.

Néanmoins, l'utilisation d'un modèle DAC a l'avantage d'être compréhensible aisément ; la simple vue de la matrice d'accès permet de connaitre l'ensemble des droits dont un sujet dispose sur l'ensemble des objets du système et inversement, de connaître l'ensemble des droits qui ont été attribués à un objet pour l'ensemble des sujets du système. Chaque sujet peut, dans le cas d'un modèle basé sur le concept de propriété, définir quelles sont les types d'accès dont peut bénéficier tel ou tel autre sujet. Cette vision du modèle en fait un modèle décentralisé car chaque sujet se voit attribuer la tâche de donner les accès nécessaires aux autres sujets.

2.2.2 MAC

2.2.2.1 Introduction

Le contrôle d'accès mandataire (MAC) est défini comme :

> Un moyen de restreindre l'accès à des objets basé sur la sensibilité (représentée par un label/étiquette) de l'information contenue dans les objets et sur l'autorisation formelle (c'est-à-dire l'habilitation) des sujets d'accéder aux informations d'une telle sensibilité (Latham, 1985 p. 109). [Notre traduction]

Ce type de sécurité est également dénommé « sécurité à plusieurs niveaux » (Multilevel Security) définie comme :

> Une classe de système contenant des informations de sensibilités différentes qui permet un accès simultané à des sujets ayant des habilitations différentes, mais empêche des sujets d'obtenir un accès aux informations pour lesquelles ils n'ont pas l'autorisation (Latham, 1985 p. 109). [Notre traduction]

2.2.2.2 Le modèle de Bell et La Padula

C'est en 1976 que Bell et La Padula proposent leur modèle mandataire qui est un des plus connus et utilisés. Dans leur publication, ils donnent la description de celui-ci ainsi que des éléments entrant en compte (Bell, et al., 1976 p. 9) :

$$Sujet = l'entité\ active, notée\ S_i\ de\ manière\ individuelle\ ou\ S\ pour\ l'ensemble$$
$$Objet = l'entité\ active, notée\ O_i\ de\ manière\ individuelle\ ou\ O\ pour\ l'ensemble$$

Aucune restriction n'est faite en ce qui concerne les entités qui peuvent être à la fois des sujets et des objets.

Le dernier élément du modèle concerne les modes d'accès appelés « attributs d'accès ». Ils ont pour origine une combinaison des deux effets que peut avoir un accès sur un objet, à savoir :

- La récupération d'informations (observation/lecture)
- L'insertion d'information (modification)

Dès lors, les attributs d'accès possibles sont :

e (¬ observation,¬ modification) = execute (exécuter)

r (observation, ¬ modification) = read (lire)

a (¬observation, modification) = append (ajouter)

w (observation et modification) = write (lire et ajouter = écrire)

En plus de ces éléments, le modèle de Bell La Padula utilise la notion de niveau de sécurité, représenté par le couple (classification/habilitation, ensemble de catégories) où :

- L'habilitation est un élément d'un ensemble totalement ordonné. Ex : Top Secret (***TS***), Secret (***S***), Confidentiel (***C***) et Libre (***L***) où $TS > S > C > L$;
- L'ensemble de catégories est un ensemble d'éléments variant en fonction du domaine d'application. Ex : Armée, Marine, Aviation, Nucléaire.

Afin de déterminer la dominance/supériorité d'un niveau de sécurité sur un autre, Bell et La Padula proposent cette relation (Bell, et al., 1976 pp. 13-14) :

Soit f_i *le niveau de sécurité du sujet* S_i
f_k *le niveau de sécurité du sujet* S_k

représentée respectivement par les couples (L_i, SC_i) et (L_k, SC_k) où L_i, L_k correspondent au niveau d'habilitation et SC_i, SC_k à l'ensemble de catégories des sujets respectifs.

Alors f_i domine/est supérieur à f_k noté $f_i \geq f_k$ si et seulement si :

- $L_i \geq L_k$: le niveau d'habilitation c_i est ≥ au niveau d'habilitation de c_k
- $SC_i \supseteq SC_k$: l'ensemble de catégories de c_i inclus l'ensemble de catégories c_k

On dira également que :

- f_i est strictement supérieur à f_k ssi $(L_i > L_k) \wedge (SC_i > SC_k)$;
- f_i est incomparable à f_k (noté $f_i <> f_k$) si $\neg(f_i \geq f_k) \wedge \neg(f_k \geq f_i)$.

Une particularité de ce niveau de sécurité a été définie : l'utilisation d'un niveau de sécurité courant et maximum. Grâce à cette fonctionnalité, il est possible pour un sujet de « travailler » avec un niveau moins élevé si les conditions ne l'obligent pas ; cela apporte une sécurité supplémentaire aux données plus sensibles.

Pour ce faire Bell et La Padula ont donc défini (Bell, et al., 1976 pp. 13-14) :

$f_s(S_i)$ = *le niveau maximum de sécurité pour le sujet* S_i

$f_c(S_i)$ = *le niveau courant de sécurité pour le sujet* S_i

$f_o(O_j)$ = *le niveau maximum de sécurité pour le sujet* O_j

où il paraît évident que $f_s(S_i) \geq f_c(S_i)$.

Afin de pouvoir utiliser ces éléments (sujet, objet, accès) du modèle et la notion de niveau de sécurité, Bell et La Padula y ont intégré deux grandes propriétés de sécurité.

La première est appelée la propriété de sécurité simple (« simple security property » ou « ss-property ») et est définie comme suit :

> Un modèle de règle de sécurité autorisant pour un sujet, un accès en lecture sur un objet seulement si le niveau de sécurité du sujet domine/est supérieur au niveau de sécurité de l'objet (Latham, 1985 p. 111). [Notre traduction]

Cette propriété apporte de la confidentialité au modèle car il est impossible de lire un objet de sécurité supérieure ; elle est d'ailleurs également appelée NRU signifiant « no-read-up ». Formellement, elle se traduit par (Bell, et al., 1976 p. 16) :

Soit $f_c(S_i)$ = *le niveau courant de sécurité du sujet* S_i

$f_o(O_j)$ = *le niveau de sécurité de l'objet* O_j

Alors la « ss-property » est satisfaite si l'attribut d'accès est :

- « a »
- « r » ou « w » et $f_c(S_i) \geq f_o(o_j)$

Cette propriété est illustrée à la Figure 3.

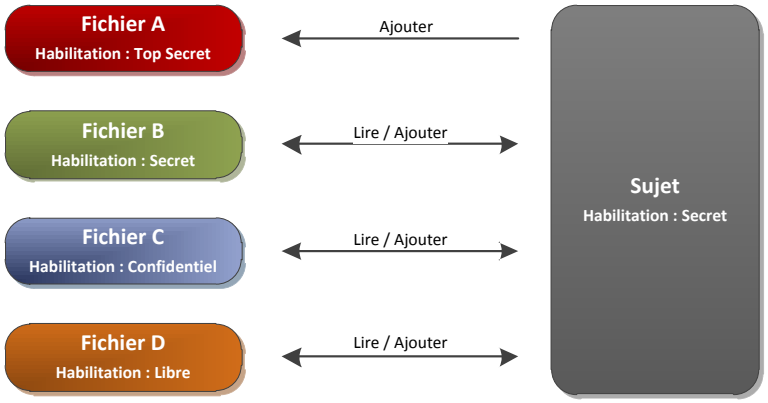

Figure 3 Exemple de la ss-property du modèle de Bell La Padula (Red Hat, 2011)

La deuxième propriété est appelée la propriété * (« *-property ») définie comme suit :

> Un modèle de règle de sécurité autorisant pour un sujet un accès en écriture sur un objet seulement si le niveau de sécurité du sujet est dominé/inférieur au niveau de sécurité de l'objet (Latham, 1985 p. 111). [Notre traduction]

Cette propriété apporte du confinement au modèle car il est impossible d'écrire dans un objet ayant un niveau de sécurité inférieur ; elle est d'ailleurs également appelée NWD signifiant « no-write-down ». Formellement, cela se traduit par (Bell, et al., 1976 p. 18) :

Soit $f_c(S_i) = le\ niveau\ courant\ de\ sécurité\ du\ sujet\ S_i$

$f_o(O_j) = le\ niveau\ de\ sécurité\ de\ l'objet\ O_j$

Alors la « *-property » est satisfaite si l'attribut d'accès est :

- « r »
- « a » ou « w » et $f_c(S_i) \leq f_o(o_j)$

Un exemple de cette propriété est donné à la Figure 4.

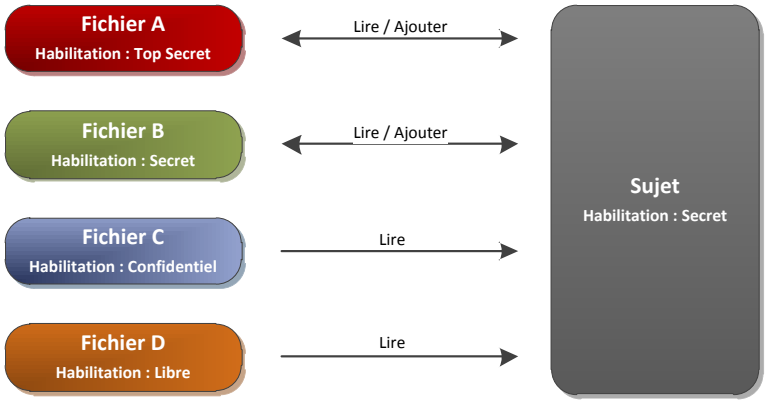

Figure 4 Exemple de la *-property du modèle de Bell La Padula (Red Hat, 2011)

La combinaison des deux propriétés de sécurité donne le modèle illustré à la Figure 5 dans lequel un sujet possède un niveau d'habilitation « secret ». Il lui est dès lors possible, en vertu des règles définies par la combinaison des deux propriétés de lire tout fichier possédant un niveau inférieur (confidentiel, libre), d'écrire dans tout type de fichier de même niveau (secret) et d'ajouter uniquement dans ceux d'un niveau supérieur (top secret).

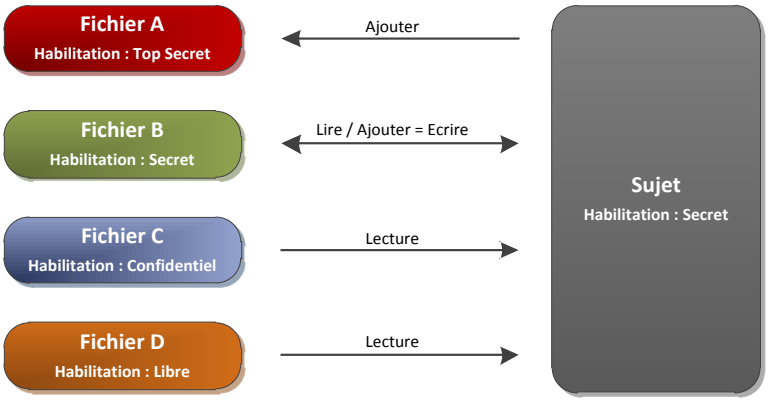

Figure 5 Exemple du modèle Bell La Padula (Red Hat, 2011)

Ces deux propriétés apportent à la fois de la confidentialité et du confinement mais ne sont pas suffisantes pour l'élaboration d'un modèle complet sécurisé. La propriété de sécurité simple (« ss-property ») garanti la confidentialité des données car elle empêche un sujet S_i de lire des dossiers contenus dans un objet O_j possédant un niveau de sécurité plus élevé ($f_o(O_j) > f_c(S_i) \rightarrow \neg r$). Mais la deuxième propriété (« *-property ») ne permet pas de garantir l'intégrité car celle-ci est satisfaite dans le cas où un sujet S_i souhaite ajouter (attribut d'accès « a ») de l'information dans un objet O_j ayant un niveau de sécurité plus élevé.

2.2.2.3 Le modèle de Biba

Un autre modèle mandataire, dual à celui de Bell La Paluda, est le modèle de Biba publié en 1975 par K.J. Biba (Biba, 1975). Ce modèle possède la particularité de protéger l'intégrité des données, mais ne permet pas d'assurer la confidentialité. Pour ce faire, Biba utilise la notion de niveaux d'intégrité représenté par le couple (intégrité, ensemble de catégories) où (Biba, 1975 pp. 20-22) :

- L'intégrité est un élément d'un ensemble totalement ordonné. Ex : Crucial (*C*), très important (*TI*), Important (*I*) où $C > TI > I$.
- L'ensemble des catégories est un ensemble d'éléments variant en fonction du domaine d'application. Ex :{Logistique, Simulation, Contrôle de budget}.

De manière analogue à Bell La Padula, il existe une relation (notée *g*) permettant de déterminer la dominance/supériorité d'un niveau d'intégrité sur un autre et deux grands principes sont mis en place par Biba (Biba, 1975 p. 28).

Le premier est appelé l'intégrité simple (« simple secure integrity » ou « ss-integrity »). Il est défini comme suit : un modèle de règle de sécurité autorisant pour un sujet, un accès en lecture sur un objet seulement si le niveau d'intégrité du sujet est dominé/inférieur par le/au niveau d'intégrité de l'objet (Latham, 1985 p. 111). Cette propriété apporte du confinement au modèle car il est impossible de lire un objet ayant un niveau d'intégrité inférieur ; on l'appelle par ailleurs également RU signifiant « read-up ». Formellement, cela se traduit par (Biba, 1975 p. 29) :

Soit $g_s(S_i)$=*le niveau d'intégrité du sujet* S_i

$g_o(O_j)$= *le niveau d'intégrité de l'objet* O_j

Alors la « ss-integrity » est satisfaite si l'attribut d'accès est :

- « a »
- « r » ou « w » et $g_s(S_i) \leq g_o(O_j)$

Cette propriété est illustrée à la Figure 6.

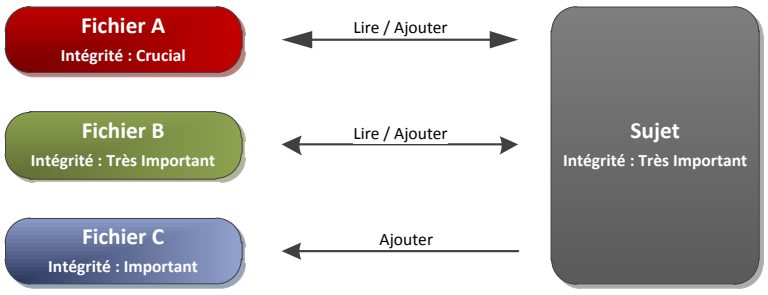

Figure 6 Exemple de la ss-integrity du modèle de Biba (Red Hat, 2011)

Le deuxième est appelé l'intégrité * (« *-integrity ») et est définie comme suit : Un modèle de règle de sécurité autorisant un sujet, un accès en écriture sur un objet seulement si le niveau d'intégrité du sujet domine le niveau d'intégrité de l'objet.

Cette propriété apporte de l'intégrité au modèle car il est impossible d'écrire dans un objet ayant un niveau d'intégrité supérieur ; on l'appelle par ailleurs également WD signifiant « write-down ». Formellement, elle se traduit par (Biba, 1975 p. 29) :

Soit $g_s(S_i) = le\ niveau\ d'intégrité\ du\ sujet\ S_i$

$g_o(O_j) = le\ niveau\ d'intégrité\ de\ l'objet\ O_j$

Alors la « *-integrity » est satisfaite si l'attribut d'accès est :

- « r »
- « a » ou « w » et $g_s(S_i) \geq g_o(O_j)$

Cette propriété est illustrée à la Figure 7.

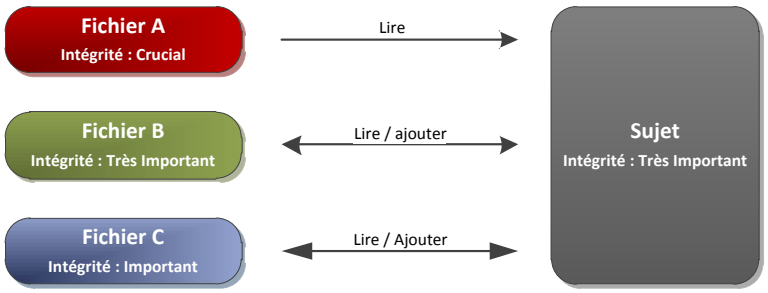

Figure 7 Exemple de la *-integrity du modèle de Biba (Red Hat, 2011)

La combinaison des deux propriétés d'intégrité dans le modèle de Biba donne le modèle illustré à la Figure 8 dans lequel un sujet possède un niveau d'intégrité « Très important ». Il lui est dès lors possible, en vertu des règles définies par la combinaison des deux propriétés d'ajouter dans tout type de fichier de niveau inférieur (important), d'écrire dans tout type de fichier de même niveau et de lire uniquement ceux de niveau supérieur (crucial).

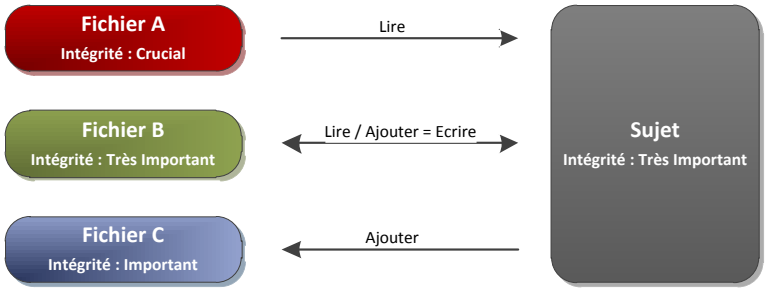

Figure 8 Exemple du modèle de Biba (Red Hat, 2011)

Ces deux principes apportent à la fois de l'intégrité et du confinement mais tout comme dans le modèle de Bell La Padula, ils ne permettent pas d'assurer une sécurité optimale. La deuxième propriété (*-integrity) ajoute l'intégrité des données car elle empêche un sujet S_i d'altérer des données contenues dans un objet O_j possédant un niveau d'intégrité plus élevé $(g_o(O_j) > g_s(S_i) \rightarrow \neg w \wedge \neg a)$. Mais la propriété d'intégrité simple ne permet pas de garantir la confidentialité.

2.2.2.4 Combinaison des modèles Bell La Padula et Biba

Une combinaison de ces deux modèles permettrait de protéger à la fois la confidentialité des données à l'aide de Bell La Padula mais également de leur intégrité grâce à Biba. Cependant, cette combinaison ne peut être envisageable qu'après modification des modèles. En effet, il est nécessaire que tous les sujets S_i et objet O_j bénéficient d'un niveau de sécurité mais aussi d'un niveau d'intégrité. Dès lors, il est possible d'établir les deux propriétés à intégrer à ce nouveau modèle.

La première est définie comme : Un modèle de règle de sécurité autorisant, pour un sujet, un accès en lecture sur un objet seulement si le niveau de sécurité du sujet domine celui de l'objet et si le niveau d'intégrité du sujet est dominé par celui de l'objet. Formellement, cela se traduit comme suit :

Soit $f_c(S_i) = le\ niveau\ courant\ de\ sécurité\ du\ sujet\ S_i$
$f_o(O_j) = le\ niveau\ de\ sécurité\ de\ l'objet\ O_j$
$g_s(S_i) = le\ niveau\ d'intégrité\ du\ sujet\ S_i$
$g_o(O_j) = le\ niveau\ d'intégrité\ de\ l'objet\ O_j$

Alors la « ss-property-integrity » est satisfaite si l'attribut d'accès est :

- « a »
- « r » ou « w » et $f_c(S_i) \geq f_o(O_j) \wedge g_s(S_i) \leq g_o(O_j)$

Cette propriété est illustrée à la Figure 9.

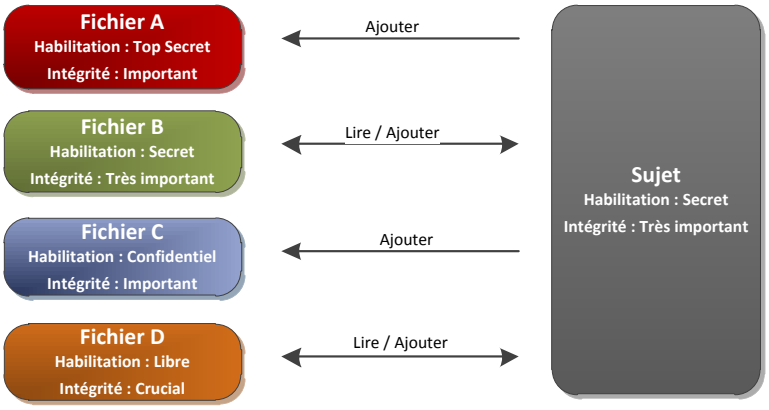

Figure 9 Exemple de la ss-property-integrity (Red Hat, 2011)

La seconde est définie comme : un modèle de règle de sécurité autorisant pour un sujet, un accès en écriture sur un objet seulement si le niveau de sécurité du sujet et dominé par celui de l'objet et si le niveau d'intégrité du sujet domine celui de l'objet. Formellement, cela se traduit par :

Soit $f_c(S_i) = le\ niveau\ courant\ de\ sécurité\ du\ sujet\ S_i$

$f_o(O_j) = le\ niveau\ de\ sécurité\ de\ l'objet\ O_j$

$g_s(S_i) = le\ niveau\ d'intégrité\ du\ sujet\ S_i$

$g_o(O_j) = le\ niveau\ d'intégrité\ de\ l'objet\ O_j$

Alors la « *-property-integrity » est satisfaite si l'attribut d'accès est :

- « r »
- « a » ou « w » et $f_c(S_i) \leq f_o(O_j) \wedge g_s(S_i) \geq g_o(O_j)$

Un exemple de cette propriété est illustré à la Figure 10.

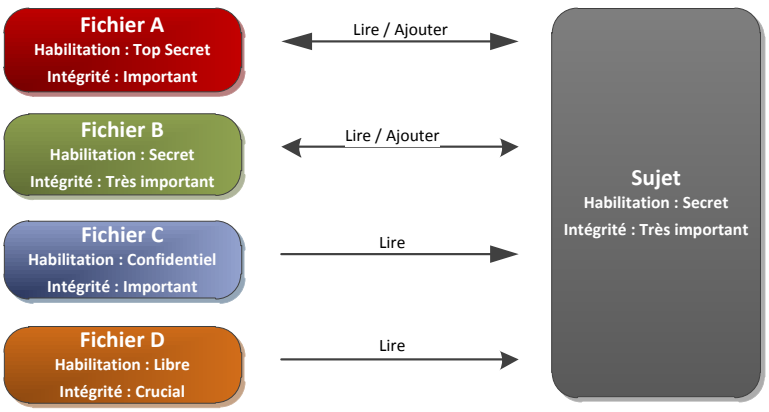

Figure 10 Exemple de la *-property-integrity (Red Hat, 2011)

La combinaison des deux propriétés donne le modèle illustré à la Figure 11 dans lequel un sujet possède un niveau d'habilitation « Secret » et un niveau d'intégrité « Très important ». Il lui est dès lors possible d'ajouter dans le Fichier A, d'écrire dans le Fichier B et de lire le Fichier D. Aucune action n'est autorisé sur le Fichier C.

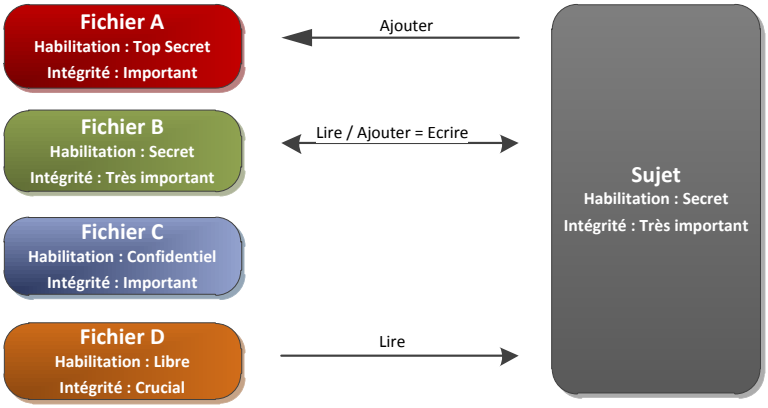

Figure 11 Exemple de la combinaison des modèles de Bell La Padulla et de Biba (Red Hat, 2011)

2.2.2.5 Conclusion

Aucun des modèles mandataires présentés ne dispose d'une solution de sécurité complète. L'absence de gestion de l'intégrité dans le modèle de Bell La Padula est compensée dans celui de Biba mais en contrepartie, on y perd la protection sur la confidentialité. La combinaison de ceux-ci permet toutefois de résoudre les manquements à la sécurité mais la complexité de mise en place et l'administration en fait perdre toute la souplesse de départ. De plus, ces modèles nécessitent une gestion centralisée car les niveaux de sécurité et d'intégrité ne peuvent être définis par l'ensemble des sujets ; ils sont complètement statiques. Leur utilisation est plus appropriée dans des domaines d'application militaires possédant plusieurs niveaux de sécurité.

2.2.3 RBAC

2.2.3.1 Introduction

C'est en 1992 que fût proposé par D.F. Ferraiolo et D.R. Kuhn le modèle basé sur les rôles RBAC (Role-Based Access Controls) (Ferraiolo, et al., 1992). Celui-ci a été reconnu comme standard ANSI en 2004.

Le modèle RBAC a été développé dans le but de répondre aux limitations présentes dans les modèles discrétionnaires et mandataires cités précédemment ; que cela soit la complexité d'administration des autorisations, ou bien la limitation liée au domaine d'application (militaire/hospitalier,…).

Dans le modèle DAC présenté dans la section 2.2.1, des droits sont accordés à certains utilisateurs leur permettant d'octroyer des permissions à d'autres utilisateurs. Ce modèle n'est pas très flexible et n'est pas représentatif des entreprises dans lesquelles les accès aux documents/objets sont plus basés sur les fonctions des utilisateurs (ex : les dossiers d'un patient peuvent être accessibles à toutes les personnes travaillant en tant que médecin mais pas pour celles travaillant en logistique). Avec cette notion de fonction au sein d'une entreprise, on peut faire intervenir la notion de rôle, chaque rôle correspondant à une fonction, c'est-à-dire un ensemble d'utilisations de documents/objets. La différence avec le modèle discrétionnaire est que les accès ne sont plus attribués par des utilisateurs mais en fonction de leurs rôles au sein de l'entreprise.

Le modèle RBAC se rapproche plus du modèle mandataire (voir §2.2.2) dans lequel la structure se base sur une classification et un ensemble de catégories. On pourrait utiliser l'ensemble de catégories afin de représenter les documents/objets accessibles uniquement à certains « groupes » de personnes. Le modèle RBAC peut donc être vu comme un modèle mandataire mais à un seul niveau de sécurité. La différence entre ces modèles repose principalement sur la manière dont les modes d'accès sont représentés. Dans le modèle MAC, ils sont basés sur les types d'accès classiques, c'est-à-dire lecture, écriture,… Le modèle RBAC, quant à lui, se base sur l'utilisation d'un objet, c'est-à-dire la combinaison d'un type d'accès et de son objet associé.

Dans leur modèle, Ferraiolo et Kuhn définissent un rôle comme suit : « Un ensemble de transactions qu'un utilisateur peut effectuer au sein d'une organisation. » (Ferraiolo, et al., 1992 p. 4) [Notre traduction]

De même, une transaction est définie comme suit : « Une procédure de transformation et un ensemble de données associées. » (Ferraiolo, et al., 1992 p. 4) [Notre traduction]

Il s'agit en fait de la combinaison d'une opération et de son objet. Par exemple, l'opération « lecture » ne peut être considérée comme une transaction car elle n'est en rien associée à un objet. Par contre, la « lecture d'un dossier patient » peut être considérée comme tel.

L'importance des contrôles au travers de transactions et pas par des simples accès de type lecture, écriture peut être utile dans le cadre d'une analyse « fine » car si par exemple un utilisateur S_1 nécessite un accès « r » sur un objet O_1 dans le cadre d'une transaction T_1, il est possible de différencier cette transaction T_1 d'une transaction T_2 utilisée par un utilisateur S_2 ayant le même droit d'accès sur le même objet O_1. Sans la notion de transaction, nous ne verrions simplement pas que S_1 et S_2 possèdent le même droit d'accès sur l'objet O_1 ; avec celle-ci il est possible de distinguer l'origine de l'accès (ex : sauvegarde automatique d'une base de données ; lecture d'une information contenue dans la base de données).

L'ensemble de ces relations (appartenance à un rôle, transformation sur un objet) peut être représentée de la manière suivante :

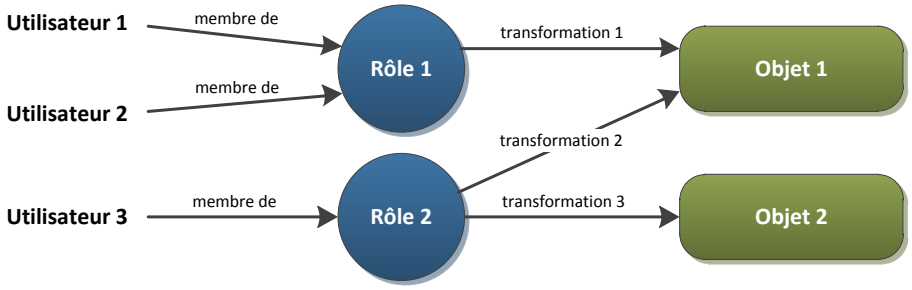

Figure 12 RBAC et les relations entre les utilisateurs/rôles /objets (Ferraiolo, et al., 1992 p. 5)

2.2.3.2 Description formelle

D.F. Ferraiolo et D.R. Kuhn ont décrit le modèle RBAC de manière formelle, en termes d'ensembles et relations de la manière suivante (Ferraiolo, et al., 1992 pp. 5-6) :

Soit $s = un\ sujet$

$r = un\ rôle$

$t = une\ transaction$

Alors $AR(s) = le\ rôle\ actif\ du\ sujet\ s$

$RA(s) = l'ensemble\ des\ rôles\ autorisés\ pour\ le\ sujet\ s$

$TA(r) = l'ensemble\ des\ transactions\ autorisées\ pour\ le\ rôle\ r$

$exec(s,t) = vrai\ si\ le\ sujet\ s\ peut\ exécuter\ la\ transaction\ t$ à $cet\ instant\ donné,$

$faux\ dans\ le\ cas\ contraire$

3 règles de bases sont requises :

1. « Assignation de rôle » : un sujet peut exécuter une transaction seulement si le sujet est assigné à un rôle.

$$\forall s: sujet, t: transaction, (exec(s,t) \Longrightarrow AR(s) \neq \varnothing)$$

2. « Autorisation de rôle » : un rôle actif du sujet doit être autorisé pour le sujet.

$$\forall s: sujet, (AR(s) \subseteq RA(s))$$

   Avec la règle 1, cette règle assure que les sujets ne peuvent assumer que les rôles pour lesquels ils sont autorisés.

3. « Autorisation de transaction » : un sujet peut exécuter une transaction seulement si la transaction est autorisée pour le rôle actif du sujet.

$$\forall s: sujet, t: transaction, (exec(s,t) \Longrightarrow t \in TA(AR(s)))$$

Comme le signalent D.F. Ferraiolo et D.R. Kuhn dans leur article, une transaction est une procédure de transformation pour laquelle un ensemble d'objets ont été associés (Ferraiolo, et al., 1992 p. 6). Le contrôle d'accès dans les règles de base ne requiert aucune vérification des droits d'accès de l'utilisateur sur l'objet, ni de ceux de la procédure de transformation car ceux-ci sont directement intégrés dans la transaction.

Une quatrième règle proposée par les auteurs D.F. Ferraiolo et D.R. Kuhn permet de renforcer un peu plus les contrôles sur les modes dans lesquels les utilisateurs peuvent accéder aux objets (Ferraiolo, et al., 1992 p. 6). Elle consiste à redéfinir le concept de transaction pour se référencer uniquement sur la transaction seule et non plus sur la combinaison d'une transformation et des objets associés.

Formellement, cette 4ème règle se traduit par la relation (Ferraiolo, et al., 1992 p. 6) :

$$\forall s: sujet, t: transaction, o: objet, (exec(s,t) \Longrightarrow access(AR(s),t,o,x)$$

Cette règle indique s'il est permis pour un sujet s dans le rôle $AR(s)$ d'accéder à un objet o dans le mode x via la transaction t (redéfinie en procédure de transformation) où x est pris parmi un ensemble de modes d'accès tels que lire, écrire,…

Cette 4ème règle apporte un contrôle plus fin sur les accès et permet de renforcer les exigences sur la confidentialité, intégrité car il est possible de spécifier les modes d'accès directement. L'autre solution, sans cette 4ème règle, serait de définir deux transactions différentes, où la première autoriserait par exemple un accès en lecture sur un objet et la deuxième un accès en écriture.

2.2.3.3 Hiérarchie

Il est également possible de lier les rôles entre eux de manière à former une hiérarchie où un rôle pourrait être composé d'un certain nombre d'autres rôles (Ferraiolo, et al., 1992 p. 8). Les transactions autorisées pour un rôle le seraient également pour les sous-rôles de celui-ci. Cette notion de hiérarchie est illustrée à la Figure 13.

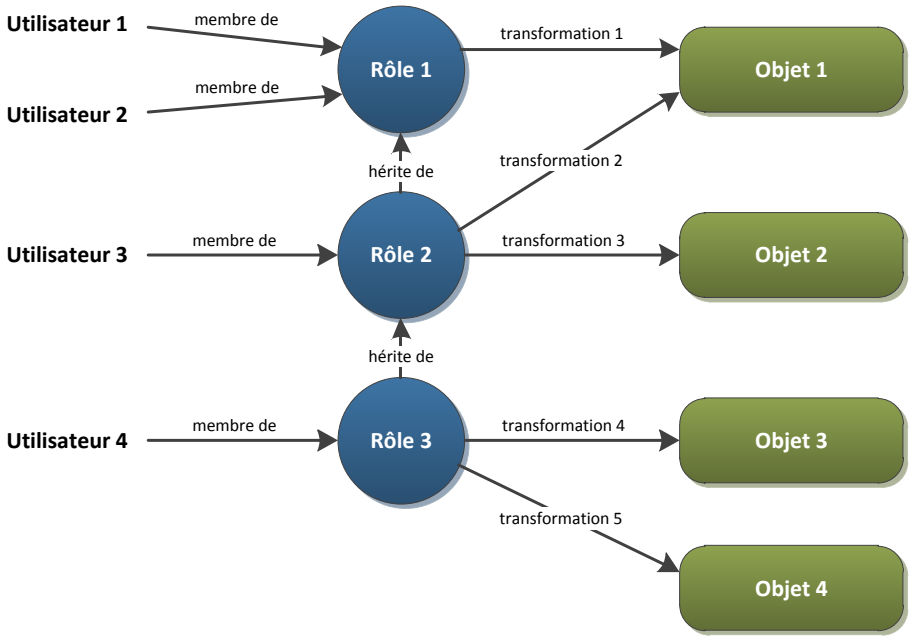

Figure 13 RBAC et la hiérarchie des rôles (Ferraiolo, et al., 1992 p. 8)

Dans l'exemple ci-dessus, le *Rôle 1* est autorisé pour les *Utilisateurs 1 et 2*, ce qui leur donne accès à la *transformation 1*. L'*Utilisateur 3* dont le rôle autorisé est *Rôle 2*, peut effectuer les *transformations 2* et *3* mais également celle du *Rôle 1* car celui-ci est inclus dans le *Rôle 2* (on peut dire que le *Rôle 2* hérite du *Rôle 1*). De la même manière, le *Rôle 3* ⊃ *Rôle 2* ⊃ *Rôle 1* (⊃ = relation d'inclusion), ce qui permet à l'*Utilisateur 4* d'avoir accès à l'ensemble de toutes les transformations existantes dans cet exemple.

2.2.3.4 Session

Afin de limiter le nombre de privilèges/transactions à attribuer pour un utilisateur, le principe du « moindre privilège » peut aussi être utilisé (Ferraiolo, et al., 1992 p. 9). Cela signifie que pour un utilisateur devant réaliser un travail nécessitant un certain nombre de privilèges, aucun autre privilège supplémentaire ne lui sera attribué. Pour ce faire, il faut déterminer l'ensemble minimum des privilèges requis pour la réalisation de ce travail et de restreindre l'utilisateur à un rôle pour lequel seul ces privilèges lui sont autorisés.

Une méthode permettant la mise en place de ce principe serait d'utiliser les « sessions d'utilisateurs ». Une session peut être vue comme une méthode d'authentification d'un utilisateur donnant droit à une liste de rôles autorisés pour celui-ci. Quand l'utilisateur s'est authentifié, il peut activer certains de ses rôles et bénéficier dès lors des autorisations

associées et ce, jusqu'à la fermeture de sa session. Pour des raisons de sécurité, les sessions peuvent être liées à une notion temporelle qui, après un certain laps de temps d'inactivité, déconnecte l'utilisateur de sa session. Cela permet, en cas d'oubli de fermeture de session d'un utilisateur, d'éviter qu'un utilisateur tiers puisse usurper l'identité de celui-ci et bénéficier de ses autorisations.

Les deux nouvelles relations sont :

$$U - R = un\ utilisateur\ est\ associé\ un\ à\ un\ ensemble\ de\ rôles$$
$$U - S - R = un\ utilisateur\ est\ associé\ à\ un\ rôle\ à\ travers\ une\ session$$

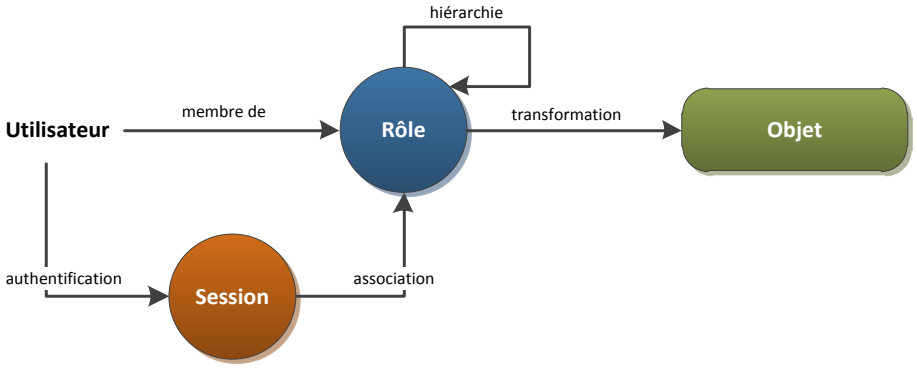

Figure 14 RBAC et la notion de session

2.2.3.5 Séparation des devoirs

Comme le signalent D. F. Ferraiolo et D. R. Kuhn, les mécanismes de leur modèle peuvent être utilisés pour renforcer la politique de réparation tâches/devoirs (Ferraiolo, et al., 1992 p. 9). Il s'agit d'interdire pour un même utilisateur la possibilité d'effectuer deux tâches normalement séparées. L'exemple le plus couramment utilisé et qui décrit parfaitement cette nécessité de séparation concerne les transactions financières où il y a un utilisateur à l'origine du payement et un autre pour l'acceptation de ce payement. L'utilisateur ne peut être la même personne pour des raisons de sécurité car sinon la fraude au payement serait simple à mettre en place.

La séparation des tâches peut être de deux types : soit statistique, soit dynamique. La séparation statique des tâches implique qu'un utilisateur ne peut être autorisé à « jouer » des rôles incluant des tâches normalement séparées. Par exemple, les rôles de caissiers et comptables ne peuvent être autorisés pour un utilisateur.

La séparation dynamique des tâches, quant à elle, ne limite pas les rôles autorisés pour un utilisateur mais impose simplement qu'un utilisateur ne peut pas exercer simultanément les rôles dont les tâches sont normalement séparées. Cela signifie qu'un

utilisateur peut jouer le rôle de caissier et comptable mais pas en même temps. Cela ajoute plus de flexibilité au système, avec la limitation que l'utilisateur à l'origine de la réception du payement (jouant le rôle de caissier) ne peut être la personne comptabilisant l'argent reçu (jouant le rôle de comptable). Il peut comptabiliser l'argent à condition de ne pas être à l'origine de cette réception.

2.2.3.6 Conclusion

Pour conclure, le modèle RBAC proposé par D. F. Ferraiolo et D. R. Kuhn apporte des améliorations par rapport aux modèles discrétionnaires et mandataires ; il est plus à même de représenter les besoins en termes de sécurité au sein d'une organisation. En y intégrant la notion de hiérarchie, il est également possible de simplifier la gestion des rôles et transactions en utilisant ceux préexistants. Cette gestion, réalisée au niveau de l'organisation, inclut l'ajout ou la suppression d'un nouvel utilisateur dans le système en leur accordant un ensemble de rôles en accord avec les tâches à accomplir. Cette aisance de gestion a cependant un désavantage dès lors qu'un rôle doit être modifié. Que cela soit un ajout ou une suppression de transaction pour ce rôle, tous les utilisateurs ayant l'autorisation de jouer celui-ci seront impactés. Il est difficile voire impossible de modifier individuellement les accès au contraire des modèles discrétionnaires. De plus, il est très difficile avec ce modèle de définir des règles de type « les utilisateurs habilités en tant que médecin ont la possibilité de lire les informations contenues dans le dossier de leur patients » ; car si le rôle de médecin apporte la possibilité de lire le contenu du dossier d'un patient, un sujet ayant ce rôle activé pourrait dès lors lire le contenu de n'importe quel dossier patient. Il n'y a pas la possibilité de spécifier quels sont les patients sous la responsabilité du médecin. Finalement, la notion de hiérarchie n'est pas réellement représentative de l'environnement d'une entreprise. Par exemple, dans le cadre d'un environnement hospitalier, un directeur possède une fonction hiérarchiquement supérieure à celle d'un médecin ; ce qui implique dès lors que le directeur serait autorisé à exécuter les mêmes transformations que ce dernier. Mais il parait évident qu'une opération de chirurgie autorisée pour un médecin ne peut être autorisée pour son directeur.

2.2.4 ABAC

2.2.4.1 Introduction

Le modèle basé sur les attributs ABAC (Attribute Based Access Control) a été proposé comme une nouvelle approche des modèles de contrôle d'accès. Les modèles énoncés dans les chapitres précédents ne permettent pas de répondre aux besoins actuels dans le sens où les politiques de contrôle d'accès n'y sont pas décrites avec un niveau de détail suffisant. Seul RBAC propose l'utilisation d'une caractéristique (le rôle) pour les utilisateurs mais il ne prend pas en compte d'autres caractéristiques que celui-ci pourrait avoir (telles que sa fonction, sa date d'engagement,…), ni même celles des objets (comme la date de création, l'auteur,…). Le modèle ABAC essaye de combler cette lacune en

apportant la notion d'attribut aux différents éléments faisant partie d'une politique de contrôle d'accès, à savoir le sujet, la ressource et l'environnement.

Comme l'explique E. Yuan et J. Tang, les attributs sont de trois types (Yuan, et al., 2005 p. 4) :

- *Attributs du sujet* : dans les modèles de contrôle d'accès, le sujet correspond à l'entité effectuant une action sur une ressource. Ce sujet peut être caractérisé par un ensemble d'attributs permettant de le décrire plus en détail. De tels attributs peuvent être par exemple la fonction actuelle (comparable au rôle du modèle RBAC), l'e-mail, la date d'engagement,…
- *Attributs de la ressource* : la ressource est donc l'entité qui subit l'action sollicitée par le sujet. Tout comme pour le sujet, la ressource peut être décrite par un ensemble d'attributs tels que la date de création, l'auteur, le nombre de pages, …
- *Attributs de l'environnement* : Ceux-ci décrivent l'environnement dans lequel l'accès à l'information peut se dérouler. Y sont inclus des attributs tels que la date actuelle, l'heure, le niveau de sécurité,… Il s'agit d'attributs qui ne sont associés ni aux sujets, ni aux ressources mais qui peuvent également jouer un rôle quant à la prise de décision.

2.2.4.2 Description formelle

E. Yuan et J. Tang décrivent de manière formelle le modèle de la manière suivante (Yuan, et al., 2005 p. 4) :

Soit S = l'ensemble des sujets

R = l'ensemble des ressources

E = l'environnement

$SA_k = l'ensemble\ des\ attributs\ prédéfinis\ pour\ les\ sujets$ où $(1 \leq k \leq K)$

$RA_m = l'ensemble\ des\ attributs\ prédéfinis\ pour\ les\ ressources$ où $(1 \leq m \leq M)$

$EA_n = l'ensemble\ des\ attributs\ prédéfinis\ pour\ les\ environnements$ où $(1 \leq n \leq N)$

Alors $Attr(s) = la\ relation\ d'assignation\ d'attributs\ pour\ le\ sujet\ s$

$Attr(r) = la\ relation\ d'assignation\ d'attributs\ pour\ la\ ressource\ r$

$Attr(e) = la\ relation\ d'asignation\ d'attributs\ pour\ l'environnement\ e$

Où $Attr(s) \subseteq SA_1 \times SA_2 \times \ldots \times SA_k$

$Attr(r) \subseteq RA_1 \times RA_2 \times \ldots \times RA_m$

$Attr(e) \subseteq EA_1 \times EA_2 \times \ldots \times EA_n$

$(\times = produit\ cartésien)$

A l'aide de ces relations, il est désormais possible de définir une règle d'accès pour un sujet *s* sur une ressource *r* dans un environnement *e*. Cette règle est la fonction booléenne *f* suivante :

$$can\_access(s,o,e) \leftarrow f(Attr(s), Attr(r), Attr(e))$$

Si la fonction d'évaluation est « vraie », alors l'accès est autorisé. Dans le cas contraire, il se voit refusé. Toutes ces fonctions d'évaluation sont stockées dans une base de règles appelée « Policy Store » utilisée pour établir ou pas l'accès à une ressource.

2.2.4.3 Conclusion

En fin compte, on constate que le modèle ABAC permet d'élaborer des règles de politique plus complexes et s'adaptant mieux aux besoins des entreprises cherchant un contrôle plus poussé sur l'accès aux ressources. De plus, l'ensemble des règles stockées dans la base de règles pourraient être distribué à d'autres entreprises/sous-organisations facilitant dès lors leur déploiement. Cependant, la définition ainsi que la maintenance de ces règles est à charge de l'administrateur du système, ce qui, en fonction du nombre de règles, peut être relativement complexe car il est nécessaire de déterminer correctement quels sont les attributs à prendre en compte et comment les « lier » entre eux.

Finalement, le modèle ABAC n'offre pas de réelle structuration dans la mesure où seuls les attributs servent à distinguer les sujets, ressources et environnements. Il est par exemple impossible de définir un accès sur tous les « fichiers-dossiers patients » à moins d'ajouter un attribut sur tous les fichiers permettant de les distinguer ou dans chaque règle utilisée, spécifier l'ensemble des attributs la caractérisant.

2.2.5 Or-BAC

2.2.5.1 Introduction

Le modèle Or-BAC (Organization Based Access Control) a été défini en 2003 par Abou El Kalam et al. Il s'agit d'un modèle de contrôle d'accès basé sur les organisations.

Tout comme le modèle RBAC décrit précédemment (voir §2.2.3), Or-BAC s'appuie sur le principe de rôles pour lesquels on attribue un ensemble de privilèges. La différence par rapport à RBAC est qu'ici, l'ensemble des sujets, objets et actions sont structurés au niveau d'une organisation. De plus, Or-BAC permet d'exprimer des interdictions et des obligations, ce qui n'est pas le cas du modèle RBAC qui se limite aux permissions.

La structure de base du modèle Or-BAC est représentée à la Figure 15.

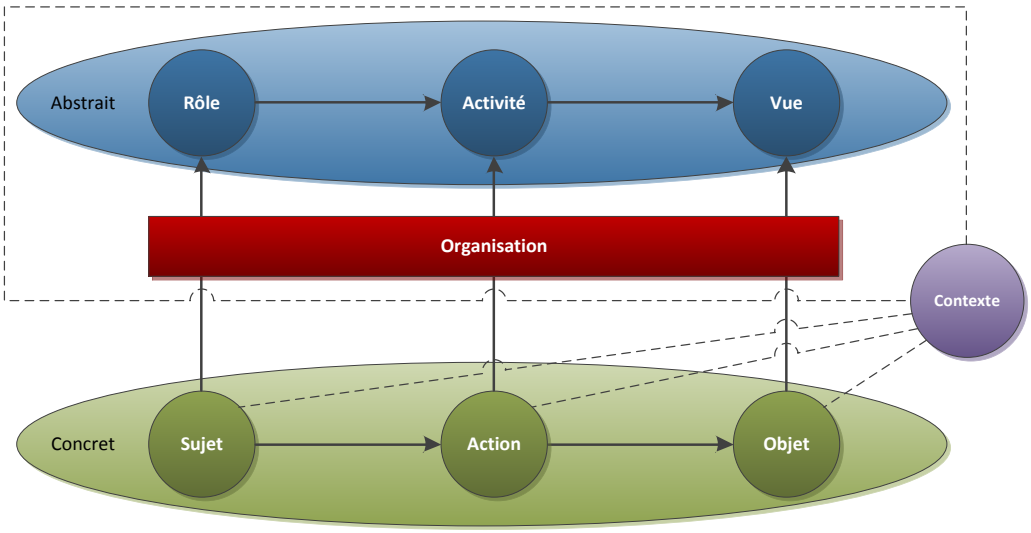

Figure 15 Structure du modèle Or-BAC (Equipe SERES, 2008)

Ce schéma fait apparaître une structure à deux niveaux : un niveau concret (sujet, action, objet) et un niveau abstrait (rôle, activité, vue). Cela a pour avantage de pouvoir spécifier au niveau abstrait une politique de sécurité qui sera indépendante de l'implémentation qui est réalisée au niveau concret.

« Ainsi, dans OrBAC, un rôle est un ensemble de sujets sur lesquels sont appliquées les mêmes règles de sécurité. Identiquement, une activité est un ensemble d'actions sur lesquelles sont appliquées les mêmes règles de sécurité. Une vue est une ensemble d'objets sur lesquels sont appliquées les mêmes règles de sécurité. » (Equipe SERES, 2008)

Le schéma représenté à la Figure 15 indique également que l'entité organisation est au centre du modèle et que celui-ci fait intervenir la notion de contexte.

Les éléments structurels du modèle sont décrits de manière détaillée dans la suite de ce chapitre :

- Le §2.2.5.2 traite de l'organisation ;
- Le §2.2.5.3 traite des entités et de leurs relations ;
- Le §2.2.5.4 traite de la notion de contexte ;
- Le §2.2.5.5 traite de la politique de sécurité à deux niveaux ;
- Et finalement le §2.2.5.6 traite de la gestion des conflits.

2.2.5.2 L'organisation

Comme expliqué précédemment, la notion d'organisation est au centre du modèle Or-BAC. Elle y est définie comme suit : « une organisation peut être vue comme un groupe

structuré d'entités actives, c'est-à-dire des sujets jouant certains rôles. » (Abou El Kalam, et al., 2003 p. 3)

Selon F. Cuppens et A. Miège, le modèle Or-BAC permet également de subdiviser une organisation en plusieurs sous-organisations disposant chacune de règles de sécurité spécifiques (Cuppens, et al., sd p. 3). Il est également possible de définir une politique de sécurité générique au niveau d'une organisation mère ; les sous-organisations hériteraient alors de la même politique de sécurité mais avec la possibilité de la modifier soit en ajoutant ou en supprimant des règles. Cela leur permettrait alors de définir leur propre politique de sécurité.

2.2.5.3 Les entités et leurs relations

2.2.5.3.1 Les sujets et les rôles

Selon Abou El Kalam et al., un sujet peut être soit une entité active (c'est-à-dire un utilisateur tel que « Jean », « Marc »,…), soit une organisation (« Département Informatique », « Service d'urgence de l'hôpital St Luc ») (Abou El Kalam, et al., 2003 p. 3).

Pour rappel, un rôle est un ensemble de sujets sur lesquels sont appliquées les mêmes règles de sécurité (Equipe SERES, 2008). L'entité « rôle » permet donc de structurer le lien entre les sujets et les organisations.

Les permissions obtenues par « Jean », « Marc », dépendent ainsi de leur rôle et de l'organisation dans laquelle ils exercent.

Dans le modèle, la relation qui lie le sujet s, le rôle r et l'organisation org est appelée $Habilite$. Dès lors $Habilite(org, s, r)$ signifie que l'organisation org habilite le sujet s à jouer le rôle r. Par exemple, la relation $Habilite(Y, Jean, informaticien)$ signifie que la société Y habilite Jean dans le rôle d'informaticien.

La relation $Habilite$ est représentée à la Figure 16.

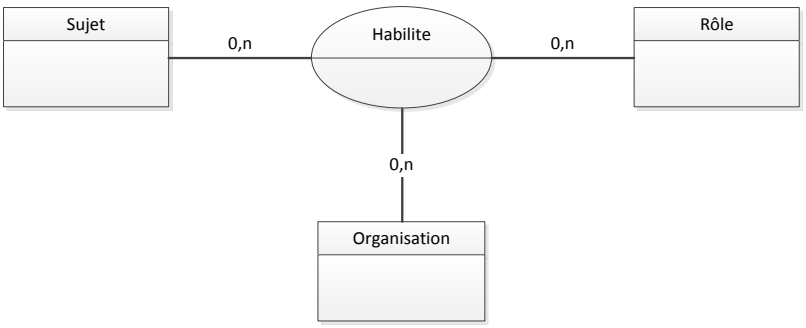

Figure 16 Relation Habilite (Abou El Kalam, et al., 2003 p. 4)

Tout comme dans le modèle RBAC (§2.2.3), Or-BAC prend en compte la notion de hiérarchisation. Cela permet l'héritage des permissions ou interdictions lorsque l'on descend dans la hiérarchie.

2.2.5.3.2 Les objets et les vues

Tout comme pour les sujets, il est également nécessaire de structurer les objets et de pouvoir en ajouter de nouveau au système. Dès lors, une entité comparable au rôle pour les objets est créée : l'entité *Vue*.

Pour rappel, une vue est définie comme un ensemble d'objets sur lesquels sont appliquées les mêmes règles de sécurité (Equipe SERES, 2008).

Dans Or-BAC, la relation qui lie l'objet o, la vue v et l'organisation org dans laquelle cette vue s'applique est appelée $Utilise$. Dès lors $Utilise(org, o, v)$ signifie que l'organisation org utilise l'objet o dans la vue v. Par exemple, la relation $Utilise(Y, listing.doc, dossier\ client)$ signifie que la société Y utilise listing.doc comme dossier client.

La relation $Utilise$ est représentée à la Figure 17.

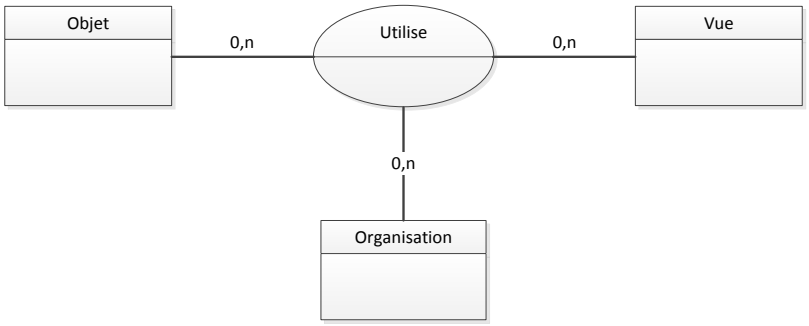

Figure 17 Relation Utilise (Abou El Kalam, et al., 2003 p. 4)

Comme pour les rôles, il est également possible de définir des hiérarchies de vues.

2.2.5.3.3 Les actions et les activités

De la même manière que pour les rôles et les vues, une nouvelle entité, utilisée comme abstraction des actions, est définie : l'entité *Activité*. Ainsi les activités sont définies comme un ensemble d'actions sur lesquelles sont appliquées les mêmes règles de sécurité (Equipe SERES, 2008).

Dans le modèle, la relation qui lie l'action a, l'activité act et l'organisation org dans laquelle cette action correspond à cette activité est appelée $Considère$. Dès lors $Considère(org, a, act)$ signifie que l'organisation org considère l'action a comme faisant

partie de l'activité *act*. Par exemple, la relation $Considère(Y, intervenir, dépannage)$ signifie que la société Y considère l'action intervenir comme un dépannage.

La relation *Considère* est représentée à la Figure 18.

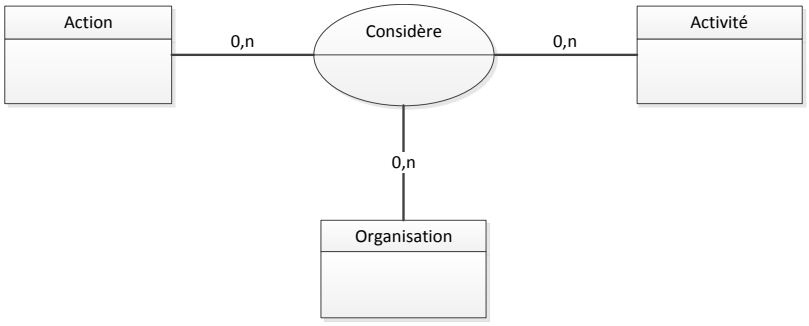

Figure 18 Relation Considère (Abou El Kalam, et al., 2003 p. 4)

Comme pour les rôles et les vues, il est également possible de définir des hiérarchies d'activités.

2.2.5.4 Les contextes

Abou El Kalam et al. utilisent les contextes pour spécifier les circonstances concrètes dans lesquelles les organisations accordent des permissions de réaliser des activités sur des vues (Abou El Kalam, et al., 2003 p. 5). Dans leur modèle, les entités Organisation, Sujet, Objet, Action et Contexte sont liées par une nouvelle relation appelée $Définit$. Dès lors $Définit(org, s, a, o, c)$ signifie qu'au sein de l'organisation org, le contexte c est vrai entre le sujet s, l'objet o et l'action a. Par exemple, la relation $Définit(Y, Jean, compléter, listing.doc, nouveau\ contrat)$ signifie que Jean peut compléter le fichier listing.doc en cas d'obtention d'un nouveau contrat client.

La relation $Définit$ est représentée à la Figure 19.

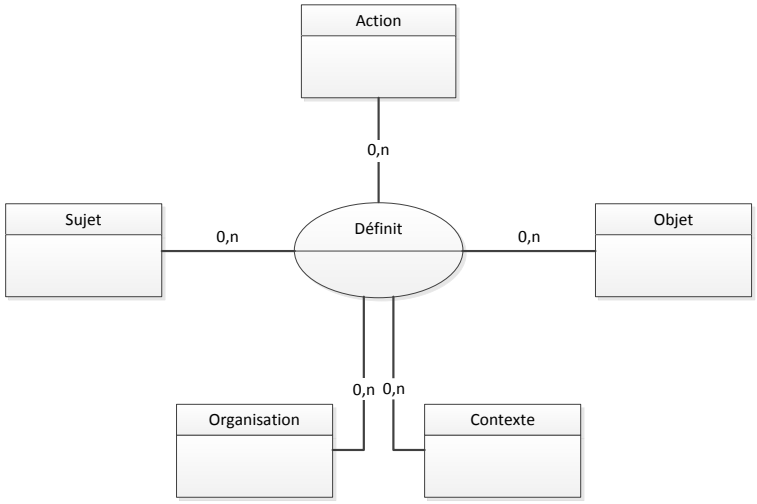

Figure 19 Relation Définit (Abou El Kalam, et al., 2003 p. 5)

Le concept de contexte offre l'avantage de pouvoir accorder des droits dans certaines circonstances.

Cinq types de contexte sont définis dans le modèle Or-BAC (Cuppens, et al., 2003 p. 4) :

1. Le contexte temporel : il limite la durée de validité d'une autorisation. Par exemple, Jean ne pourrait accéder au fichier listing.doc que pendant les heures de bureau ou uniquement certains jours de la semaine.
2. Le contexte spatial : il est lié à la situation spatiale du sujet. Selon F. Cuppens et A. Miège, il existe deux types de contexte spatial : le contexte physique correspondant à la localisation physique du sujet (ex : son bureau) et le contexte logique correspondant à la « localisation logique » dans laquelle le sujet se trouve (ex : l'ordinateur, le réseau,…) (Cuppens, et al., 2003 p. 5).
3. Le contexte déclaré par le sujet : il permet à un sujet d'obtenir certaines permissions en fonction du contexte dans lequel il se trouve. En déclarant ce contexte, les permissions sont accordées. Par exemple, Jean est amené à remplacer son collègue Marc qui est malade. En déclarant ce contexte, il pourrait avoir accès aux dossiers sur lesquels travaille Marc.
4. Le contexte prérequis : il permet d'accorder une permission à un sujet si certaines contraintes sont satisfaites. Par exemple, Jean serait autorisé à accéder à un dossier uniquement s'il s'agit d'un dossier dont il est le gestionnaire.
5. Le contexte provisionnel : il permet d'activer certaines permissions (obligations ou interdictions) en fonction de l'historique, c'est-à-dire des actions réalisées préalablement. Par exemple, Jean pourrait rencontrer un client de son collègue Marc actuellement malade mais avec l'obligation provisionnelle de lui transmettre le PV de son entrevue.

2.2.5.5 Politique de sécurité à 2 niveaux

Le modèle Or-BAC permet de diviser la politique de sécurité en deux niveaux. D'une part on a les sujets, les actions et les objets – ceux-ci appartenant à la partie dite concrète – et d'autre part leur abstraction en rôles, activités et vues représentent la couche abstraite.

Afin de connecter les organisations, rôles, vues, activités et contextes, le modèle fait appel à la relation *Permission* dans le niveau abstrait et son équivalente *Est_permis* dans le niveau concret reliant les sujets, actions, objets et contextes peut en être déduite. Dès lors $Permission(org, r, act, v, c)$ signifie que l'organisation org donne la permission à un rôle r de réaliser l'activité act sur une vue v dans le contexte c. Ces deux relations sont présentées sur la Figure 20 Politique de sécurité à deux niveaux.

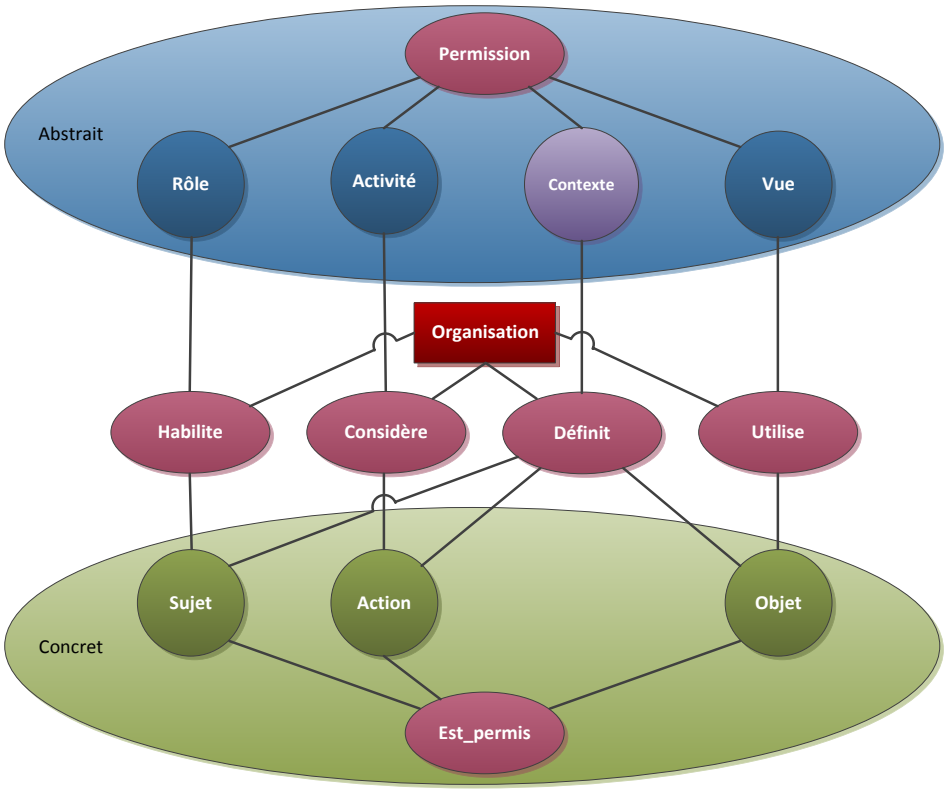

Figure 20 Politique de sécurité à deux niveaux

De la même manière, les relations *Interdiction* et *Obligation* (avec leur équivalent *Est_interdit* et *Est_obligé*) peuvent être définies (voir Figure 21).

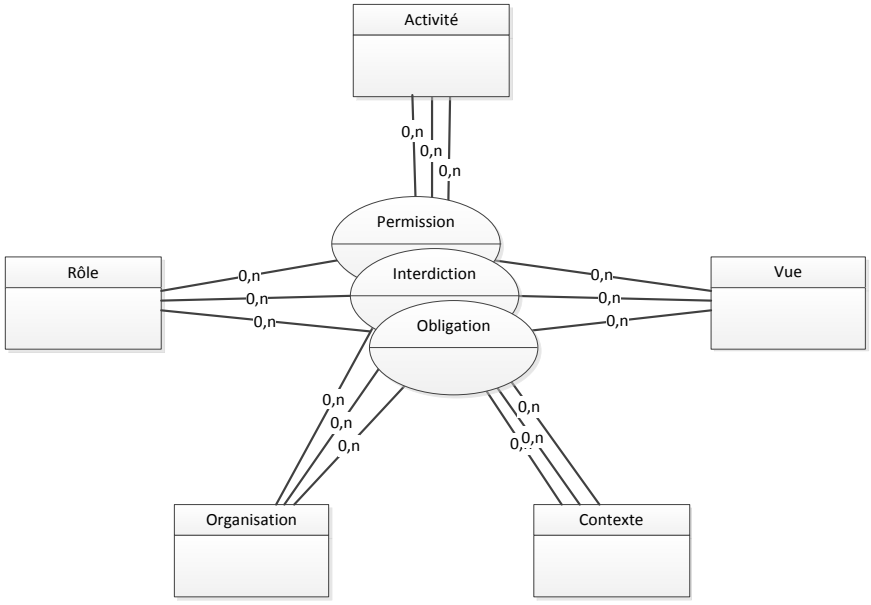

Figure 21 Relations Permission, Interdiction, Obligation (Cuppens, et al., 2003 p. 6)

L'objectif dans le modèle OrBAC est de rédiger la politique de sécurité à l'aide de permissions abstraites. Des permissions concrètes sont alors dérivées des permissions abstraites. La règle de dérivation pour la règle de *Permission*, définie par Abou El Kalam et al. (Abou El Kalam, et al., 2003 p. 8), est la suivante :

Soit $org = une\ organisation$

$r = un\ rôle$

$act = une\ activité$

$v = une\ vue$

$s = un\ sujet$

$o = un\ objet$

$c = un\ contexte$

Si $Permission(org, r, act, v, c) \wedge Habilite(org, s, r) \wedge Considère(org, a, act) \wedge Utilise(org, o, v) \wedge Définit(org, s, a, o, c)$

Alors $Est\_permis(s, a, o)$

Si l'organisation *org* dans le contexte *c* accorde la permission au rôle *r* de réaliser l'activité *act* sur la vue *v*, si *org* habilite le sujet *s* dans le rôle *r*; si *org* utilise l'objet *o* dans la vue *v*, si *org* considère l'action *a* comme faisant partie de l'activité *act* et si au sein de l'organisation *org* le contexte *c* est vrai entre *s*, *a* et *o*, alors il est permis au sujet *s* de réaliser l'action *a* sur l'objet *o*.

2.2.5.6 Les conflits

Comme expliqué par Abou El Kalam et al. dans leur article, les accès à un système sont réglementés par la politique de sécurité via les permissions, interdictions ou obligations (Abou El Kalam, et al., 2003 p. 5).

Il parait évident que l'utilisation à la fois de permissions, d'interdictions et d'obligations peut faire survenir des conflits. Du fait des deux couches du modèle OrBAC (abstraite et concrète), on pourrait avoir des conflits à chaque niveau. Par exemple :

- $Est\_permis(Marc, lire, fichier.pdf)$ et
  $Est\_interdit(Marc, lire, fichier.pdf)$

- $Permission(Département\_informatique, consultant, modifier, fichier\_db)$ et
  $Interdiction(Département\_informatique, consultant, modifier, fichier\_db)$

Le modèle Or-BAC permet de détecter et de gérer les conflits. Selon F. Cuppens et A. Miège, afin de régler les conflits potentiels, un niveau de priorité peut être introduit dans les règles de sécurité (Cuppens, et al., sd p. 8). La relation *Permission* (par analogie, de même pour *Interdiction* et *Obligation*) au niveau abstrait et son équivalente *Est_permis* (par analogie, de même pour *Est_interdit* et *Est_obligé*) au niveau concret sont donc redéfinies de la manière suivante :

Abstrait : $Permission(Organisation\ Rôle, Activité, Vue, Contexte, Niveau)$
Concret : $Est\_permis(Sujet, Action, Objet, Niveau)$

Si on reprend l'exemple précédent en y apportant les modifications :

- $Permission(Département_{informatique}, consultant, modifier, fichier\_db, 1)$ et
  $Interdiction(Département_{informatique}, consultant, modifier, fichier\_db, 3)$

Dans ce cas-ci, la permission l'emporte sur l'interdiction.

En cas de conflits (si par exemple le niveau de priorité de deux règles est identique), cela sera au concepteur de la politique d'intervenir pour soit supprimer ou ajouter des règles, soit pour changer les niveaux de priorités des règles conflictuelles.

2.3 Conclusion générale

Le présent chapitre avait pour objectif de décrire les principaux modèles de contrôle d'accès existants, à savoir les modèles DAC, MAC, RBAC, ABAC et Or-BAC.

Le modèle de contrôle d'accès discrétionnaire (DAC) offre la possibilité à tout utilisateur de définir les droits qu'il attribue aux autres utilisateurs. Cela permet un niveau de sécurité très élevé car pour chaque utilisateur est attribué un droit d'accès pour chaque objet qu'il « utilise ». Mais ce niveau de détails entraîne une complexité de mise en place et une difficulté de maintenance.

Le modèle de contrôle d'accès mandataire (MAC) apporte plus de souplesse que DAC en intégrant une notion de niveau de sécurité. En fonction du niveau de sécurité d'un utilisateur, celui-ci aura automatiquement accès à un certain nombre d'objets. Cette solution apporte une gestion simplifiée du contrôle d'accès mais selon le modèle utilisé (Bell La Padula – Biba), différents désavantages apparaissent tels que le non-respect de l'intégrité ou de la confidentialité. Une solution permet cependant de satisfaire en même temps ces deux désavantages mais sa mise en œuvre et son utilisation la rendent trop complexe.

Le modèle basé sur les rôles (RBAC) se rapproche plus du fonctionnement d'une entreprise où le contrôle d'accès n'est plus directement lié aux différents utilisateurs mais à des rôles attribués à ceux-ci. En fonction des rôles attribués, un utilisateur aura la permission d'effectuer diverses actions sur des objets. Les avantages de ce modèle concernent un rapprochement du mode de fonctionnement d'une entreprise – ce qui le rend facilement maintenable et représentatif de la réalité – et la possibilité de hiérarchiser l'ensemble, ce qui facilite également sa gestion. Néanmoins, la notion de rôle et hiérarchie complique la mise en place d'une structure de sécurité forte car il est difficile de limiter les accès pour un seul sujet faisant partie d'un groupe. Ce modèle nécessite une structure organisationnelle correcte pour pouvoir être mis en place.

Le modèle contextuel basé sur les attributs (ABAC) prend en compte des attributs pour les éléments du modèle, à savoir le sujet, l'objet et le contexte/environnement. Chaque attribut apporte une information supplémentaire qui peut être utilisée afin de prendre une décision lors d'une demande d'accès. Ces attributs permettent de bénéficier d'une granularité plus fine pour le contrôle d'accès que les précédents modèles mais en contrepartie, ils augmentent la complexité de mise en œuvre et de maintenance : quelle valeur donner à tel attribut afin d'autoriser un accès à un objet ?

Le dernier modèle analysé est le modèle Or-BAC dont la principale caractéristique est une politique à double niveau, à savoir un niveau concret se rapportant aux éléments de base de tout modèle (sujet, objet, action) et un niveau abstrait dans lequel chacune des entités possède un lien avec son homologue concret (rôle → sujet ; activité → action et vue → objet). Cette caractéristique permet la mise en place de règles abstraites, ayant chacune leur équivalence concrète, dont la gestion est largement facilitée. De même que pour le rôle dans RBAC permettant de structurer les utilisateurs selon certains critères, il est possible

dans Or-BAC de structurer les objets et les actions en un ensemble de vues et d'activités. Avec ce niveau d'abstraction, l'établissement de règles impliquant un rôle, une activité et une vue est plus aisé et représentatif de la réalité.

2.3.1 Concepts en commun

L'analyse de tous ces modèles a permis de mettre en évidence un ensemble commun d'entités présentes dans chacun d'eux. A l'exception des modèles ABAC et Or-BAC, tous se contentent des entités de base, à savoir un sujet, un objet et une opération. Les deux derniers modèles étudiés ont introduit la notion de contexte. Chaque modèle utilise une « notation » propre pour chacune des entités mais il est possible de les faire correspondre les unes aux autres (voir Tableau 2).

Tableau 2 Entités présentes dans les modèles de contrôle d'accès étudiés

	Sujet	Objet	Opération	Contexte
DAC	x	x	x	-
MAC	x	x	x	-
RBAC	x	x	x	-
ABAC	x	x	x	x
Or-BAC	x	x	x	x

A l'aide de ce tableau, on peut établir un modèle générique satisfaisant à chacun d'eux. Pour ce faire, il est nécessaire dans un premier temps de définir de manière générique chacune des entités entrant en compte dans l'élaboration de ce modèle.

2.3.1.1 Sujet

Le sujet est l'entité qui dispose d'un certain nombre d'autorisations lui permettant de réaliser une opération sur un objet dans un contexte particulier.

Dans les modèles DAC et MAC, cette entité est présente sous la notion portant le même nom.

Dans le modèle RBAC, cette entité est présente sous le nom d' « Utilisateur » pour lequel il est possible d'associer des rôles.

Dans le modèle ABAC, le sujet est défini par des attributs qui définissent son identité et ses caractéristiques. Le rôle du sujet peut aussi être considéré comme un attribut.

Dans le modèle Or-BAC, le sujet peut être soit une entité active, c'est-à-dire un utilisateur, soit une organisation pouvant être vue comme étant un groupe organisé d'entités actives, c'est-à-dire de sujets jouant certains rôles. Le sujet représente la partie concrète du modèle, son équivalent abstrait est donné par le rôle pour lequel il joue.

2.3.1.2 Objet

L'objet correspond à la ressource sur laquelle un sujet effectue une opération.

Dans les modèles DAC et MAC, cette entité est présente sous la notion portant le même nom.

Dans le modèle ABAC, la notion d'objet est reprise sous le concept de « Ressource » et correspond à la ressource pour laquelle on veut obtenir un droit d'accès. De même que pour l'entité « Sujet », il est défini par un ensemble d'attributs permettant de le caractériser.

Dans le modèle OrBAC, l'objet représente les entités non actives, c'est-à-dire toutes les ressources de l'organisation, comme les fichiers, les courriers électroniques, les formulaires,… Le sujet représente la partie concrète du modèle, son équivalent abstrait est donné par la vue auquel il appartient.

2.3.1.3 Opération

L'opération représente l'action que peut faire un sujet sur un objet.

Dans le modèle DAC, cette entité est présente sous le nom de « Droit d'accès ».

Dans le modèle MAC, elle porte le nom d' « Attributs d'accès ».

Dans le modèle RBAC, elle est appelée « Transaction », représentée par une combinaison d'une transformation et d'un objet sur lequel s'applique celle-ci.

Dans le modèle ABAC, les opérations sont définies à l'aide de règles présentes dans une base de règles[1]. On accorde un accès à un sujet sur un objet dans un contexte s'il existe une règle définie par l'administrateur pouvant être satisfaite à partir des attributs liés au triplet sujet, objet, contexte. Dans le cas contraire, l'accès est interdit.

Le modèle OrBAC supporte également le concept d'opération. Pour ce faire, une relation « Permission » est créée dans la couche abstraite reliant à chaque fois les « Rôles », les « Activités » et les « Vues ». La règle de dérivation permet d'obtenir son équivalent « Est_permis » dans la couche concrète reliant les « Sujets », les « Actions » et les « Objets ». Cette relation spécifie de manière claire quel est le type d'opération que peut effectuer un sujet sur un objet.

2.3.1.4 Contexte

Le contexte peut être vu comme l'ensemble des informations permettant de décrire l'environnement actuel, au moment où l'action se déroule.

[1] Appelée « Policy Store » dans le modèle ABAC

Dans le modèle ABAC, le contexte correspond aux attributs environnementaux. L'environnement peut en effet être décrit par des informations telles que l'heure, la date, la localisation,… Il s'agit en fin de compte d'attributs qui ne sont associés ni aux sujets, ni aux objets mais qui peuvent également jouer un rôle quant à la prise de décision.

Dans le modèle Or-BAC, la notion de contexte permet au sujet d'effectuer des opérations dans certaines circonstances telles que l'heure actuelle, la localisation du sujet, …

2.3.2 Modèle générique

Le modèle générique pouvant s'appliquer aux modèles de contrôle d'accès analysés dans ce chapitre est représenté de la manière suivante (sous forme de diagramme de classe) :

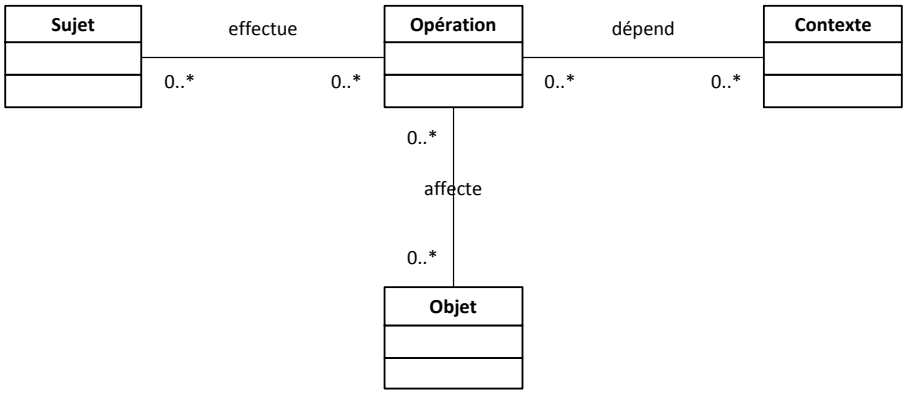

Figure 22 Modèle générique des modèles de contrôles d'accès

Dans ce diagramme, les relations entre les entités sont de cardinalité 0..* car un sujet a la possibilité d'effectuer un certain nombre d'opérations, allant de 0 (aucune opération) à * (un nombre fini d'opérations). De même, l'opération affecte un certain nombre d'objets, allant également de 0 à *. Et ces opérations dépendent d'un certain nombre de valeurs prises par le contexte (heure, date, …).

Pour satisfaire le modèle ABAC, il faut y inclure différents attributs aux entités « Sujet », « Objet » et « Contexte ». Le modèle RBAC sera satisfait par la présence d'un attribut de rôle pour l'entité « Sujet ». De même, Or-BAC, quant à lui, sera également satisfait dès lors qu'il existe en plus de l'attribut rôle pour l'entité « Sujet », un attribut de type vue pour l'entité Objet et un attribut activité pour l'entité « Action ».

3 Systèmes de gestion des droits numériques

3.1 Introduction

Dans le passé, avant l'apparition des supports numériques, il était aisé de devenir propriétaire d'un support musical ou audiovisuel : il suffisait d'acheter l'original. Une fois acquis, le support pouvait être utilisé à souhait ; il pouvait même être copié. A cette époque, la copie de données se faisait avec une importante perte de qualité et il était facile de distinguer une copie de l'original.

Aujourd'hui, la situation est bien différente de l'époque. Avec l'apparition et le développement considérable des technologies numériques, il est maintenant possible de réaliser un nombre illimité de copies d'un support à moindre coût et ce, sans aucune perte de qualité. Il est donc devenu impératif de protéger les œuvres créées afin de faire valoir les droits des auteurs.

« En Europe, le cadre légal établissant ces droits est énoncé dans la Directive 2001/29/EC du 22 mai 2001 sur l'harmonisation de certains aspects du droit d'auteur et des droits voisins dans la société de l'information. » (European Commission, 2010) [Notre traduction]

Cette Directive traite de trois droits principaux :

- Le droit de reproduction dans son article 2 ;
- Le droit de communication d'œuvre au public et droit de mettre à la disposition du public d'autres objets protégés (article 3) ;
- Le droit de distribution (article 4).

Afin de protéger leurs œuvres, les propriétaires et distributeurs cherchent donc des moyens de stopper cette diffusion illégale de données. Un des moyens utilisé pour contrer ce problème est de développer des technologies permettant de rendre la copie difficile, voire impossible. C'est ce que l'on appelle les « technologies de gestion des droits numériques » (ou Data Right Management – DRM technologies).

L'Europe participe au développement des DRMs au travers de divers programmes qu'elle finance (Commission of the European communities, 2002).

Le présent chapitre a pour objectifs de définir et de décrire de manière générale les systèmes de gestion de droits numériques (ci-après appelés « systèmes DRM »). Dans ces systèmes, les droits des utilisateurs sont définis sous forme de « langage d'expression de droits » (ou Rights Expression Language – REL). Les principaux REL existants seront également décrits.

Le présent chapitre traitera également de la politique de contrôle de gestion XACML. La combinaison de ce modèle de gestion avec un système de contrôle d'accès (ex : RBAC) forme en effet un système complet d'expression de droits.

Il est à noter que le XACML et les RELs peuvent être structurés en deux parties :

- Un « modèle de données » regroupant l'ensemble de toutes les informations statiques, c'est-à-dire l'ensemble des informations représentant la structure de base.
- Une partie « protocole » qui concerne toutes les transactions possibles entre les entités du modèle de données.

Etant donné que seul le modèle de données sera utilisé pour la définition de notre modèle générique, les aspects concernant le protocole sortent du cadre du présent mémoire et ne seront donc pas analysés ici.

3.2 Description générale des systèmes DRM

3.2.1 Définition et objectifs

Comme expliqué dans la précédente section (§3.1), les systèmes DRM ont été développés afin de protéger les droits d'auteur. En vue d'atteindre cet objectif, les DRM se concentrent sur deux principaux aspects :

- D'une part, la *gestion des droits digitaux* qui concerne l'identification, la description du contenu et spécifie les règles d'utilisation et de distribution de ce contenu ;
- D'autre part, la *gestion digitale de droits* qui concerne la sécurisation du contenu et la mise en application des droits. Il convient ici de s'assurer que le contenu ne puisse être utilisé que selon les termes définis dans les règles d'utilisation.

3.2.2 Principe de fonctionnement d'un système DRM

3.2.2.1 Fonctionnement général

Le principe de fonctionnement d'un système DRM de base est illustré à la Figure 23.

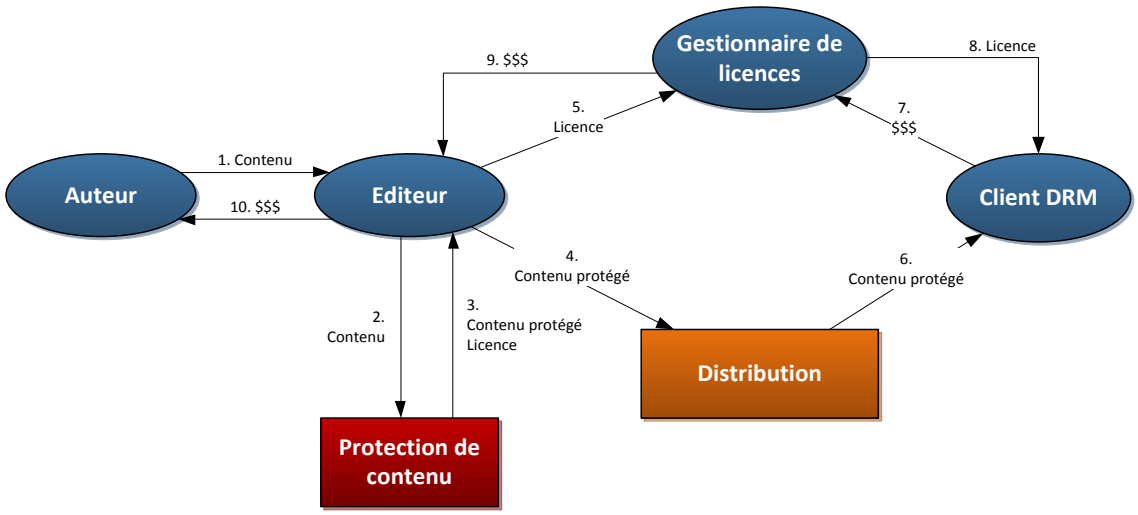

Figure 23 Principe de fonctionnement d'un système DRM de base (Ku, et al., 2004 p. 4)

Comme l'indique la figure, le fonctionnement d'un DRM met en jeu quatre principaux acteurs :

- L'« *Auteur* » qui est le propriétaire du contenu et dispose de tous les droits sur celui-ci ;
- L' « *Editeur* » qui gère la protection du contenu et envoie le contenu sécurisé vers le réseau de distribution et les licences vers le « *Gestionnaire de licences* » ;
- Le *« Gestionnaire de licences »* qui gère l'ensemble des transactions auxquelles est soumise la licence. Une licence contient les informations relatives aux permissions (droits, termes et conditions) pour l'utilisation du contenu.
- Le *« Client DRM »* qui est l'application qui gère le contenu DRM dans le sens où il s'assure que les permissions accordées à l'utilisateur sont bien respectées tout en intégrant un certain nombre de limitations préalablement définies par le titulaire de droits.

Selon W. Ku et C.-H. Chi (Ku, et al., 2004 pp. 4-5), le principe de fonctionnement d'un système DRM est le suivant :

1. L'« *Auteur* » envoie son contenu à l'« *Editeur* » et définit avec lui les conditions d'utilisation ainsi que les droits d'auteur ;
2. La deuxième étape concerne la protection du contenu. Elle consiste notamment à intégrer des métadonnées décrivant le contenu et à définir des droits et règles d'utilisation. Cette étape est détaillée dans le §3.2.2.2.
3. L'« *Editeur* » reçoit ensuite le contenu protégé ainsi que la (ou les) licence(s) associée(s). Sans celle(s)-ci, l'utilisateur n'a pas accès au contenu ; la licence contient en effet la clé permettant de lire le contenu protégé.
4. Une fois ces étapes réalisées, le contenu peut être « distribué » au travers différents réseaux de distribution (internet, CD,…).
5. La (ou les) licence(s) est (sont) transmise(s) au « *Gestionnaire de licences* ».
6. Le « *Client DRM* » récupère le contenu protégé d'un réseau de distribution et identifie la licence nécessaire pour accéder au contenu. Il identifie également le « *Gestionnaire de licences* » qui pourrait fournir la licence requise.
7. Si l'utilisateur du contenu ne possède pas de licence valide, il peut l'acquérir moyennant payement. Le « *Client DRM* » sert alors d'intermédiaire avec le « *Gestionnaire de licences* ».
8. Une fois le payement reçu, le « *Gestionnaire de licences* » transmet une licence au « *Client DRM* » qui accorde certains droits à l'utilisateur (dépendant de ce pourquoi il a payé).
9. Le « *Gestionnaire de licences* » paye les droits d'auteur à l'« *Editeur* ».
10. Ce dernier rémunère ensuite l'« *Auteur*. »

3.2.2.2 Mécanisme de protection de contenu

Le mécanisme de protection de contenu au sein d'un système DRM est illustré à la Figure 24.

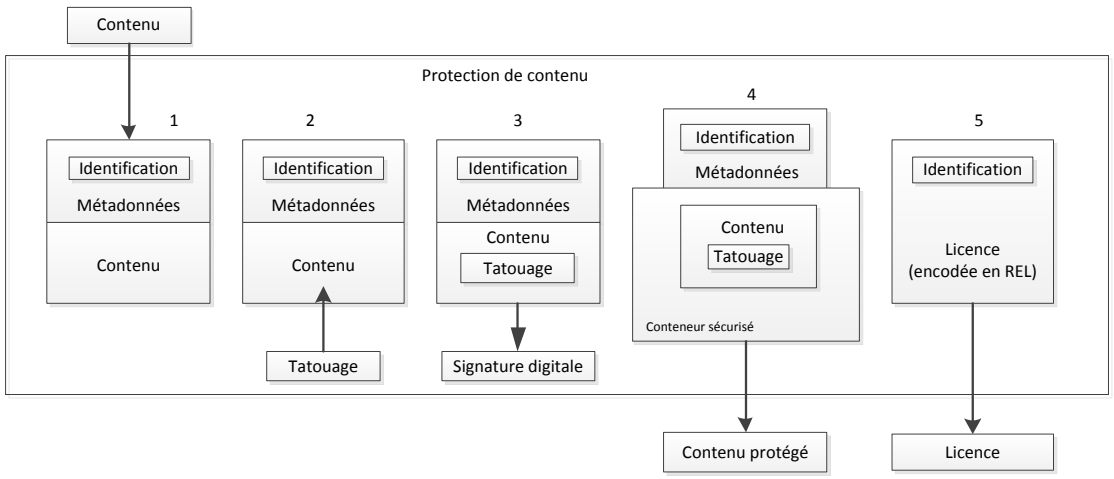

Figure 24 Procédé de protection de contenu (Ku, et al., 2004 p. 8)

W. Ku et C.-H. Chi (Ku, et al., 2004 p. 8) décrivent le mécanisme de protection d'un contenu de la manière suivante :

1. Le contenu est associé à un *identifiant unique* et à des *métadonnées*. Celles-ci permettent d'ajouter des informations descriptives au contenu telles que l'auteur, la publication, l'année,… ;
2. Un *tatouage numérique* est éventuellement inséré au contenu afin de prouver son authenticité (exemple : altération d'une image suite à une modification d'une séquence de bits invisible par l'œil humain mais permettant d'identifier l'auteur) ;
3. Une *signature digitale* est générée à partir du contenu. Il s'agit d'un identifiant unique qui, tout comme le tatouage numérique, est utilisé pour l'authentification.
4. Le contenu est protégé par un *conteneur sécurisé* qui permet d'empêcher tout accès non autorisé (exemple : le format de fichier .wwf empêchant toute impression (WWF, 2010)) ;
5. Une licence définissant les droits et conditions d'utilisation du contenu est encodée dans un langage d'expression de droits (ou REL).

3.2.3 Principales exigences d'un système DRM

Afin d'avoir un système DRM performant, un certain nombre d'exigences doivent être satisfaites. Celles-ci sont brièvement présentées ici.

Les principales exigences requises concernent la sécurité. Un système DRM doit notamment assurer que seuls les utilisateurs autorisés ont accès à un contenu et obligatoirement selon les termes et conditions définis dans la licence. Si tel n'était pas le cas, le système DRM doit être capable de détecter les fraudes et de révoquer les accès non autorisés. Un système DRM doit également assurer que les licences ne soient distribuées qu'aux utilisateurs autorisés et que les conditions d'utilisation qui y sont définies ne puissent pas être modifiées.

De manière plus générale, pour avoir un système DRM performant, il faut également que les flux d'informations entre les différents acteurs définis à la Figure 23 soient limités.

3.2.4 Principaux systèmes DRM existants

Les principaux systèmes DRM existants, à savoir FairPlay, Helix, OMA, Adobe Flash Access 2.0, PlayReady et DRM-X sont décrits pour information en annexe. D'autres systèmes DRM plus récents tels que BD+ et Ultraviolet y sont également repris.

3.3 Expression des droits

3.3.1 Introduction

Comme expliqué précédemment, l'expression des droits et des règles d'utilisation dans les systèmes DRM se fait sous forme de langage d'expression de droits, plus communément appelés les « RELs ».

L'objectif du présent chapitre est de définir et de décrire de manière générale la structure de ces langages d'expression de droits et de présenter de manière détaillée les principaux RELs existants, à savoir l'ODRL, le XrML et le MPEG-21 REL.

3.3.2 Structure des RELs

Les RELs sont utilisés pour décrire les opérations que peut effectuer un sujet sur un objet dans un contexte particulier.

De manière générale, les langages d'expression de droits sont composés d'une syntaxe permettant de représenter des expressions de droits et d'un dictionnaire de données décrivant l'ensemble des éléments utilisés dans celle-ci.

L'expression de droits repose sur l'utilisation de quatre principaux composants (Chong, et al., 2003 p. 2) :

- Le sujet qui est l'acteur à l'origine d'une opération sur un objet ;
- L'objet qui est le contenu sur lequel est effectuée l'opération du sujet ;
- L'opération qui définit ce qu'un sujet peut faire sur un objet. Elle peut être de deux types : soit considérée comme un « droit », c'est-à-dire une action qui peut être directement effectuée sur un objet ; soit comme une « obligation », c'est-à-dire une action qui doit précéder ou succéder une autre opération ;
- La contrainte qui décrit dans quels cas/conditions l'opération peut être réalisée.

Un exemple dans lequel tous ces composants entrent en compte serait le suivant : pour écouter une musique sans publicité et librement, Alice doit payer 1€ de manière anticipée.

Dans cet exemple, le sujet est « Alice », les objets sont « une musique » et « 1€ », le droit est « écouter », l'obligation est « payer » et les contraintes sont « sans publicité », « librement » et « anticipé ».

Avec cet exemple, on remarque que :

- Le sujet est décrit à l'aide d'un nom (ex : Alice). Il doit également être capable de distinguer les différents sujets en fonction de leur type à l'aide de noms tels que : créateur, distributeur, utilisateur final,…

- L'objet est également décrit à l'aide d'un nom qui peut correspondre par exemple au titre de la chanson, au nom de l'auteur,… Il doit également être capable de supporter tout type d'objet comme des fichiers multimédias (audio, vidéo) ou des documents (texte, classeurs,…).
- Le REL dispose de deux types d'opérations tel qu'expliqué ci-dessus.
  Les droits peuvent également être catégorisés en fonction de leur utilisation :
    - L'ensemble des droits selon lesquels l'objet peut être consommé (ex : afficher, imprimer, lire, …) ;
    - L'ensemble des droits selon lesquels l'objet peut être réutilisé (ex : modifier, …) ;
    - L'ensemble des droits selon lesquels le droit d'un sujet sur un objet peut être transféré (ex : vendre, louer, prêter, …) ;
    - L'ensemble des droits utilisés pour la gestion des objets (ex : déplacer, dupliquer, supprimer, …).

  Les obligations correspondent aux opérations permettant d'autoriser ou d'activer un ou des droits sur un objet (ex : payer, s'enregistrer, …)
- Les contraintes sont représentées en fonction de différents critères :
    - Temporel : limitant l'opération dans le temps (ex : date, heure, intervalle de temps) ;
    - Quantitatif : limitant l'opération sur un nombre fini de possibilité (ex : compteur d'utilisation, intervalle de page imprimable, …) ;
    - Environnemental : limitant l'opération sur la localisation (ex : emplacement physique tel qu'un pays, ville ou emplacement logique défini par exemple à l'aide de l'adresse internet (ip)) ;
    - Aspect : liée aux caractéristiques de l'objet (ex : qualité, format de fichier, présence de tatouage numérique, …).

Ces composants mis en relation sous forme de diagramme de classe sont présentés à la Figure 25.

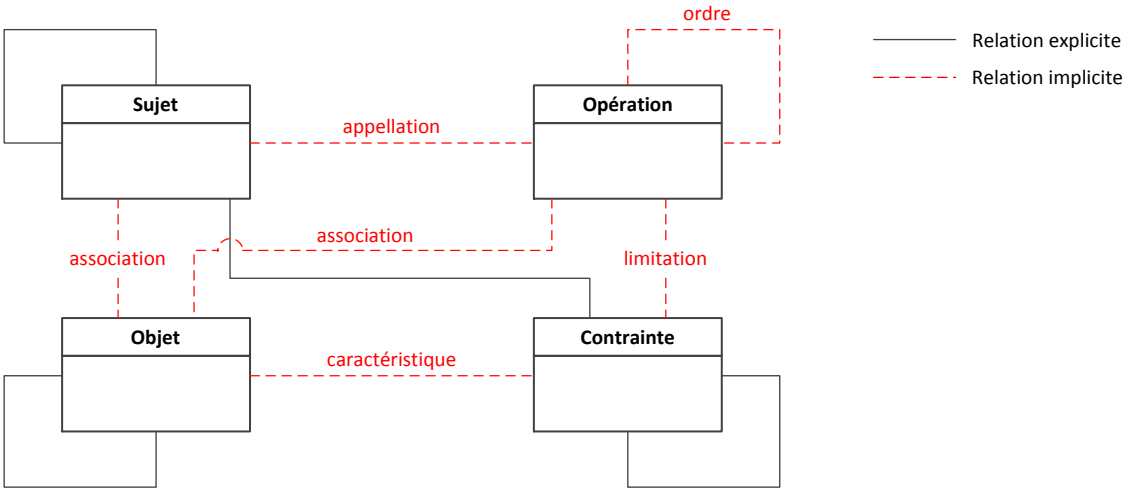

Figure 25 Principaux concepts et leurs relations dans un REL (Chong, et al., 2003 p. 2)

Comme expliqué par Chong et al. (Chong, et al., 2003 p. 3), un langage d'expression de droits doit spécifier des relations, qu'elles soient explicites ou implicites, entre les différents composants. Selon lui, il existe 5 types de relations explicites[2] :

- La relation d'*ordre* concerne les relations de type « opération-opération » et décrit comment celles-ci sont liées. Par exemple « paye avant d'écouter » peut être vu comme une obligation préalable alors que « écoute puis paye » est plutôt une obligation de conséquence. L'ordre peut également être de deux types : total ou partiel. Dans le premier cas, cela spécifie complètement l'ordre dans lequel les opérations doivent se succéder (ex : s'enregistrer, payer, écouter). Dans le deuxième, les opérations peuvent être réalisées sans tenir compte de l'ordre indiqué (ex : « payer, écouter » autorise l'écoute avant le payement) ;
- La relation d'*association* concerne les relations de type « sujet-objet » et « opération-objet » (ex : écouter une musique) ;
- La relation d'*appellation* concerne les relations de type « sujet-opération ». Elle spécifie le nom de l'opération qu'un sujet peut effectuer (ex : Alice *écoute* de la musique) ;
- La relation de *limitation* concerne les relations de type « opération-contrainte ». Elle implique que les opérations soient restreintes par un certain nombre de contraintes (ex : Alice *écoute* de la musique *sans publicité*) ;
- La relation de *caractéristique* concerne les relations de type « objet-contrainte ». Elle permet de décrire plus en détail l'objet sur lequel une opération est effectuée (ex : une musique *sans perte de qualité*).

[2] Les deux premières relations, à savoir « ordre » et « association » ont été identifiées par David Parrott (Parrott, 2001).

Chong et al. définissent également plusieurs relations implicites (indirectes) incluant les relations de type « sujet-sujet », « sujet-contrainte », « objet-objet » et « contrainte-contrainte ».

En modifiant légèrement l'exemple de départ :

> « Alice doit payer 1 € pour écouter une musique pendant 1 an dans un cadre privé. »

On peut retrouver les relations explicites suivantes :

- Relation d'*ordre total* : « payer – écouter »
- Relation d'*association* : « payer – 1€ »
  « écouter – musique »
- Relation d'*appellation* : « Alice – payer »
  « Alice – écouter »
- Relation de *limitation* : « écouter – (1 an – cadre privé) »

La relation implicite est la suivante :

- Relation « contrainte – contrainte » : « 1 an – cadre privé »

Cette relation est implicite car rien ne permet de lier ces 2 contraintes au départ. Elles sont utilisées conjointement dans la relation de limitation.

Un autre exemple serait celui-ci :

> « Alice a besoin de Bob pour prouver son identité afin qu'elle puisse louer un livre électronique. »

On peut diviser cet exemple en 2 parties :

- Alice souhaite louer un livre électronique [à condition que …]
- Bob prouve l'identité d'Alice

Il existe 2 relations implicites dans cet exemple. La première concerne Alice et Bob (« sujet-sujet ») car il n'y a pas de lien qui les unit au départ ; Alice a seulement besoin que Bob effectue son opération. La suivante concerne Alice et la contrainte « prouver son identité » car Alice ne sait satisfaire à cette contrainte elle-même. Pour ce faire, elle a besoin de l'opération de Bob. Mais cette contrainte est tout de même liée à son opération. D'une manière ou d'une autre, Alice est liée à celle-ci mais de manière indirecte.

3.3.3 ODRL

3.3.3.1 Remarque préalable

L'analyse du modèle ODRL abordée dans la présente section s'est basée exclusivement sur la publication officielle ODRL v.1.1. de l' « *IPR Systems Pty Ltd* » datant de 2002 disponible sur le site web suivant : http://odrl.net/1.1/ODRL-11.pdf.

Dans un souci de simplification et d'aisance de lecture du texte, dans les parties descriptives des composants (§3.3.3.4 et §3.3.3.5), les références du document source sont indiquées en italique au début de chaque paragraphe avec le numéro d'identification de la page concernée par l'analyse.

3.3.3.2 Introduction

ODRL (Open Digital Rights Language) est un des principaux langages d'expression de droits existants (ODRL Initiative, 2000-2011). Il est basé sur le langage de balisage XML (eXtensible Markup Language)[3]. L'ODRL a été créé en 2001 dans l'esprit « open source ». Sa dernière mise à jour a été effectuée en 2002, il s'agit de la version 1.1. Une nouvelle version 2.0 est actuellement en cours d'élaboration.

Etant donné que la version 2.0 n'est pas encore aboutie, notre analyse s'est basée sur la version 1.1. Dans cette version, deux schémas XML ont été définis, l'un définissant la syntaxe et l'autre le dictionnaire de données (IPR Systems Pty Ltd, 2002 p. 40). L'utilisation du XML permet au modèle ODRL de base d'être mis à jour/étendu facilement en y ajoutant simplement de nouvelles entrées dans le dictionnaire de données.

3.3.3.3 Le modèle simplifié

Le mémoire ne traitant que de la partie destinée au contrôle d'accès, le modèle ODRL a été simplifié afin de ne prendre en compte que le modèle de données qui, pour rappel, regroupe les données statiques. Dès lors, les entités définies dans ODRL en tant que protocole qui, pour rappel, concerne les échanges tels que le protocole d'offre, d'accord et de révocation ont été retirées dans le modèle simplifié présenté ci-après. De même, les aspects relatifs à la sécurité, à savoir les mécanismes d'encryption, de vérification de signature digitale ne sont pas utilisés pour l'élaboration du modèle générique final ; c'est pourquoi ils ne figurent pas non plus dans le modèle ODRL simplifié présenté ci-dessous à la Figure 26 (sous forme de diagramme de classe).

[3] Les langages de balisage utilisent un ensemble de balises afin d'identifier/délimiter les différents composants d'un document, ce qui permet à chacun d'entre eux d'être formaté, affiché et utilisé de manière appropriée.

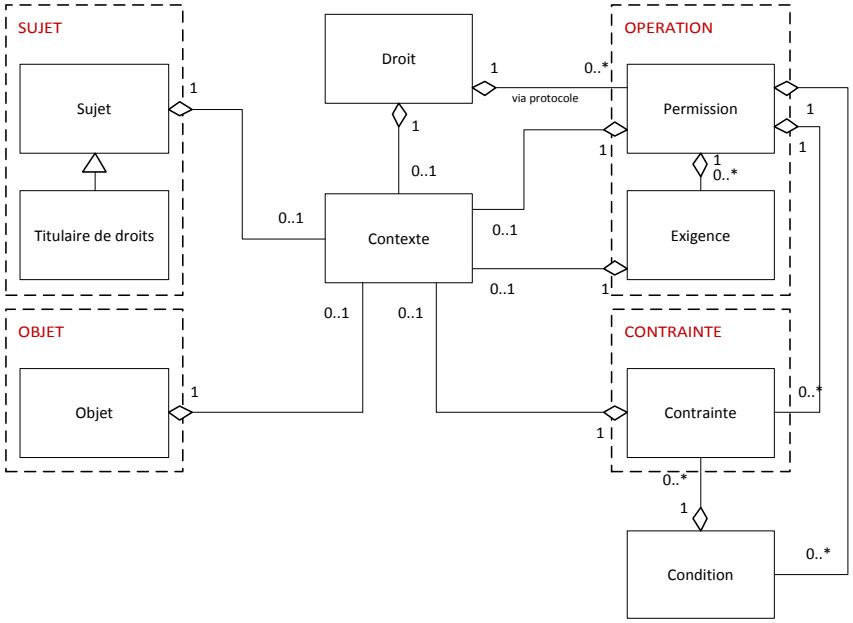

Figure 26 Modèle ODRL simplifié (IPR Systems Pty Ltd, 2002 p. 12)

Comme l'indique la figure, le modèle reprend les 4 composants de base des langages d'expression de droits définis au §3.3.2, soit le sujet, l'objet, l'opération et la contrainte. De plus, il intègre la notion de condition et de contexte. Tous ces composants et leurs relations sont décrits en détails ci-après.

3.3.3.4 Les 4 composants de base

3.3.3.4.1 Sujet

Source : (IPR Systems Pty Ltd, 2002 pp. 5, 18)

Le « Sujet » concerne les utilisateurs finaux et les propriétaires de droits. Ils peuvent être des humains, des organisations et des rôles définis. Les utilisateurs finaux sont généralement les consommateurs de l'« Objet ». Les titulaires de droits sont généralement les sujets qui ont joué un rôle dans la création, la production, la distribution de celui-ci et qui peuvent faire valoir une certaine propriété sur l'objet et/ou ses permissions.

Les titulaires de droits peuvent aussi percevoir des droits d'auteur. Le modèle ODRL des titulaires de droits est repris à la Figure 27.

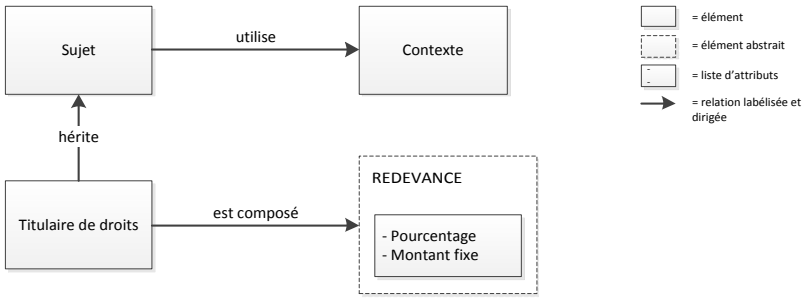

Figure 27 Concept titulaire de droits du modèle ODRL (IPR Systems Pty Ltd, 2002 p. 17)

Comme l'indique la figure ci-dessus, l'entité « Titulaire de droits » est composée d'une entité abstraite « Redevance » permettant uniquement de regrouper sur le schéma les informations de :

- « Pourcentage » : qui indique le pourcentage à faire valoir au titulaire de droits sur le montant d'une transaction réalisée par un sujet sur un objet ;
- « Montant Fixe » : qui indique le montant fixe à faire valoir au titulaire de droits sur le montant d'une transaction réalisée par un sujet sur un objet.

Un titulaire de droits peut contenir un ou plusieurs sujets (dans le cas de rémunération multiple comme par exemple lorsqu'un contenu a été créé par plusieurs auteurs) et peut être associé à un ou plusieurs objets.

L'entité « Sujet » est associée au « Contexte » qui regroupe l'ensemble des informations additionnelles permettant de le décrire.

3.3.3.4.2 Objet

Source : (IPR Systems Pty Ltd, 2002 p. 4)

L'entité « Objet » concerne les contenus physiques ou digitaux. Les objets doivent être identifiés de manière unique afin de pouvoir être différenciés les uns des autres (à l'aide de l'entité « Contexte »).

Le modèle ODRL des objets est repris à la Figure 28.

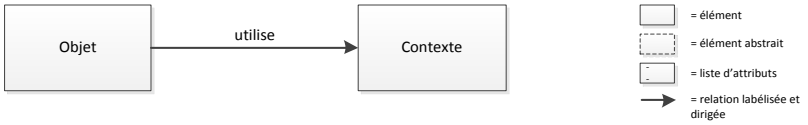

Figure 28 Concept objet du modèle ODRL

3.3.3.4.3 Opération

Source : (IPR Systems Pty Ltd, 2002 pp. 8,9,14,15)

L'entité « Opération » concerne les droits et les obligations dont un sujet dispose sur un objet. Le modèle défini par ODRL permet l'expression de droits grâce à l'entité

« Permission » et l'expression d'obligations, qui sont vues comme des pré-conditions devant être rencontrées pour obtenir les permissions associées, grâce à l'entité « Exigence ».

Le modèle ODRL des permissions est repris à la Figure 29.

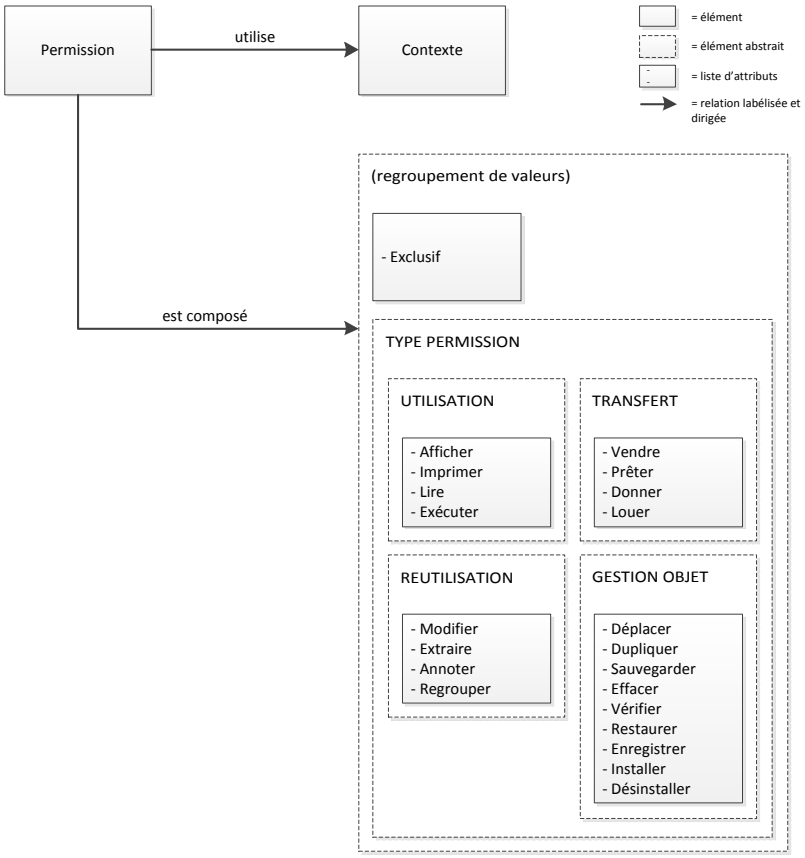

Figure 29 Concept permission du modèle ODRL (IPR Systems Pty Ltd, 2002 p. 8)

L'entité « Permission » est composée d'un ensemble de quatre entités abstraites qui, pour les mêmes raisons que celles précédemment citées (dans l'entité « Sujet »), servent uniquement à regrouper sur le schéma des permissions similaires. Ces quatre entités sont :

- L'entité « Utilisation » qui indique un ensemble de moyens selon lesquels l'objet peut être consommé (ex : afficher, imprimer, lire et exécuter) ;
- L'entité « Réutilisation » qui donne un ensemble d'opérations permettant à l'objet (ou une partie de celui-lui) d'être réutilisé (modifier, extraire, annoter, regrouper) ;
- L'entité « Transfert » qui indique un ensemble de procédures selon lesquelles les droits sur un objet peuvent être transférés (ex : vendre, prêter, donner, louer) ;
- L'entité « Gestion objet » qui indique un ensemble d'opérations de gestion de contenus digitaux (ex : déplacer, dupliquer, sauvegarder, effacer, vérifier, …).

En plus de ces quatre entités abstraites, les permissions supportent également :

- Un attribut « Exclusif » qui indique si la permission accordée est limitée à des sujets prédéterminés (ex : suite à un accord entre un sujet et un titulaire de droits, une permission a été créée spécifiquement pour ce sujet ; il a donc l'exclusivité sur cette permission) ;
- L'entité « Contexte » dont le but est d'assigner des identifiants uniques à une permission ou à un ensemble de permission.

Une permission peut être associée à zéro ou plusieurs contraintes, conditions ou exigences. Une permission qui n'est spécifiée dans aucune expression de droits n'est pas accordée.

Le modèle ODRL des exigences est repris à la Figure 30.

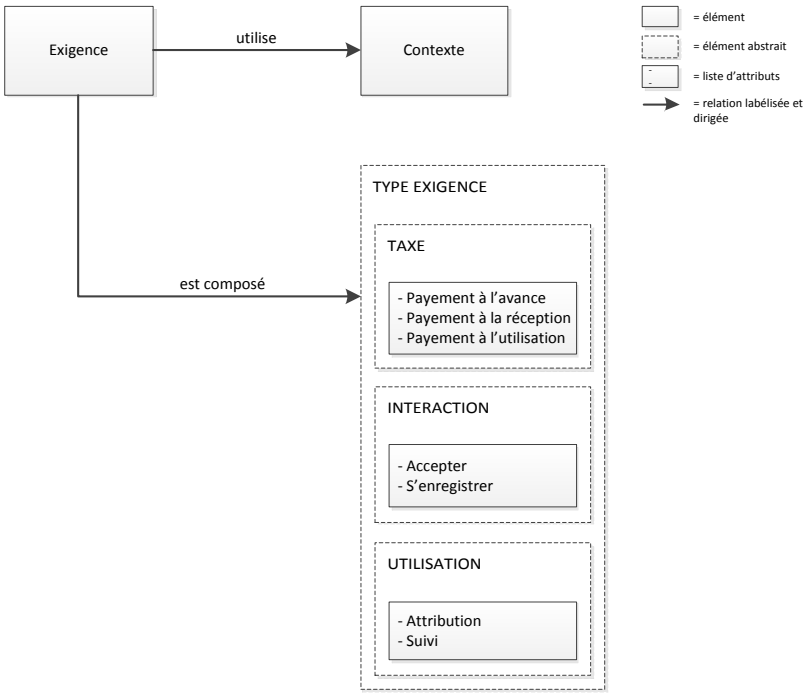

Figure 30 Concept exigence du modèle ODRL (IPR Systems Pty Ltd, 2002 p. 14)

L'entité « Exigence » est composée de trois entités abstraites utilisées pour regrouper des exigences similaires. Ces trois entités sont :

- L'entité « Taxe » qui indique un ensemble d'exigences relatives aux payements pour l'utilisation d'un objet (ex : payement à l'avance, payement à la réception, payement à l'utilisation) ;
- L'entité « Interaction » qui indique un ensemble d'exigences concernant les interactions utilisateur (ex : accepter, s'enregistrer) ;
- L'entité « Utilisation » qui indique un ensemble d'exigences sur l'utilisation d'un objet (ex : attribution, suivi).

Une exigence peut être associée à une ou plusieurs permissions. Si une exigence apparait au même niveau qu'un nombre de permissions, alors elle s'applique à chacune d'entre elles. Pour les permissions à exigences multiples, toutes les exigences doivent être satisfaites et aucun conflit ne doit survenir. Toute exigence exprimée mais pouvant être satisfaite entraine un refus de permission.

L'entité « Exigence » est également liée au « Contexte ».

3.3.3.4.4 Contrainte

Source : (IPR Systems Pty Ltd, 2002 pp. 10,11,12)

L'entité « Contrainte » concerne l'ensemble des restrictions sur les permissions d'un « Objet ». Le modèle ODRL des contraintes est repris à la Figure 31.

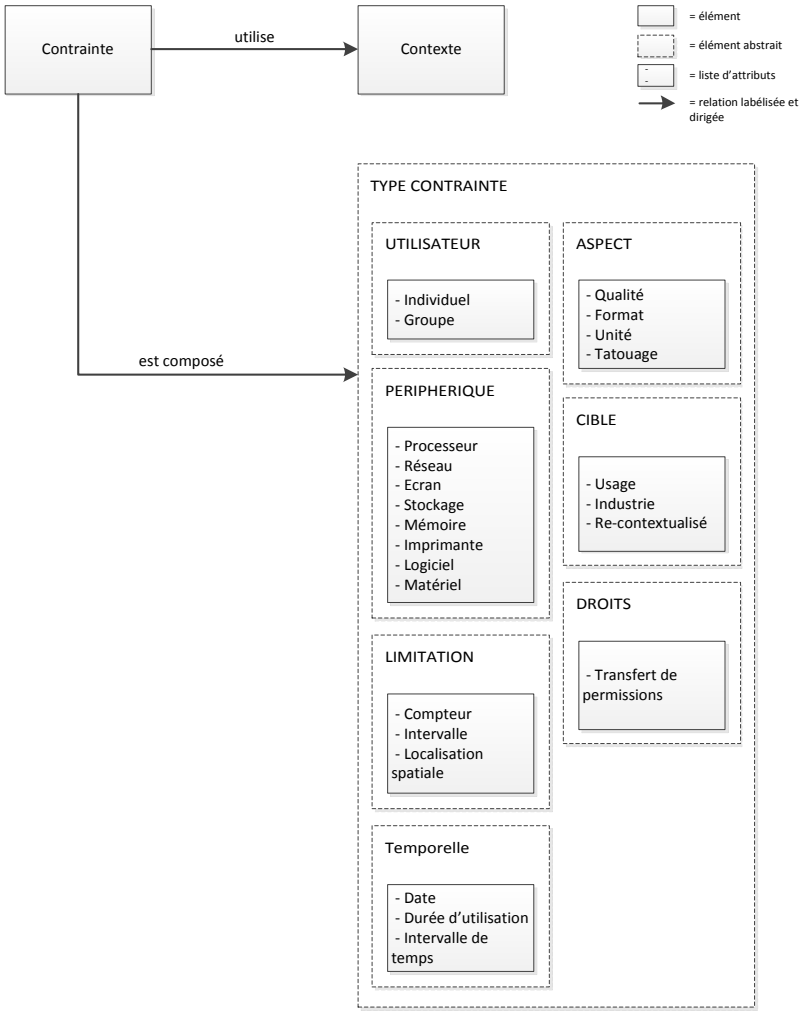

Figure 31 Concept contrainte du modèle ODRL (IPR Systems Pty Ltd, 2002 p. 10)

L'entité « Contrainte » est composée d'un ensemble de sept entités abstraites utilisées pour regrouper des contraintes similaires. Ces sept entités sont :

- L'entité « Utilisateur » qui indique un ensemble de contraintes limitant l'utilisation à un ou des utilisateur(s) identifié(s) (ex : individuel, groupe) ;
- L'entité « Périphérique » qui indique un ensemble de contraintes limitant l'utilisation à des systèmes ou périphériques physiques (ex : processeur, réseau, écran, stockage, mémoire, imprimante, logiciel et matériel) ;
- L'entité « Limitation » qui indique un ensemble de contraintes limitant l'utilisation à un nombre fixe ou à une étendue spatiale (ex : compteur, intervalle et localisation spatiale) ;
- L'entité « Temporelle » qui indique un ensemble de contraintes limitant l'utilisation dans le temps (ex : date, durée d'utilisation, intervalle de temps) ;
- L'entité « Aspect » qui est un ensemble de contraintes limitant l'utilisation à des caractéristiques distinctes ou aux expressions de l'objet (ex : qualité, format, unité, tatouage) ;
- L'entité « Cible » qui indique un ensemble de contraintes imposant une limite sur l'endroit et la manière dont est utilisé l'objet (ex : usage, industrie, re-contextualisé) ;
- L'entité « Droits » qui indique un ensemble de contraintes s'appliquant lors d'un transfert de permissions (ex : transfert de permissions). Cette contrainte « transfert de permissions » contient une liste de permissions qui seront transmises quand l'objet sera transféré. Si aucune permission n'est spécifiée dans cette contrainte, aucune ne sera transférée. Lors d'un transfert de permissions, il est possible de spécifier le type de transfert : « égal » (toutes les permissions spécifiées seront transmises), « moins » (un nombre inférieur des permissions spécifiées sera transmis) et « pas plus grand » (un nombre inférieur ou égal des permissions spécifiées sera transmis).

Une contrainte est associée à une permission. Si une contrainte apparaît au même niveau qu'un certain nombre de permissions, alors elle s'applique à toutes ces permissions. La plupart des contraintes peuvent être associées à zéro ou plusieurs autres contraintes ; dans ce cas, la ou les contrainte(s) imbriquée(s) s'appliquera(ont) à la contrainte principale. Par exemple la contrainte « compteur = 5 » imbriquée dans la contrainte « intervalle de temps = 7 jours » signifie qu'il est possible de consommer l'objet sur une période récurrente de 7 jours avec un maximum de 5 fois par période. Pour les permissions à contraintes multiples, toutes les contraintes doivent être satisfaites et aucun conflit ne doit survenir. Toute contrainte exprimée mais ne pouvant être satisfaite entraine un refus de permission.

L'entité « Contrainte » est associée au « Contexte » qui regroupe l'ensemble des informations additionnelles permettant de la décrire.

3.3.3.5 Les composants additionnels

3.3.3.5.1 Condition

Source : (IPR Systems Pty Ltd, 2002 p. 16)

L'entité « Condition » est constituée d'un ensemble de contraintes qui, lorsqu'elles deviennent vraies (ou se produisent), invalident la(les) permission(s) associée(s). Le modèle ODRL des conditions est repris à la Figure 32.

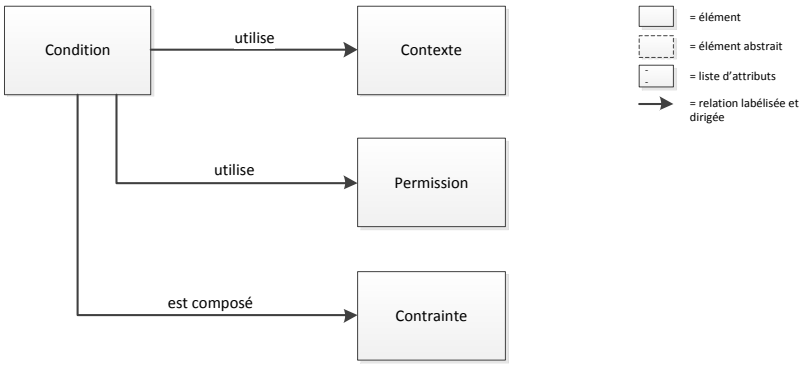

Figure 32 Concept condition du modèle ODRL (IPR Systems Pty Ltd, 2002 p. 16)

L'entité « Condition » utilise deux entités existantes :

- l'entité « Permission » qui regroupe l'ensemble des permissions déclenchant un événement ;
- l'entité « Contrainte » qui indique un ensemble de contraintes déclenchant un événement.

Lorsque la condition devient vraie, c'est-à-dire quand l'événement spécifique se produit, alors le système doit cesser d'accorder la permission à l'utilisateur. Il doit de plus l'en informer et lui fournissant les informations permettant de renégocier un nouvel accord.

Une condition peut être associée à une ou plusieurs permissions. Si une condition apparait au même niveau qu'un nombre de permissions, alors elle s'applique à chacune d'entre elles. Pour les permissions à conditions multiples, celles-ci doivent toutes être satisfaites et aucun conflit ne doit survenir.

Une condition se comporte de manière similaire à la combinaison d'une permission et d'une contrainte mais toutes deux ont des effets opposés. Un droit peut être exprimé à l'aide d'une permission et d'une contrainte limitant son utilisation. Cependant, une condition, exprimée avec la même sémantique, entraine l'annulation/suppression de la permission.

Cette entité est également associée au « Contexte ».

3.3.3.5.2 Contexte

Source : (IPR Systems Pty Ltd, 2002 pp. 19,20)

L'entité « Contexte » représente l'ensemble des informations additionnelles concernant les différentes entités du modèle et leurs relations. Le modèle ODRL du contexte est repris à la Figure 33.

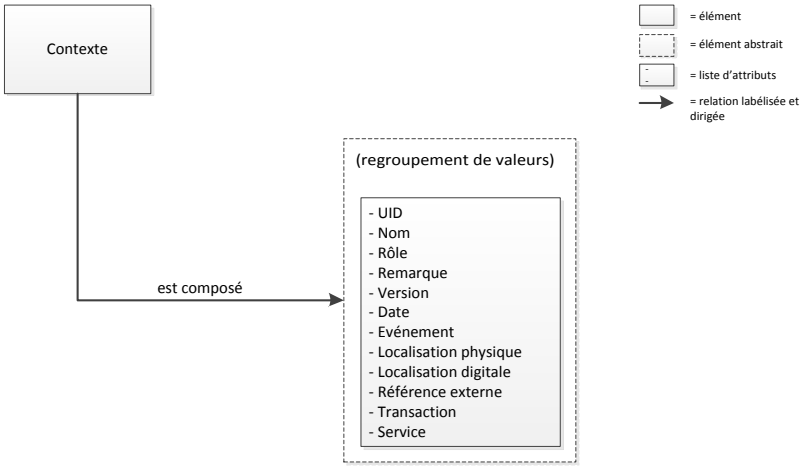

Figure 33 Concept contexte du modèle ODRL (IPR Systems Pty Ltd, 2002 p. 19)

L'entité « Contexte » est composée des douze informations suivantes :

- UID : un code d'identification unique pour l'entité ;
- Nom : un nom utilisé pour décrire l'entité ;
- Rôle : un rôle joué par l'entité ;
- Remarque : un commentaire général ou une description de l'entité ;
- Version : la version de l'entité ;
- Date : la date liée à l'entité (validité) ;
- Evénement : le type d'événement lié à l'entité ;
- Localisation physique : l'emplacement physique de l'entité ;
- Localisation digitale : l'emplacement digital de l'entité ;
- Référence externe : un lien vers une information additionnelle sur l'entité ;
- Transaction : une information au sujet de la transaction achetée liée à l'entité ;
- Service : un lien vers le service fournissant l'entité.

Un contexte peut être associé à n'importe quelle entité du modèle de données comme on peut le voir sur la Figure 34.

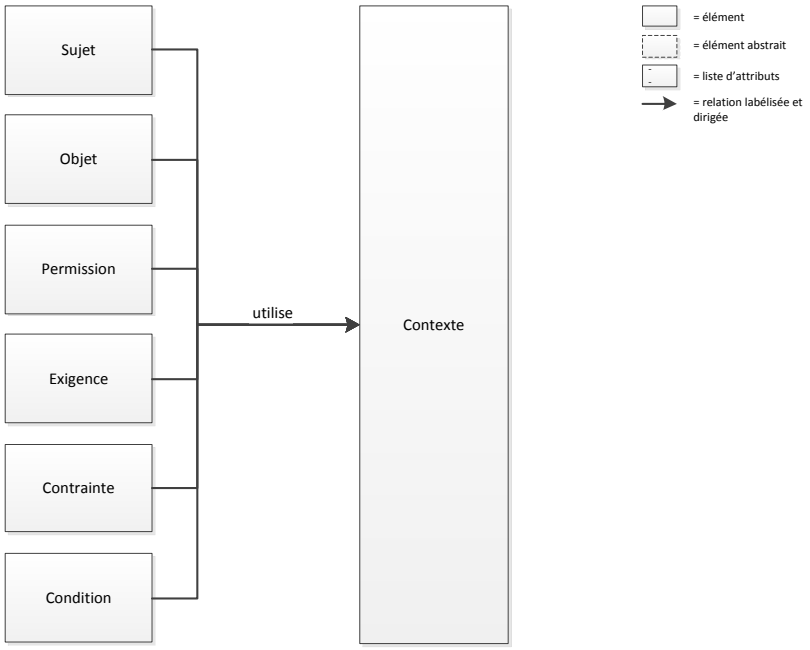

Figure 34 Concept contexte et les relations dans ODRL

Par exemple, lors de la déclaration d'un objet, le contexte peut être utilisé afin d'indiquer un identifiant unique ou bien pour ajouter une description. Dans le cas d'une déclaration d'un sujet, il est possible grâce au contexte d'indiquer un nom, un rôle associé à ce sujet. Pour la création d'une permission, il est également possible à l'aide du contexte de préciser la date de création de celle-ci.

3.3.3.5.3 Droit

Source : (IPR Systems Pty Ltd, 2002 p. 5)

L'entité « Droit » n'est pas réellement une entité à part entière. Elle est utilisée dans le modèle ODRL comme un mécanisme d'agrégation permettant de relier le composant « Contexte » à différents composants de protocole (tels que l' « Offre » et l' « Accord ») regroupant un ensemble de permissions sur divers objets. Sa représentation est illustrée à la Figure 35.

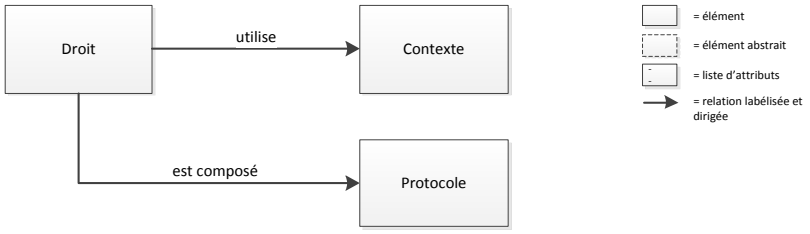

Figure 35 Concept droit du modèle ODRL

3.3.3.6 Conclusion

L'ODRL est un langage d'expression de droits basé sur le langage de balisage XML. L'utilisation de ce type de langage permet à l'ODRL d'être mis à jour en fonction des différentes exigences liées à l'environnement considéré simplement en y ajoutant de nouvelles entrées dans le dictionnaire de données.

La structure de celui-ci peut être calquée sur la structure de base des langages d'expression de droits à laquelle s'ajoutent deux nouvelles notions, à savoir la gestion des conditions et du contexte.

3.3.4 XrML

3.3.4.1 Introduction

Le XrML (eXtensible Rights Markup Language) est un langage de droit développé par la société ContentGuard (ContentGuard, 2000-2010). Celui-ci, tout comme l'ODRL, est basé sur le langage de balisage XML ce qui lui permet également d'être aisément mis à jour ou étendu par l'ajout nouvelles entrées dans le dictionnaire de données.

La version actuelle du XrML est la 2.0, annoncée le 26 novembre 2001. Une version 2.1 était également en cours d'élaboration mais fut abandonnée et non finalisée (ContentGuard, 2000-2010). C'est la raison pour laquelle notre analyse s'est uniquement basée sur la version 2.0.

3.3.4.2 L'architecture

Le langage XrML est organisé en plusieurs parties comme l'indique la Figure 36 (ContentGuard, 2002 a p. 9):

- Un « *schéma de base* » contenant les définitions abstraites des concepts de base du langage, à savoir le sujet, le droit, la ressource et la condition.
- Un « *schéma d'extension standard* » contenant les définitions permettant d'étendre l'utilisation du « schéma de base ». En particulier, il étend la notion de condition à l'aide de termes supplémentaires.
- Un « *schéma d'extension de contenu* » contenant les définitions relatives aux contenus numériques spécifiques tel que les livres, la musique et la vidéo. Cette extension permet d'étendre le « schéma de base » en y apportant la définition de droits, de ressources et conditions relatives à ces types de contenus numériques.
- Un « *autre schéma d'extension* » contenant les définitions relatives à un contenu spécifique déterminé par l'utilisateur.

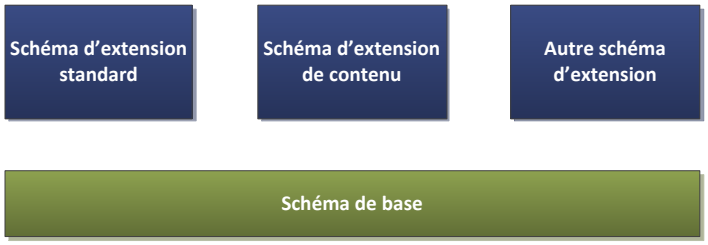

Figure 36 Architecture du modèle XrML

3.3.4.3 Le modèle simplifié

Tout comme dans le cas du modèle ODRL, le modèle XrML a été simplifié afin de ne prendre en compte que le modèle de données. Les entités définies dans XrML en tant que protocole permettant par exemple d'obtenir ou de révoquer des permissions font cependant partie de l'entité « Droit » (présente en tant que composant de base), ce qui n'était pas le cas dans le modèle ODRL. De ce fait, le modèle XrML simplifié ne diffère pas du modèle de donnée défini par ContentGuard. Celui-ci est présenté ci-dessous à la Figure 37 (sous forme de diagramme de classe).

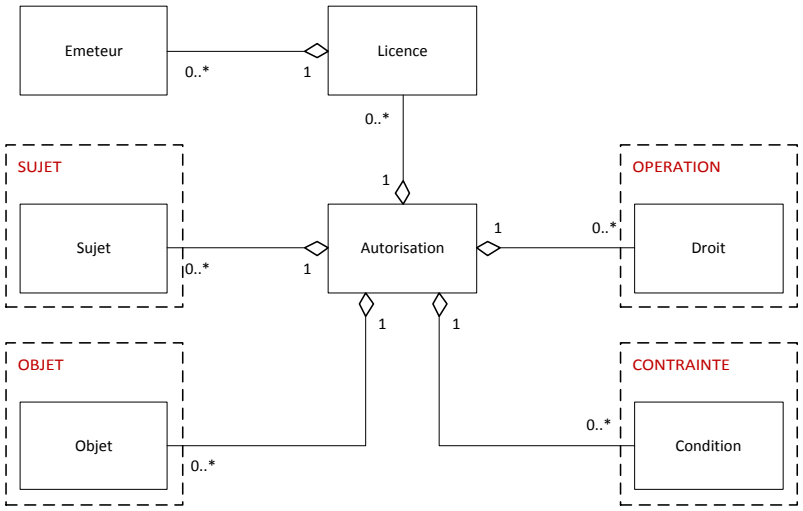

Figure 37 Modèle XrML simplifié (ContentGuard, 2002 a pp. 4,7)

Comme l'indique la figure, le modèle reprend les 4 composants de base des langages d'expression de droits définis au §3.3.2, soit le sujet, l'objet, l'opération et la contrainte. Tous ces composants sont décrits en détail ci-après.

3.3.4.4 Les 4 composants de base

3.3.4.4.1 Sujet

L'entité « Sujet » identifie les entités impliquées dans l'octroi ou l'exercice de droits.

Il s'agit en fait d'une entité abstraite. En effet, elle n'indique pas comment un sujet est identifié et authentifié. Pour ce faire, on utilise des « dérivations » de cette entité « Sujet ». Celles-ci peuvent être vues comme des mécanismes alternatifs de représentation du sujet. Ce genre de dérivation peut être défini dans les extensions du modèle XrML afin de permettre à un utilisateur de créer son propre mécanisme d'identification des sujets (ContentGuard, 2001b p. 28).

Dans le « schéma de base » de XrML, 2 types de dérivations importantes sont définies :

- « Détenteur de clé » (ou « KeyHolder ») : qui représente une personne identifiée par la possession d'une clé secrète ou d'une combinaison clé privée/publique (ContentGuard, 2001b p. 29).
- « Groupe de sujets » (ou « AllPrincipals ») : qui représente structurellement un ensemble de zéro ou plusieurs « Détenteur(s) de clé ». Sémantiquement, l'utilisation de cette dérivation ne signifie pas qu'un droit est octroyé à l'ensemble des « Détenteurs de clé » mais que tous ceux-ci agissent ensemble comme une entité « totale ». Par exemple, si un « Groupe de sujets » est identifié comme entité « Sujet » pour un droit particulier mais qui implique une signature pour l'octroi d'un droit bancaire, alors, conceptuellement, il est requis que chaque « Détenteur de clé » présent dans ce « Groupe de sujets » agisse en tant que co-signataire afin que l'entité « Sujet » puisse bénéficier du droit en question (ContentGuard, 2001b p. 28).

Néanmoins il est impossible de définir un « Groupe de sujets » comme une entité à part entière car la description de celle-ci passe obligatoirement par l'identification de l'ensemble des « Détenteurs de clé » qui la compose.

L'absence de « Détenteurs de clé » dans un droit ou l'utilisation d'un « Groupe de sujets » vide revient à définir un droit valable pour tout le monde (ContentGuard, 2001b p. 28).

Le modèle XrML des sujets est repris à la Figure 38.

Figure 38 Concept sujet du XrML (ContentGuard, 2001a p. 12)

3.3.4.4.2 Objet

L'entité « Objet » représente l'objet direct sur lequel le sujet effectue son action. Cela concerne un contenu digital (tel un libre électronique, un fichier audio, vidéo ou bien une image), un service (comme un service email, un service de transaction B2B) ou même une partie d'information appartenant à un sujet (comme un nom ou une adresse email) (ContentGuard, 2001a p. 14).

Cette entité est conceptuellement abstraite, tout comme l'entité « Sujet ». Pour caractériser un objet, on fait donc appel à des dérivations qui peuvent être définies dans les extensions du modèle XrML. Cela permet aux différents sujets de définir des objets propres à un domaine d'application particulier (ContentGuard, 2001b p. 32).

Dans le « schéma de base » de XrML (ContentGuard, 2001b p. 32), un type de dérivation appelé « Objet digital » a été défini de la manière suivante : « Il identifie un contenu digital en déterminant sa localisation, soit via l'objet lui-même ou à l'aide d'un fichier externe ou bien même un lien internet (ContentGuard, 2002 a p. 13) ». [Notre traduction]

Le modèle XrML des objets du schéma de base est représenté à la Figure 39.

Figure 39 Concept objet du schéma de base XrML (ContentGuard, 2001a p. 14)

Le schéma d'extension standard apporte 2 dérivations supplémentaires :

- « Nom » : à l'aide d'une « propriété de possession[4] », cette dérivation peut être utilisée dans les licences XrML pour associer un nom à un ou plusieurs sujets. Cela permet par exemple de pouvoir émettre des licences à des sujets identifiés non plus par leur clé secrète/publique mais par leur nom (ContentGuard, 2001c p. 38).

  Il est à noter que la dérivation « Nom » est abstraite et ne devrait pas apparaître dans une licence XrML excepté sous la forme d'une des variables de références suivantes :

  - *nom commun* : un nom par lequel une entité est familièrement connue (ContentGuard, 2001c p. 40).

[4] La définition de cette propriété est reprise dans le §3.3.4.4.3.

- *nom domaine* : un nom dans l'espace de nom de domaine (ContentGuard, 2001c p. 40).
- *nom email* : une adresse email associée à l'entité (ContentGuard, 2001c p. 39).
- *nom sujet x509* : le nom du sujet de certain certificat x509 associé à l'entité (ContentGuard, 2001c p. 41).

- « Révocable » : qui donne le droit à un sujet de révoquer un objet identifié par sa signature (ContentGuard, 2001c p. 42). Cette dernière est identifiée soit par sa valeur littérale ou par un moyen indirect (ex : fonction de hachage).

Le modèle XrML des objets dans l'extension standard est repris à la Figure 40.

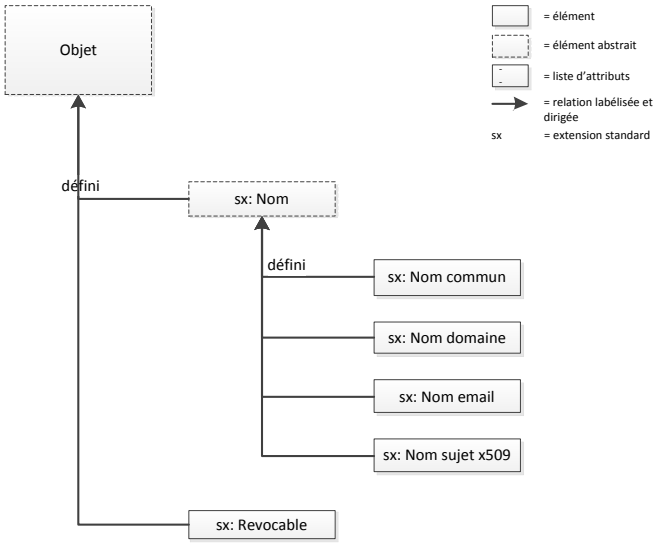

Figure 40 Concept objet de l'extension standard XrML (ContentGuard, 2001c pp. 38,42)

Le « schéma d'extension de contenu » apporte également 2 dérivations supplémentaires :

- « Travail digital » : qui représente le contenu digital auquel les droits et conditions sont appliqués (ContentGuard, 2001d p. 6). Ce « travail digital » est composé des informations suivantes (ContentGuard, 2001d p. 6) :
  - description : une description de l'objet pouvant être faite en plusieurs langues, ou simplement être plus complète ;
  - métadonnée : un ensemble d'informations additionnelles pouvant être utilisées dans de nombreux contextes (commerce électronique, …) :
    - titre : le titre du « travail digital » ;
    - créateur : le créateur du « travail digital » qui peut ne pas être le même que le(s) propriétaire(s) ;

- éditeur : l'éditeur du « travail digital » qui peut ne pas être le même que le(s) propriétaire(s) ;
- date publication : la date et optionnellement l'heure de la publication ;
- propriétaire : le propriétaire du « travail digital »qui peut ne pas être le même que le créateur ou éditeur. Les propriétaires maintiennent les droits d'auteurs ;
- droit d'auteur : la déclaration formelle des droits d'auteur ;
- autre : toute autre information supplémentaire.

- localisateur : un moyen de localiser le contenu du travail ;
- parties : une identification de chaque partie de ce travail où chacune d'entre elle est également un « travail digital ».

- « Niveau de sécurité » : qui est une indication abstraite du niveau de sécurité dont un sujet dispose et est définie à l'aide de la « Propriété de possession » (ContentGuard, 2001d p. 29).

Le modèle XrML des objets dans le schéma d'extension de contenu est repris à la Figure 41.

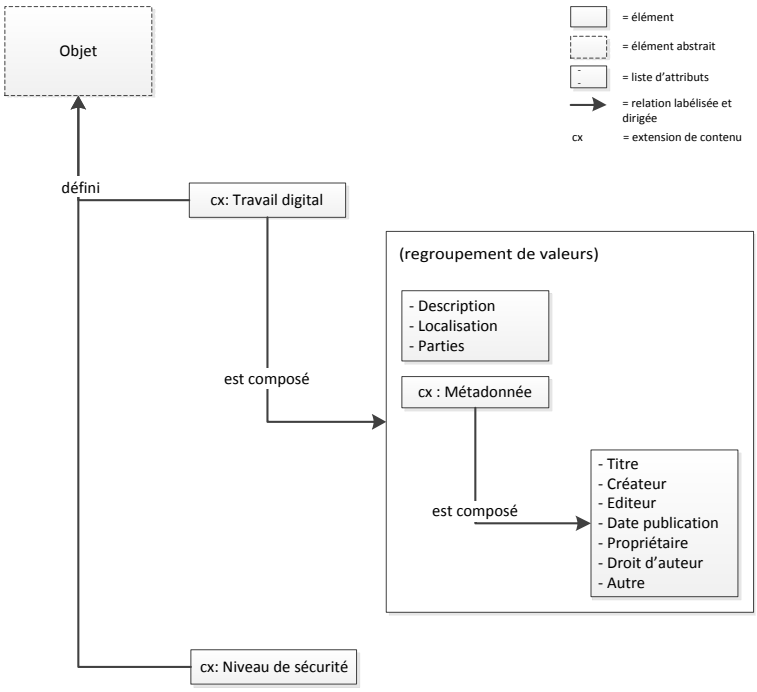

Figure 41 Concept objet de l'extension de contenu XrML (ContentGuard, 2001d pp. 5-7)

La représentation générale de l' « Objet » combinant le schéma de base, l'extension standard et l'extension de contenu est reprise à la Figure 42.

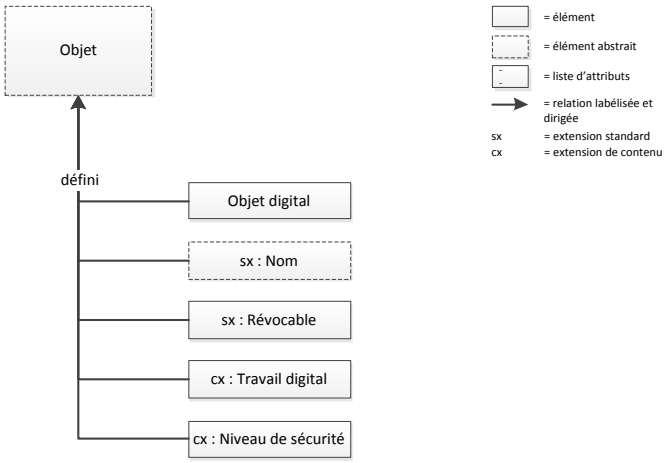

Figure 42 Concept objet du modèle XrML (ContentGuard, 2001a p. 14)

3.3.4.4.3 Opération

Le modèle défini par XrML permet l'expression de droits grâce à l'entité « Droit » et l'expression d'obligations grâce à l'entité « Condition » (pour les conditions de type obligation).

Il s'agit également d'une entité est abstraite exprimée à l'aide de dérivations qui peuvent être définies dans les extensions du modèle XrML dans le cadre de domaines d'application spécifique. (ContentGuard, 2001b p. 29).

Dans le schéma de base XrML, 4 types de dérivations sont préalablement définis :

- « Émettre » : qui donne le droit à un sujet d'émettre des licences (ContentGuard, 2001b p. 30).
- « Révoquer » : qui donne le droit à un sujet de révoquer des licences. Les sujets à l'origine de l'émission d'une licence possèdent le droit implicite de révoquer leur propre signature (ContentGuard, 2001b p. 30).
- « Propriété de possession » : qui donne le droit à un sujet de revendiquer la propriété de certaines caractéristiques telles qu'une adresse email (ContentGuard, 2001b p. 31).
- « Obtenir » : qui donne le droit à un sujet d'obtenir d'autres droits supplémentaires (ContentGuard, 2001b p. 31).

Le modèle XrML du schéma de base des droits est repris à la Figure 43.

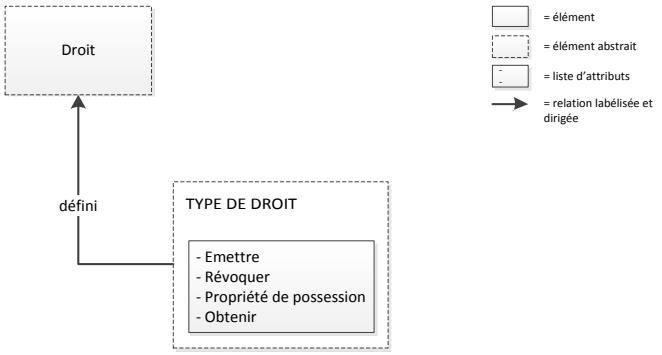

Figure 43 Concept droit du schéma de base XrML (ContentGuard, 2001a p. 13)

Aucune dérivation supplémentaire n'est définie dans le schéma d'extension standard.

Le schéma d'extension de contenu apporte 20 dérivations supplémentaires qui ont été définies dans le but d'aider la distribution et l'utilisation de « travaux digitaux ». Conceptuellement, on peut les regrouper en 5 catégories :

- « Droits de rendu » : qui régissent le rendu d'un « travail digital » (ex : jouer, imprimer, exporter) (ContentGuard, 2001d p. 7).
- « Droits de transfert » : qui régissent le mouvement d'un « travail digital » d'un dépôt[5] à un autre (ex : copier, transférer, prêter) (ContentGuard, 2001d pp. 7-8).
- Droits de réutilisation : qui régissent la réutilisation d'un « travail digital », dans son entièreté ou en partie afin de créer un travail nouveau ou composite (ex : éditer, extraire, intégrer) (ContentGuard, 2001d p. 8).
- Droits de gestion de fichier : qui régissent les accès entre différents dépôts lors de transfert de travail par exemple, mais également les sauvegardes et restaurations d'un « travail digital » (ex : lire, écrire, exécuter, effacer, sauvegarder, restaurer, vérifier, gérer dossiers et obtenir informations accès données) (ContentGuard, 2001d p. 8).
- Droits de configuration : qui régissent l'ajout et la suppression d'un logiciel système dans un dépôt (ex : installer, désinstaller) (ContentGuard, 2001d p. 8).

Le modèle XrML des droits dans le schéma d'extension de contenu est repris à la Figure 44.

[5] « En informatique, un **dépôt** ou **référentiel** (de l'anglais *repository*), est un stockage centralisé et organisé de données. Ce peut être une ou plusieurs bases de données où les fichiers sont localisés en vue de leur distribution sur le réseau, ou bien un endroit directement accessible aux utilisateurs. » (Wikipedia, 2011).

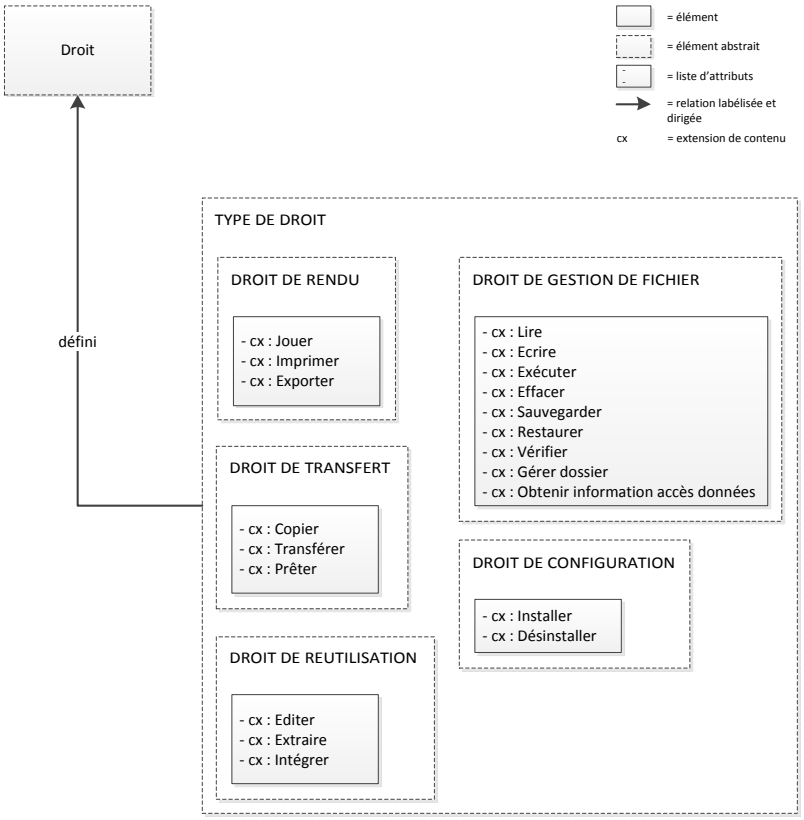

Figure 44 Concept droit de l'extension de contenu XrML (ContentGuard, 2001d p. 9)

Le modèle XrML des conditions, considérées comme obligation dans la structure des RELs, n'apporte aucun type de dérivation dans le « schéma de base ». Il en est de même pour le « schéma d'extension de contenu ». Seul le « schéma d'extension standard », dont le modèle est présenté à la Figure 45, propose quatre types de dérivation :

- « Taxe » : qui indique qu'une taxe doit être payée avant qu'un droit ne puisse être exercé (ContentGuard, 2001c p. 18). Elle regroupe un ensemble d'informations :
    - Résumé de payement : qui est un élément abstrait caractérisé par un ensemble d'informations (ex : meilleur prix, demande de prix, …) relatives au payement (ContentGuard, 2001c p. 19).
    - Minimum : qui indique le montant minimum à payer (ContentGuard, 2001c p. 19).
    - Maximum : qui indique le montant maximum à payer (ContentGuard, 2001c p. 19).
    - Vers : qui indique à qui le payement sera envoyé (ContentGuard, 2001c p. 19).
- « Rechercher approbation » : qui indique que l'exercice d'un droit nécessite l'approbation d'un service (ContentGuard, 2001c p. 8).
- « Rapport suivi » : qui indique que l'exercice d'un droit doit être rapporté à un service de suivi désigné (ContentGuard, 2001c p. 9).

- « Requête suivi » : qui indique une condition sur l'état du suivi (ContentGuard, 2001c p. 10). Cette condition peut par exemple être utilisée pour prédire l'autorisation d'un droit à la suite d'un exercice réussi d'un autre.

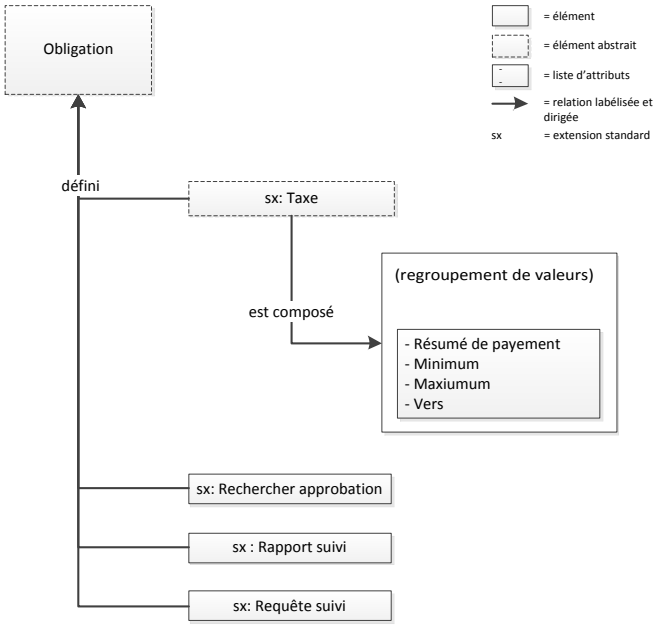

Figure 45 Concept obligation de l'extension standard XrML (ContentGuard, 2001c p. 4)

3.3.4.4.4 Contrainte

L'entité « Condition » représente les contraintes qu'un sujet doit satisfaire avant de bénéficier d'une autorisation.

Il s'agit également d'une entité abstraite définie à l'aide de dérivations.

Dans le schéma de base, 5 types de dérivations sont préalablement définis :

- « Toutes les conditions » : qui est en fait un ensemble de conditions (ContentGuard, 2001b p. 34). Afin que la dérivation soit satisfaite, toutes les conditions la composant doivent l'être également.
- « Droit existant » : qui vérifie si certains droits ont bien été accordés par un émetteur de confiance (ContentGuard, 2001b p. 37).
- « Droit prérequis » : qui vérifie si un sujet donné dispose d'un droit donné sur un objet donné soumis soit à une condition satisfaite ou à aucune condition (ContentGuard, 2001b p. 38).
- « Intervalle de mise à jour de validité » : qui indique l'intervalle de temps dans lequel une vérification de la validité de la signature sera effectuée (afin de vérifier si celle-ci n'a pas été révoquée) (ContentGuard, 2001b pp. 35-36).

- « Intervalle de validité » : qui indique une période de temps durant laquelle le droit est valable (ContentGuard, 2001b p. 35).

Le modèle XrML des conditions dans le schéma de base est repris à la Figure 46.

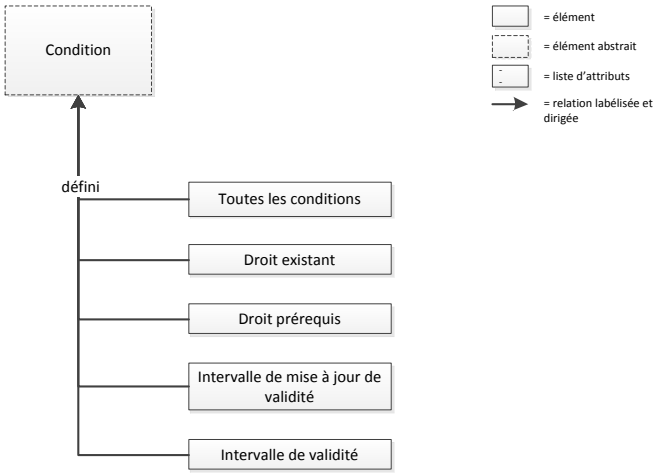

Figure 46 Concept obligation du schéma de base XrML (ContentGuard, 2001a p. 15)

Le schéma d'extension standard apporte 9 dérivations supplémentaires dont 4 ont été présentées dans l'entité « Droit » en tant que conditions d'obligations, c'est-à-dire des exigences.

Ces 4 dérivations sont :

- « Limite exercice » : qui indique le nombre de fois que le droit peut être exercé (ContentGuard, 2001c p. 7).
- « Territoire » : qui limite l'exercice du droit à certaines localisations (ContentGuard, 2001c p. 21). Ces localisations peuvent correspondre à des régions physiques ou géographiques, ou à des localisations virtuelles, digitales.
- « Intervalle flottant » : qui indique l'intervalle de temps qui commence avec le premier exercice du droit (ex : une autorisation qui propose un droit qui peut être exercé pendant 1 semaine après sa première utilisation) (ContentGuard, 2001c p. 13).
- « Mesure validité » : qui indique une période d'accumulation de temps (non contigüe) (ContentGuard, 2001c p. 14). Une utilisation peut démarrer et stopper l'exercice de son droit, et le temps mesuré s'additionne au fur et à mesure. Tant que le temps mesuré est inférieur au temps de validité, le droit peut être exercé.
- « Périodique » : qui indique que le droit peut être exercé de manière périodique (ex : chaque semaine pendant une heure) (ContentGuard, 2001c p. 16).

Le modèle XrML des « conditions » dans le schéma d'extension standard est repris à la Figure 47.

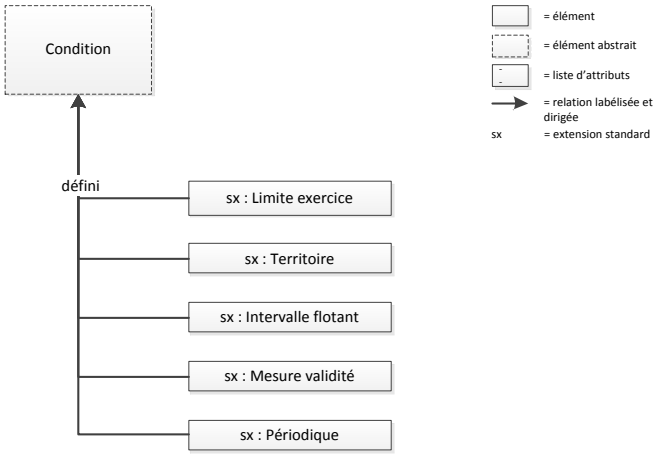

Figure 47 Concept condition de l'extension standard XrML (ContentGuard, 2001c p. 4)

Le « schéma d'extension de contenu » apporte 5 dérivations supplémentaires (ContentGuard, 2001d p. 10) :

- « Destination » : qui indique le dépôt vers lequel un travail peut être déplacé (lors du droit « transfert » par exemple).
- « Aide » : qui indique le programme qui peut être utilisé pour exercer un droit (ex : un lecteur vidéo spécifique).
- « Rendu » : qui identifie le périphérique qui peut être utilisé pour rendre un travail.
- « Source » : qui indique le dépôt source ou périphérique à utiliser lors de l'exercice du droit.
- « Tatouage » : qui spécifie une liste d'information à intégrer lors d'une copie d'un travail par un périphérique lors de la production de la copie (ex : pour le copyright).

Le modèle XrML des conditions dans le schéma d'extension de contenu est repris à la Figure 48.

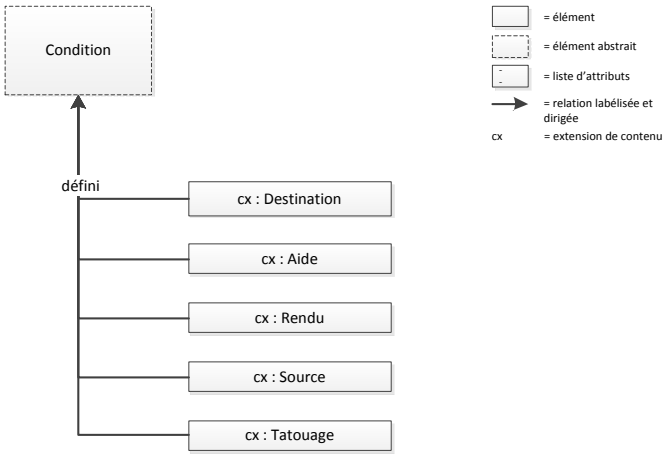

Figure 48 Concept condition de l'extension de contenu XrML (ContentGuard, 2001d p. 10)

3.3.4.5 Les composants additionnels

3.3.4.5.1 Autorisation

Une autorisation est un composant d'une licence (définie ci-après) qui permet à certains sujets de réaliser une opération. En d'autres termes, elle autorise un sujet particulier à exercer un certain droit sur un objet identifié éventuellement soumis à certaines conditions (ContentGuard, 2001a p. 11).

Cette entité comprend les éléments suivant :

- Le sujet
- L'objet
- Le droit
- La condition

Et est représentée de la manière suivante :

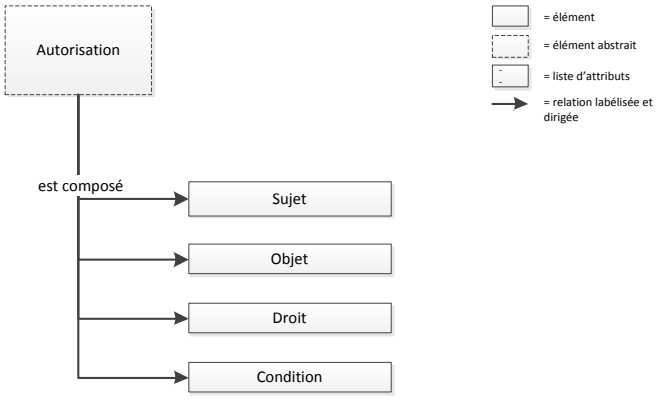

Figure 49 Concept autorisation du XrML (ContentGuard, 2001a p. 11)

Un groupe d'autorisations est simplement un conteneur d'autorisations. En utilisant cet élément, il est possible de regrouper plusieurs autorisations pour, par exemple, y appliquer des conditions communes à toutes.

3.3.4.5.2 Emetteur

Il s'agit de l'entité émettrice de la licence qui accorde les autorisations aux différents sujets (ContentGuard, 2001a p. 10).

3.3.4.5.3 Licence

Une licence peut être vue comme un conteneur d'autorisations composé des éléments suivants (ContentGuard, 2001a p. 10) :

- Un ensemble d'autorisations ;
- Un ensemble d'émetteurs ;
- Des informations additionnelles comme le nom de la licence, …

Elle est représentée de la manière suivante :

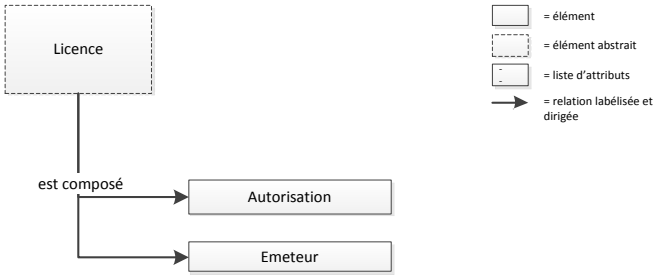

Figure 50 Concept licence du XrML (ContentGuard, 2001a p. 10)

Un groupe de licence est simplement un conteneur de licences. En utilisant cet élément, il est possible de définir plusieurs licences dans le même fichier de licence XrML.

3.3.4.6 Conclusion

Le XrML est un langage de haute expressivité du fait de l'utilisation du langage de balisage XML. Il permet dès lors de répondre à des besoins divers, que ce soit pour des applications commerciales (à l'aide principalement de l'extension de contenu) mais également pour d'autres domaines d'application comme par exemple le contrôle d'accès à des données confidentielles (avec l'utilisation du niveau de sécurité présent dans l'extension de contenu, de la source, destination du dépôt du travail, …). Il est, de plus, facilement modifiable par l'ajout d'extension personnelle.

Le réel inconvénient de ce modèle consiste en sa compréhension. En effet, en plus de la définition d'un grand nombre d'entités abstraites ne facilitant pas la représentation sémantique du modèle, l'ordre dans lequel sont présentées les différentes informations (sujet, objet, …) n'est pas prédéfini et l'interprétation d'une licence XrML peut porter à confusion. Il serait nécessaire d'apporter une sémantique formelle à ce langage.

On peut ajouter que depuis l'introduction du XrML en 2001, le développement s'est arrêté. Tous les travaux portant sur le XrML ont abouti à l'élaboration d'un nouveau langage, le MPEG-21 REL.

3.3.5 MPEG21 REL

3.3.5.1 Introduction

Le MPEG21-REL (Moving Picture Experts Group 21 Rights Expression Language) est un langage de droits développé par le groupe de travail MPEG (Moving Picture Experts Group, sd.). Celui-ci, défini par la norme ISO/IEC 21000-5 est basé sur la version 2.1 du XrML (ContentGuard, 2002 b).

3.3.5.2 Architecture

Dans cette version, non finalisée en raison du développement en parallèle au sein de ContentGuard et MPEG, le XrML est composé de deux parties : le schéma de base et le schéma d'extension standard. L'extension relative à la gestion de contenus digitaux est développée par le groupe MPEG (ContentGuard, 2002 b p. 5).

L'architecture du MPEG-21 REL est la suivante :

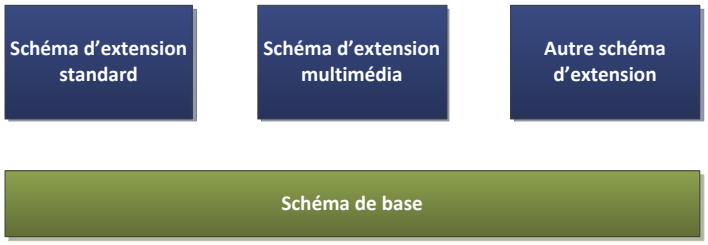

Figure 51 Architecture du modèle MPEG-21 REL (ContentGuard, 2002 b p. 9)

Parallèlement au MPEG-21 REL, un dictionnaire de données (MPEG-RDD – Rights Data Dictionary) a été développé. Ce développement a abouti à la création d'une norme ISO-IEC 21000-6, qui comprend dès lors un ensemble clair, consistant, structuré de termes pouvant être utilisés dans le MPEG-21 REL (Multimedia Description Schemas (MDS) Group, 2005).

3.3.5.3 Composants

En raison d'une architecture de base commune entre le MPEG-21 REL et le XrML 2.0, l'analyse portant sur le modèle de données ainsi que les composants en faisant partie ne sera pas détaillée ici. Cependant, comme expliqué ci-dessus, on peut remarquer le remplacement du « schéma d'extension de contenu » présent dans le XrML 2.0 par le « schéma d'extension multimédia ». Ce changement entraine des modifications car bien que le « schéma d'extension multimédia » soit analogue à celui du contenu, certain droits ont disparus et d'autres ont été ajoutés ou remplacés.

Des 20 droits issus de l'extension de contenu, il n'en reste plus que 14 dans l'extension multimédia (Multimedia Description Schemas (MDS) Group, 2005). Ceux-ci sont définis de la manière suivante (Wang, et al., 2004 p. 9) :

- « Adapter » : qui permet la création d'un nouvel objet basé sur un objet existant sur lequel on aurait ajouté/supprimé du contenu (équivaut à un « edit » du XrML).
- « Effacer » : qui permet de supprimer un objet (équivaut au « effacer » du XrML).
- « Diminuer » : qui permet la création d'un nouvel objet basé sur un objet existant sur lequel on aurait supprimé du contenu.
- « Intégrer » : qui permet d'intégrer un objet dans un autre (équivaut au « intégrer » du XrML).
- « Augmenter » : qui permet la création d'un nouvel objet basé sur un objet existant sur lequel on aurait ajouté du contenu (équivaut au « edit » du XrML).
- « Agrandir » : qui permet d'agrandir un objet en y ajoutant du contenu.
- « Exécuter » : qui permet l'exécution de contenu (équivaut au « exécuter » du XrML).
- « Installer » : qui permet l'installation d'un nouvel objet (équivaut au « installer » du XrML).
- « Modifier » : qui permet de modifier un objet soit en ajoutant ou supprimant du contenu.
- « Déplacer » : qui permet de déplacer un objet (équivaut au « transférer » du XrML).
- « Jouer » : qui permet d'utiliser le contenu (équivaut au « lire » de XrML).
- « Imprimer » : qui permet d'imprimer le contenu (équivaut au « imprimer » de XrML).
- « Réduire » : qui permet de réduire un objet en y enlevant du contenu.
- « Désinstaller » : qui permet la désinstallation d'un objet (équivaut au « désinstaller » du XrML).

L'ensemble des droits issus de l'extension de contenu du XrML et ceux définis dans le MPEG-21 REL sont repris et comparés dans le Tableau 3. La comparaison s'est basée sur les définitions de l'entité « Droit » du schéma d'extension de contenu (ContentGuard, 2001d pp. 7-8) ainsi que celle de l'extension Multimédia du MPEG-21 (Wang, et al., 2004 p. 9).

Tableau 3 Comparaison XrML / MPEG-21

XrML	**MPEG-21**
Jouer	Jouer
Imprimer	Imprimer
Exporter	/
Copier	/
Transférer	Déplacer

XrML	MPEG-21
Prêter	/
Editer	Adapter
Extraire	Diminuer
Intégrer	Intégrer
Lire	/
Ecrire	/
Exécuter	Exécuter
Effacer	Effacer
Sauvegarder	/
Restaurer	/
Vérifier	/
Gérer dossiers	/
Obtenir informations accès données	/
Installer	Installer
Désinstaller	Désinstaller
/	Augmenter
/	Modifier
/	Agrandir
/	Réduire

Le Tableau 3 montre que :

- Certaines définitions de droits sont équivalentes dans les deux langages (comme par exemple les termes Installer, Exécuter,…) ;
- Certaines étaient présentes dans XrML mais ont disparu dans le MPEG-21. C'est le cas des entités suivantes : Exporter, Copier, Prêter, Lire, Ecrire, Sauvegarder, Restaurer, Vérifier, Gérer dossiers, Obtenir information accès données.
- D'autres sont par contre présentes dans MPEG-21 mais n'existaient pas dans le XrML. C'est le cas pour : Augmenter, Modifier, Agrandir, Réduire.

Il est toutefois à noter que la définition de l'entité « Gérer dossier » dans XrML pourrait être incluse dans l'entité « Déplacer » de MPEG-21. De même, l'entité « Augmenter » défini dans MPEG-21 pourrait être inclus dans la définition de l'entité « Editer » dans XrML. L'analyse détaillée du dictionnaire de données permettrait probablement d'éclaircir ce point[6].

[6] Celui-ci, disponible moyennant payement à l'adresse suivante http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=36096, n'a toutefois pas pu être analysé dans le cadre du présent mémoire.

L'extension de contenu dans XrML permettait également la définition d'un « travail digital » caractérisé par un ensemble d'informations telles qu'une description, un créateur, un droit d'auteur, … Dans le MPEG-21 REL, cette possibilité n'est pas présente dans l' « extension multimédia » mais dans une autre norme, définie par l'ISO/IEC 21000-2 et 3 qui permet :

- de décrire un ensemble de termes abstraits ainsi que des concepts utilisés pour la création d'un modèle permettant de définir un objet digital (Multimedia Description Schemas (MDS) Group, 2002 p. 5) ;
- et d'identifier de manière unique un objet digital (Multimedia Description Schemas (MDS) Group, 2002 p. 8).

3.4 XACML

3.4.1 Introduction

Le XACML (eXtensible Access Control Markup Language) est un langage dédié au contrôle d'accès et basé sur le XML (OASIS, 2011). Il combine à la fois un langage de politiques de contrôle d'accès basé sur les attributs ; c'est-à-dire des caractéristiques permettant de décrire plus en détail les entités du modèle (comme pour le modèle de contrôle d'accès basé sur les attributs ABAC) mais également un protocole de type requête/réponse. En plus de l'apport d'une syntaxe, il propose une architecture permettant la mise en place de cet échange d'information.

La version actuelle du XACML est la 2.0, standardisée par OASIS le 1er avril 2005 (OASIS, 2011). Une version 3.0 à l'état de brouillon est actuellement en cours d'élaboration mais étant donné que celle-ci n'est pas finalisée et non standardisée, notre analyse s'est basée sur la version 2.0 (OASIS, 2011).

3.4.2 Le modèle simplifié

Le modèle XACML (OASIS, 2005 p. 19) a été simplifié afin de faire correspondre son analyse aux langages d'expressions de droits étudiés dans les paragraphes précédents. Dès lors, le protocole de requête/réponse ne sera pas détaillé dans le cadre de ce mémoire.

Le modèle XACML simplifié est représenté ci-dessous à la Figure 52 (sous forme de diagramme de classe).

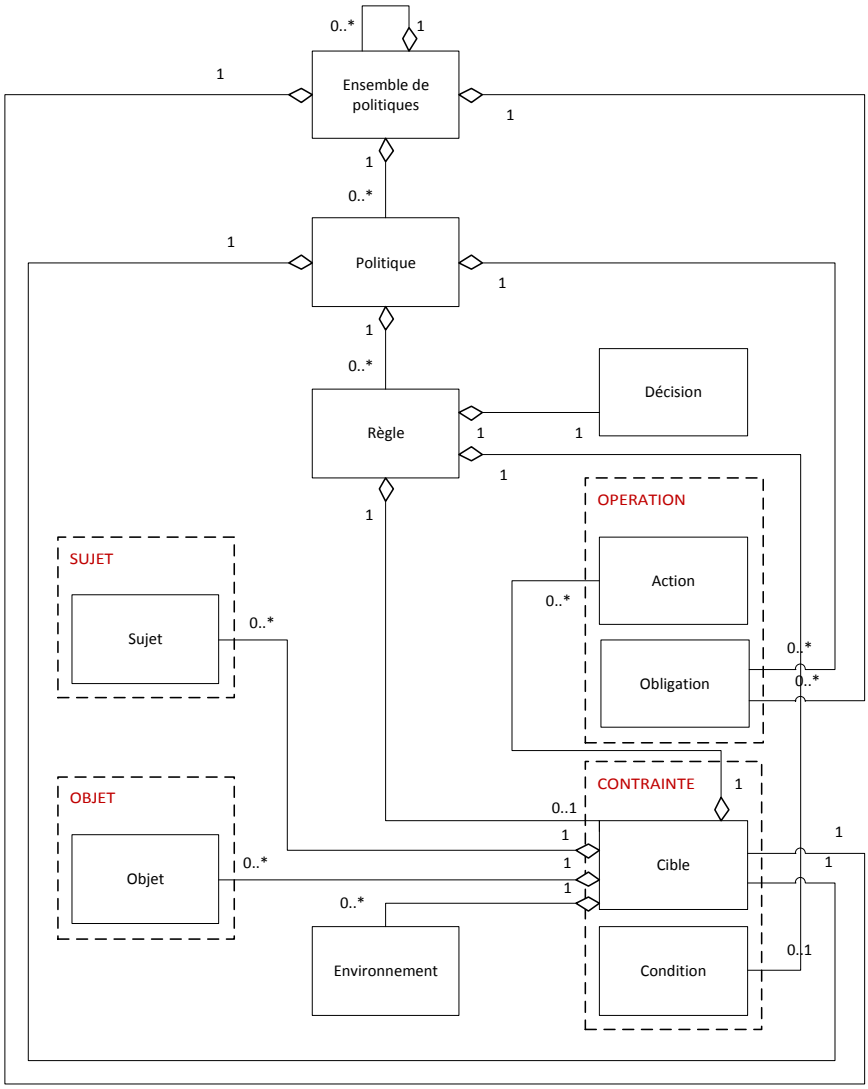

Figure 52 Modèle XACML simplifié (OASIS, 2005 p. 9)

Comme l'indique ce diagramme, le modèle reprend les 4 composants de base des langages d'expression de droits défini au §3.3.2 (bien qu'il s'agisse d'un langage de politiques), soit le sujet, l'objet, l'opération et la contrainte. Tous ces composants sont décrits en détail ci-après.

3.4.3 Les 4 composants de base

3.4.3.1 Sujet

Il est caractérisé par un ensemble d'attributs comme le nom, le domaine de sécurité ou bien même la clé publique nécessaire à prouver son identité, il correspond à l'origine de la demande d'accès sur un objet (OASIS, 2005 p. 128).

3.4.3.2 Objet

Il est caractérisé également par un ensemble d'attributs comme le nom, le type, il correspond à l'information dont le sujet souhaite accéder (OASIS, 2005 p. 129).

3.4.3.3 Opération

Le modèle XACML permet l'expression d'opérations en combinant les deux entités suivantes :

- L'entité « Action », caractérisée par un ensemble d'attributs (ex : le nom) identifiant l'action pour laquelle l'accès est demandé (OASIS, 2005 p. 129) ;
- L'entité « Obligation » qui correspond à une action supplémentaire à exécuter en conjonction avec la décision générée (OASIS, 2005 p. 87).

3.4.3.4 Contrainte

Cette entité correspond à une restriction (sous la forme d'une expression booléenne) permettant d'affiner une règle de politique (OASIS, 2005 p. 21). Elle peut être réalisée grâce à l'entité « Cible » et « Condition ».

Une cible est un ensemble de conditions simples, c'est-à-dire des fonctions booléennes rapides à évaluer, sur le sujet, l'objet, l'action et l'environnement et est présent dans une règle, une politique ou un ensemble de politiques (OASIS, 2005 p. 9).

Le modèle XACML d'une cible est repris à la Figure 53

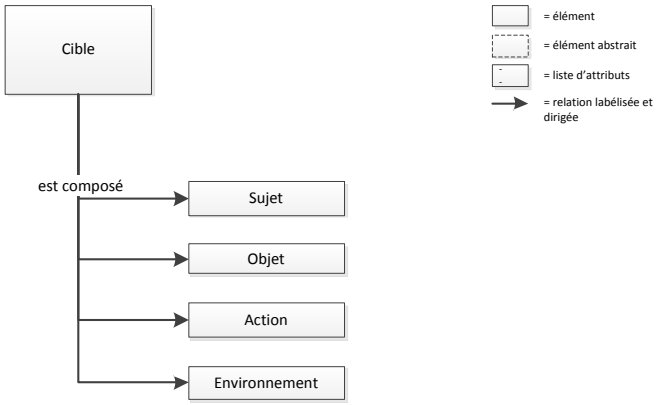

Figure 53 Concept cible du modèle XACML (OASIS, 2005 p. 19)

Une condition est également une fonction booléenne mais dont l'évaluation peut être plus compliquée (nécessitant une analyse de plusieurs caractéristiques par exemple). Elle porte également sur les entités « Sujet », « Objet », « Action » et « Environnement » et tout particulièrement sur les valeurs de leurs attributs. Afin de pouvoir effectuer des comparaisons, différentes fonctions d'évaluations ont été prédéfinies et elles peuvent être de plusieurs types (arithmétique, logique, temporelle, …) (OASIS, 2005 p. 107).

3.4.4 Les composants additionnels

3.4.4.1 Environnement

Cette entité est caractérisée par un ensemble d'attributs comme l'heure, la date,… Elle apporte des informations supplémentaires n'étant pas associées aux différents composants (OASIS, 2005 p. 130) et permet d'ajouter différents paramètres utiles lors de la prise de décision.

3.4.4.2 Règle

L'entité « Règle » combine l'entité « Cible » à un ensemble de conditions auxquelles on ajoute une décision (OASIS, 2005 p. 20).

Le modèle XACML d'une règle est repris à la Figure 54.

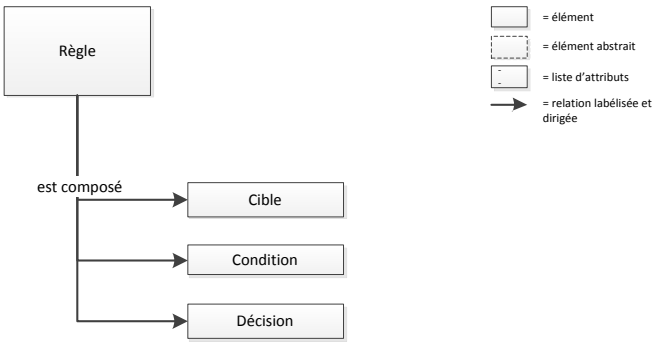

Figure 54 Concept règle du modèle XACML (OASIS, 2005 p. 19)

Une règle est valide si la cible et les conditions sont satisfaites. L'action demandée sera réalisée en fonction de la valeur de l'entité « Décision »

3.4.4.3 Décision

L'entité « Décision » indique si l'action est autorisée ou refusée en fonction de la validité d'une règle. Par exemple, si la règle est validée (la cible et les conditions sont satisfaites) mais que la décision est de type « refusée », cela signifie que dans ces conditions, cette cible n'a pas le droit d'exécuter cette action sur cet objet (OASIS, 2005 p. 21).

Dans le cas où la cible ne correspond pas à la requête ou bien si les conditions ne sont pas satisfaites, la règle est dite « non applicable ». En cas d'erreur, ou d'informations manquantes pour l'évaluation de conditions, la règle est dite « indéterminée ».

3.4.4.4 Politique

L'entité « Politique » combine plusieurs règles auxquelles on ajoute une cible et un ensemble d'obligations (OASIS, 2005 p. 21).

Le modèle XACML d'une politique est repris à la Figure 55.

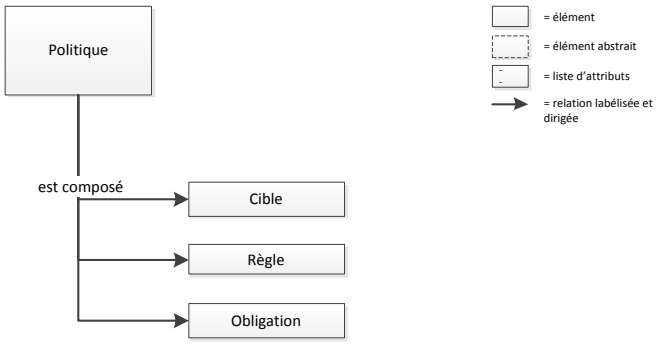

Figure 55 Concept politique du modèle XACML (OASIS, 2005 p. 19)

L'objectif d'une politique est de regrouper des règles en des ensembles, voire des sous-ensembles afin d'optimiser leur exploitation. L'utilisation d'une cible unique au sein d'une politique permet de limiter l'ensemble des règles qui la composent. Cela évite de devoir analyser toutes les cibles de chaque politique si chacune ne devait contenir qu'une seule règle. Le fait de regrouper des règles au sein d'une politique commune permet donc d'accélérer la recherche de règles s'appliquant à la requête et de structure l'ensemble des règles de façon plus « logique ».

Etant donné la possibilité de présence de plusieurs règles au sein d'une même politique ; et que chacune d'entre elles est associée à une décision, plusieurs règles peuvent dès lors s'appliquer à une même requête et par conséquent, plusieurs décisions peuvent être prises.

Pour faire face à ce genre de situation, XACML propose d'utiliser un algorithme de combinaison de règles spécifiant la procédure par laquelle les décisions issues de l'évaluation des différentes règles seront combinés lors de l'évaluation de la politique (OASIS, 2005 p. 22).

Trois comportements sont définis (OASIS, 2005 p. 134) :

- « Préséance du refus » : dans l'ensemble de toutes les règles d'une politique, si au moins une règle est évaluée à « refuser », alors le résultat de la combinaison sera « refuser ». Si au moins une règle est évaluée à « autoriser » et que l'ensemble des autres règles est évaluée à « non applicable », alors le résultat de la combinaison sera « autorisé ». Dans les autres cas (tout « non applicable »), le résultat retourné sera « non applicable ».
- « Préséance de l'autorisation » : dans l'ensemble de toutes les règles d'une politique, si au moins une règle est évaluée à « autoriser », alors le résultat de la combinaison sera « autoriser ». Si au moins une règle est évaluée à « refuser » et que l'ensemble des autres règles est évaluée à « non applicable », alors le résultat de la combinaison sera « refuser ». Dans les autres cas (tout « non applicable »), le résultat retourné sera « non applicable ».

- « Premier applicable » : chaque règle est évaluée dans l'ordre dans lequel elle est listée dans la politique. L'évaluation de la politique s'arrête dès lors qu'une règle s'applique, c'est-à-dire que la cible correspond et que les éventuelles conditions soient satisfaites. Le résultat de l'évaluation de la politique correspondra à la décision de cette règle. Si aucune règle ne peut s'appliquer, le résultat sera « non applicable ».

Il est également possible au sein d'une politique de spécifier un ensemble d'obligations.

3.4.4.5 Ensemble de politiques

L'entité « Ensemble de politiques » combine plusieurs politiques auxquelles on ajoute également une cible et un ensemble d'obligations.

Le modèle XACML d'un ensemble de politiques est repris à la Figure 56.

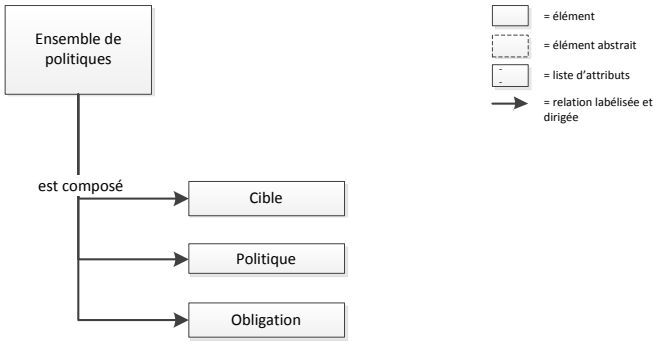

Figure 56 Concept ensemble de politiques du modèle XACML (OASIS, 2005 p. 19)

L'objectif d'un « Ensemble de politiques » est similaire à celui d'une politique elle-même, c'est-à-dire regrouper à un niveau supérieur un ensemble de politiques afin d'en optimiser leur exploitation. De cette manière, XACML dispose d'une structure à 3 niveaux : les règles sont incluses dans les politiques qui sont elles-mêmes incluses dans des ensembles de politiques. Les raisons de l'utilisation d'une cible unique et d'obligations sont identiques à celles de l'entité « Politique ».

Etant donné la possible présence de plusieurs politiques au sein d'un même ensemble de politiques et le fait que chacune d'entre elles soit associée à une décision, XACML propose un ensemble d'algorithmes de combinaison de politiques afin de pouvoir évaluer l'ensemble de politiques. Quatre comportements sont définis dont 3 ont déjà été évoqués pour l'entité « Politique » auxquels il suffit d'adapter la définition pour faire correspondre aux politiques et non plus aux règles (OASIS, 2005 p. 138) :

- « Préséance du refus » : voir §3.4.4.4 ;
- « Préséance de l'autorisation » : voir §3.4.4.4 ;
- « Premier applicable » : voir §3.4.4.4 ;

- « Un seul applicable » : Dans l'ensemble de toutes les politiques d'un ensemble de politiques, si aucune politique n'est applicable, le résultat de la combinaison de politiques est « non applicable ». Si plusieurs politiques sont considérées comme « applicables », le résultat donné est dit « indéterminé ». Si une et une seule politique est applicable, alors le résultat de la combinaison de politiques correspondra à l'évaluation de celle-ci.

3.4.5 Conclusion

Le XACML ne peut être utilisé seul ; celui-ci doit être combiné à un langage de contrôle d'accès afin de pouvoir bénéficier d'une sémantique complète. En plus d'un apport syntaxique, le XACML apporte également une architecture ainsi que différents concepts permettant :

- D'assurer à l'aide de règles, politiques, … une sécurité dans l'accès aux objets du système ;
- Une mise en place de ce système indépendamment du contrôle d'accès déjà présent utilisé dans le système.

L'inconvénient de ce modèle réside en la multitude de règles, politiques, ensemble de politiques à mettre en place. En effet, toutes celles-ci sont à la charge de l'administrateur du système et plus leur nombre est important, plus il est compliqué de maintenir un système sécurisé à jour.

3.5 Conclusion générale

Le présent chapitre avait pour objectif de décrire les principaux langages d'expression de droit, à savoir ODRL, XrML et MPEG-21 REL, ainsi qu'un langage de politiques, à savoir XACML.

Le langage d'expression de droits ODRL, dans sa partie modèle de données, regroupe les entités principales définies au §3.3.2. Une distinction est cependant présente pour le composant « Sujet ». En effet, cette séparation se situe au niveau du rôle liant le sujet à l'objet en question : il peut le consommer ou être à l'origine de sa création, production ou distribution. Le composant « Opération » est divisé en deux parties également, la première concerne les droits dont dispose un sujet, et la seconde traite des obligations à réaliser avant de pouvoir bénéficier de ces droits. L'ODRL subdivise ce composant en deux parties distinctes. Les composants additionnels permettent à ce langage de bénéficier d'un système d'invalidation de permission ainsi que d'un système central proposant pour chaque entité, un ensemble d'informations permettant de les décrire.

Le langage d'expression de droits XrML regroupe également les entités principales définies au §3.3.2. De même que pour l'ODRL, une distinction est présente au sein de l'entité « Sujet ». D'une part nous avons les consommateurs et d'autre part, les personnes à l'origine de la création de licence de permissions. L'architecture de ce modèle repose sur un

schéma de base sur lequel s'emboite une série de schémas apportant des extensions supplémentaires. Grâce à cela, il est possible d'exprimer des exigences, absentes du schéma de base.

Un développement en parallèle à celui du XrML a amené la création d'un nouveau langage d'expression de droit, le MPEG-21 REL. Basé sur le schéma de base du XrML, il utilise également une des extensions de celui-ci. La principale différence se situe au niveau de la représentation des contenus digitaux. En effet, le MPEG-21 REL a proposé une nouvelle version du schéma d'extension de contenu défini dans XrML, à savoir l'extension Multimédia. De plus, dans MPEG-21, un dictionnaire de données permettant la mise en place d'un standard pour la sémantique à utiliser a été développé en parallèle à l'extension Multimédia.

Le langage de protocoles XACML, reconnu en tant que standard par OASIS, n'est pas un langage de droits en tant que tel (comme c'est le cas pour l'ODRL, le XrML et le MPEG-21 REL). Il est utilisé pour retourner le résultat d'une décision prise suite à une requête d'utilisation d'un objet par un utilisateur. Mais il a l'avantage de pouvoir proposer un mécanisme à la fois de permission mais également d'interdiction. C'est en effet une limitation des RELs cités ci-dessus qui ne proposent que des mécanismes de permission. Il est très difficile, voire impossible à l'aide de ceux-ci de pouvoir proposer une liberté d'action aussi souple que ne le fait XACML. De plus, le langage XACML permet de regrouper des règles similaires touchant des cibles identiques afin de pouvoir mettre en place un système hiérarchique correspondant par exemple à la hiérarchie organisationnelle d'une entreprise. Néanmoins ce langage XACML a besoin d'être combiné à un système de contrôle d'accès afin de pouvoir bénéficier d'une sémantique établie.

3.5.1 Concepts en commun

L'analyse de tous ces modèles a permis de mettre en évidence un ensemble commun d'entités présentes dans chacun d'eux. On a pu remarquer dans chacun des modèles, la présence des 4 concepts de base, proposés comme source commune selon Chong et al. (§3.3.2), à savoir le sujet, l'objet, l'opération et la contrainte. Néanmoins, tous ont subdivisé le concept d'opération en deux parties distinctes, à savoir l'action et l'exigence. Cette représentation permet une compréhension plus claire quant à leur utilisation et dès lors, nous utiliserons cette distinction lors de l'élaboration du modèle générique. De même que pour le concept « Opération », le concept « Sujet » est subdivisé en deux selon les modèles, d'une part nous avons les consommateurs et d'autre part les émetteurs de droits. Toutes ces différences seront expliquées plus en détail ci-après. Chaque modèle utilise une « notation » qui leur est propre mais il est possible de faire correspondre leurs entités les unes aux autres (voir Tableau 4).

Tableau 4 Liste des concepts en commun dans les RELs et XACML

	ODRL	XrML	MPEG-21 REL	XACML
Sujet	x	x	x	x
Objet	x	x	x	x

	ODRL	XrML	MPEG-21 REL	XACML
Opération	x	x	x	x
Contrainte	x	x	x	x
Attribut	x	x	x	x
Décision				x
Invalidation	x			
Cible				x
Permission	x	x	x	x
Droit	x	x	x	x

A l'aide de ce tableau, on peut établir un modèle générique satisfaisant à chacun des modèles étudiés séparément. Comme cela avait été réalisé pour le modèle générique des contrôles d'accès, il est nécessaire dans un premier temps de définir de manière générique chacune des entités.

3.5.1.1 Sujet

Le sujet est l'entité qui dispose d'un certain nombre de permissions lui permettant de réaliser une action sur un objet dans un contexte particulier.

Dans le modèle ODRL, la notion de sujet est présentée sous l'entité « Sujet ». Celui-ci inclut les utilisateurs finaux et les titulaires de droits. Généralement, les utilisateurs finaux sont définis comme étant les consommateurs de l'objet. Les titulaires de droits correspondent aux sujets jouant un rôle dans la création, la distribution ou la production d'un objet.

Dans le modèle XrML et MPEG-21 REL, la notion de sujet est représentée par le concept de même nom et celui d' « Emetteur ». Le sujet correspond à l'entité autorisée à effectuer une action sur un objet et l'émetteur accorde les permissions aux différents sujets.

Dans le modèle XACML, le sujet est identifié sous le concept portant le même nom et est défini comme le demandeur d'une requête (une demande d'accès).

Le concept « Sujet » est donc présent dans chacun des modèles étudiés. Cependant, un raffinement existe pour certains de ces modèles : le sujet est subdivisé en deux entités. Ces deux entités sont : les utilisateurs finaux, que l'on peut appeler « Consommateurs » et les utilisateurs impliqués dans la création et/ou la distribution d'un objet, appelés les « Titulaires de droits ».

La subdivision de ce concept en deux entités est jugée nécessaire pour l'élaboration du modèle générique car elles ne possèdent pas les mêmes propriétés. Les titulaires de droits sont des consommateurs disposant de droits supplémentaires comme la possibilité d'émettre de nouveaux droits mais également de percevoir des revenus suite à leur utilisation.

Le modèle générique du sujet est représenté à la Figure 57.

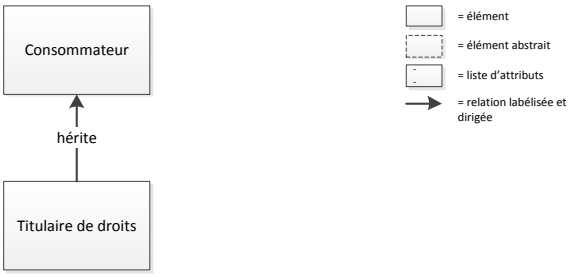

Figure 57 Concept sujet du modèle générique des expressions de droits/politiques

Il est possible de faire correspondre ce modèle générique à l'ensemble des modèles étudiés :

- Pour ODRL, le XrML et MPEG-21 REL il y a une correspondance identique. Seuls les intitulés des entités du modèle générique diffèrent :
  - L'entité « Consommateur » faisant référence à l'entité « Sujet » dans ODRL, XrML et MPEG-21 REL ;
  - L'entité « Titulaire de droits » faisant référence à l'entité « Titulaire de droits » dans ODRL et « Emetteur » dans XrML et MPEG-21 REL ;
- Pour le modèle XACML, on ne fait pas de distinction entre ces deux entités, tous les acteurs sont regroupés dans le même concept « Sujet ».

3.5.1.2 Objet

L'objet est la ressource sur laquelle un sujet effectue une action.

Le modèle ODRL intègre la notion d'objet sous le concept d'« Objet » qui englobe aussi bien des contenus physiques (DVD, CD,…) que digitaux (document texte, image,…).

Dans le XrML et le MPEG-21 REL, la notion d'objet est également intégrée sous le concept d' « Objet ». Elle concerne les objets digitaux (tels que les e-books, les images, les fichiers audio ou vidéo), les services (tels que les services e-mail, les services de transaction Business-to-Business) ou encore des informations détenues par le « Sujet » (telles qu'un nom, une adresse mail,…).

Dans le modèle XACML, la notion d'objet est reprise sous le concept de même nom et correspond à la ressource pour laquelle on veut obtenir un droit d'accès.

Le concept d'objet est donc présent dans chacun des modèles considérés, avec parfois quelques nuances concernant sa nature. Dans la définition du modèle générique, la notion d'objet concerne tout type de ressource. Il peut donc s'agir d'objet digital, physique, d'un service,…

3.5.1.3 Opération

Dans les RELs et dans le XACML, le concept d'opération est divisé en deux parties : l'opération et l'obligation. Celles-ci sont décrites ci-après.

3.5.1.3.1 Opération/Action

Une opération est une action que peut faire un sujet sur un objet.

Dans le modèle ODRL, ce concept est présenté dans la notion d' « Opération » correspondant à l'ensemble des actions autorisées sur un objet. Ce concept dans l'ODRL comprend les quatre groupes de permissions similaires suivants :

- Utilisation (ex : afficher, imprimer, …)
- Réutilisation (ex : modifier, extraire, …)
- Transfert (ex : vendre, louer, …)
- Gestion de l'objet (ex : déplacer, effacer, …)

Le XrML et le MPEG-21 REL supportent également ce concept, défini comme étant l'élément de la licence accordant une opération à un sujet sur un objet. Ils sont regroupés en 5 catégories :

- Rendu (ex : imprimer, …)
- Transfert (ex : copier, transférer, …)
- Réutilisation (ex : éditer, extraire, …)
- Gestion (ex : effacer, sauvegarder, …)
- Configuration (ex : installer, …)

Dans le langage XACML, la liste des actions n'est pas définie car il n'existe pas de sémantique propre pour ce concept. Il est donc nécessaire d'utiliser un langage de contrôle d'accès proposant une liste d'actions réalisables ou de les définir manuellement au sein du modèle.

Le concept d'opération/action est donc présent dans chacun des modèles considérés, avec parfois quelques nuances concernant les actions réalisables. Dans la définition du modèle générique, la notion d'opération correspond à un ensemble d'actions proposées au sujet sur un objet.

3.5.1.3.2 Obligation/Obligation

Une obligation peut être vue comme une pré-condition devant être satisfaite pour obtenir la permission d'exécuter l'opération demandée.

Le modèle ODRL intègre ce concept dans la notion d' « Opération » et plus précisément, dans la notion d' « Exigence ». Ce concept est composé de trois entités :

- Taxe
- Interaction
- Utilisation

Dans les modèle XrML et MPEG-21 REL, ce concept est intégré dans la notion de « Condition », plus précisément dans les conditions définies dans l'extension de schéma

standard ; présent à la fois dans le XrML et le MPEG-21 REL. Il est composé des entités suivantes :

- Taxe
- Rechercher approbation
- Rapport suivi
- Requête suivi

Tout comme le concept « Opération », la liste des obligations n'est pas définie dans le langage XACML. Néanmoins, la syntaxe d'utilisation est présente, ce qui permet dès lors l'expression d'obligation dans ce langage.

Le concept d'obligation est donc présent dans chacun des modèles considérés, avec parfois quelques différences concernant les actions à devoir réaliser. Dans la définition du modèle générique, la notion d'obligation correspond à un ensemble d'actions rendues obligatoires suite à l'autorisation d'une opération.

3.5.1.4 Contrainte

La contrainte correspond aux restrictions sur les opérations d'un objet.

Le concept de contrainte est bien présent dans ODRL sous la notion portant le même nom. La contrainte ODRL est un ensemble de restrictions sur les permissions d'un objet. Différents types de contraintes sont définis :

- Utilisateur
- Périphérique
- Limites
- Temporelle
- Aspect
- Cible

Dans les modèle XrML et MPEG-21 REL, le concept de contrainte est présent également. Les possibilités d'extension offertes par ces modèles permettent de définir de nouvelles contraintes applicables à des domaines d'application particuliers. Les différents types de conditions qui sont définis sont :

- Limite exercice
- Territoire
- Intervalle flottant
- Mesure validité
- Périodique

Le XACML intègre également ce concept. Son utilisation à deux niveaux au sein des politiques de contrôle permet :

- D'accélérer les recherches de permissions s'appliquant à la situation actuelle

- D'affiner les prises de décision en utilisant des fonctions d'évaluations plus étendues

L'ensemble des fonctions d'évaluation définies peuvent être de plusieurs types :

- Arithmétique
- Logique
- Temporelle
- …

L'analyse des contraintes dans les différents modèles considérés permet d'aboutir à la définition générique suivante du concept de contrainte : la contrainte correspond aux restrictions sur les opérations d'un objet.

3.5.2 Concepts additionnels

3.5.2.1 Attributs

Il ne s'agit pas à proprement parler d'un concept à part entière mais il est exprimé en tant que tel dans le modèle ODRL sous le nom de « Contexte ». Il offre dans ce modèle une série d'informations susceptibles d'être utilisées par les entités du modèle. Il s'agit en réalité des caractéristiques communes à toutes ces entités. Communes car il en existe comme les « Titulaire de droits » qui contiennent, en plus des caractéristiques définies dans le concept « Contexte », des informations additionnelles telle que le pourcentage de revenu, …

Dans les modèles XrML, MPEG-21 REL et XACML, les attributs permettent également de définir plus en détail les différentes entités comme le sujet, l'objet, l'environnement, … et elles sont propres à chacune d'entre elles.

En plus d'apporter des caractéristiques aux différentes entités des modèles, ces attributs permettent de définir un contexte particulier d'utilisation. En mettant ces valeurs au profit des contraintes par exemple, il est possible de spécifier une opération autorisée à condition de se trouver à un endroit particulier et à un moment particulier.

L'intérêt d'intégrer le concept contexte dans le modèle générique est de pouvoir centraliser dans une seule entité les attributs communs de chacune des entités du modèle et de bénéficier d'une gestion administrative plus aisée (grâce à la centralisation, à la facilité d'ajout potentiel d'attribut commun à toutes les entités).

3.5.2.2 Cible

Cette entité correspond à un ensemble de conditions simples s'appliquant au sujet et à l'objet. Elle permet donc de restreindre l'application d'un droit et/ou d'une permission à un certain groupe de sujets et sur un certain groupe d'objets. Elle est donc liée aux entités « Sujet » et « Objet » mais également aux entités « Droit » et « Permission ».

Dans les modèles ODRL, XrML et MPEG-21 REL, cette fonctionnalité est présente dans l'entité « Contexte ».

Dans le modèle XACML, elle est présente sous le concept de même nom.

Etant donné qu'il ne s'agit pas d'un concept à part entière mais plutôt d'un moyen simple et rapide de limiter un droit et une permission à un groupe de sujets et/ou objet, il n'est pas nécessaire d'établir un schéma générique pouvant s'appliquer à chaque modèle. Toutefois, son utilisation est jugée importante car elle permet de simplifier le schéma en structurant de manière plus claire les mécanismes de limitation de droit et de permission.

La structure choisie est représentée à la Figure 58.

Figure 58 Concept cible du modèle générique des expressions de droits/politiques

3.5.2.3 Décision

La décision est l'entité qui permet de faire basculer le résultat d'une permission soit en une permission autorisée, c'est-à-dire une autorisation, soit en une permission refusée, c'est-à-dire une interdiction. Cette entité, pouvant prendre deux valeurs (autorise, refuse), permet, si les conditions sont respectées, d'autoriser ou de refuser l'action stipulée dans cette permission. Elle est donc liée à l'entité « Permission » et permet au modèle l'expression d'autorisation et d'interdiction.

Dans le modèle ODRL, cette entité n'existe pas.

Dans les modèles XrML et MPEG-21 REL, cette entité n'est pas définie non plus.

Dans le modèle XACML, elle est présente sous le même nom.

3.5.2.4 Invalidation

L'invalidation correspond à l'entité annulant une permission à la suite de la satisfaction de diverses contraintes. Cette entité permet pour l'action d'une permission d'exprimer une condition particulière qui, si elle se produit, fait en sorte que l'action n'est plus permise, c'est-à-dire que la permission n'est plus valide. Elle est donc liée à l'entité « Permission » et « Contrainte ».

Dans le modèle ODRL, elle est présente sous le même nom.

Dans les modèles XrML et MPEG-21 REL, cette entité n'existe pas.

Dans le modèle XACML, cette entité n'est pas définie non plus.

Le modèle générique de l'invalidation est représenté à la Figure 59.

Figure 59 Concept invalidation du modèle générique des expressions de droits/politiques

3.5.2.5 Permission

Cette entité n'est pas réellement une entité non plus. Elle est utilisée comme un mécanisme d'agrégation permettant de relier entre elles un certain nombre d'entités.

Dans le modèle ODRL, elle est présentée sous le nom « Permission » et regroupe toutes les opérations autorisées pour un sujet. De plus, comme chaque opération est liée aux entités « Contrainte » et « Obligation », elle définit par conséquent l'ensemble de toutes les permissions qu'un sujet possède sur un objet en indiquant les contraintes et les obligations à satisfaire pour bénéficier de l'autorisation d'exercer cette opération.

Dans les modèles XrML et MPEG-21 REL, elle est présentée sous le nom « Autorisation » et regroupe les concepts de « Sujet », « Opération », « Objet », « Contrainte » et « Obligation ». Elle définit quel sujet possède l'autorisation d'effectuer l'opération sur l'objet et quelles en sont les restrictions et obligations qui en découlent.

Dans le modèle XACML, elle est présentée sous le nom de « Règle » et regroupe les concepts de « Sujet », « Objet », « Opération » et « Environnement » mais plus particulièrement les contraintes liées à tous ceux-ci. Y sont également présentes des contraintes plus détaillées permettant de cibler au mieux la règle ; mais également une entité de décision. En fonction de la satisfaction de ces contraintes et de la valeur prise par l'entité « Décision », la règle refusera ou autorisera l'exécution de l'opération.

Etant donné qu'il ne s'agit pas d'un concept à part entière mais plutôt d'un moyen de structurer les permissions, il n'est pas nécessaire de trouver un schéma générique pouvant s'appliquer à chaque modèle. Néanmoins, son utilisation est indispensable car cela apporte à la permission une sémantique claire. La structure choisie est représentée à la Figure 60.

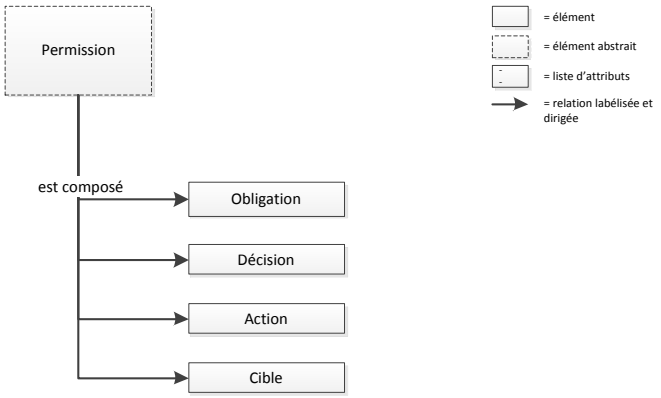

Figure 60 Concept permission du modèle générique des expressions de droits/politiques

3.5.2.6 Droit

Cette entité est à considérer comme l'entité « Permission », c'est-à-dire un mécanisme d'agrégation permettant de relier entre elles un certain nombre d'entités.

Dans le modèle ODRL, elle est représentée sous le même nom « Droit » et regroupe l'ensemble des entités du modèle utilisées lors de la création d'un droit numérique. Elle relie par conséquent l'entité « Objet », l'entité « Sujet » ainsi que l'entité « Permission » regroupant elle-même également un ensemble d'entités.

Dans les modèles XrML et MPEG-21 REL, elle est présentée sous le nom de « Licence ». Son utilisation permet, tout comme pour le modèle ODRL, de regrouper les entités nécessaires à l'élaboration d'un droit numérique / licence. Elle relie l'entité « Emetteur » avec l'entité « Autorisation/Permission ».

Le langage XACML utilise ce mécanisme d'agrégation sous l'appellation « Politique ». Celui-ci regroupe les entités « Permission », « Obligation » et un ensemble de « Condition ».

Etant donné qu'il ne s'agit pas d'un concept à proprement parler mais plus d'un moyen de structurer le droit numérique, il n'est pas nécessaire de trouver un schéma générique pouvant s'appliquer à chaque modèle. Néanmoins, son utilisation est indispensable car cela apport au droit numérique une sémantique claire. La structure choisie est représentée à la Figure 61.

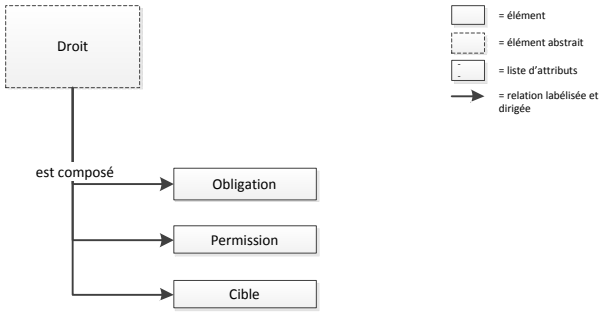

Figure 61 Concept droit du modèle générique des expressions de droits/politiques

3.5.3 Modèle générique

Le modèle générique pouvant s'appliquer aux langages d'expression de droits et au langage de politiques analysés dans ce chapitre est représenté de la manière suivante (sous forme de diagramme de classe) :

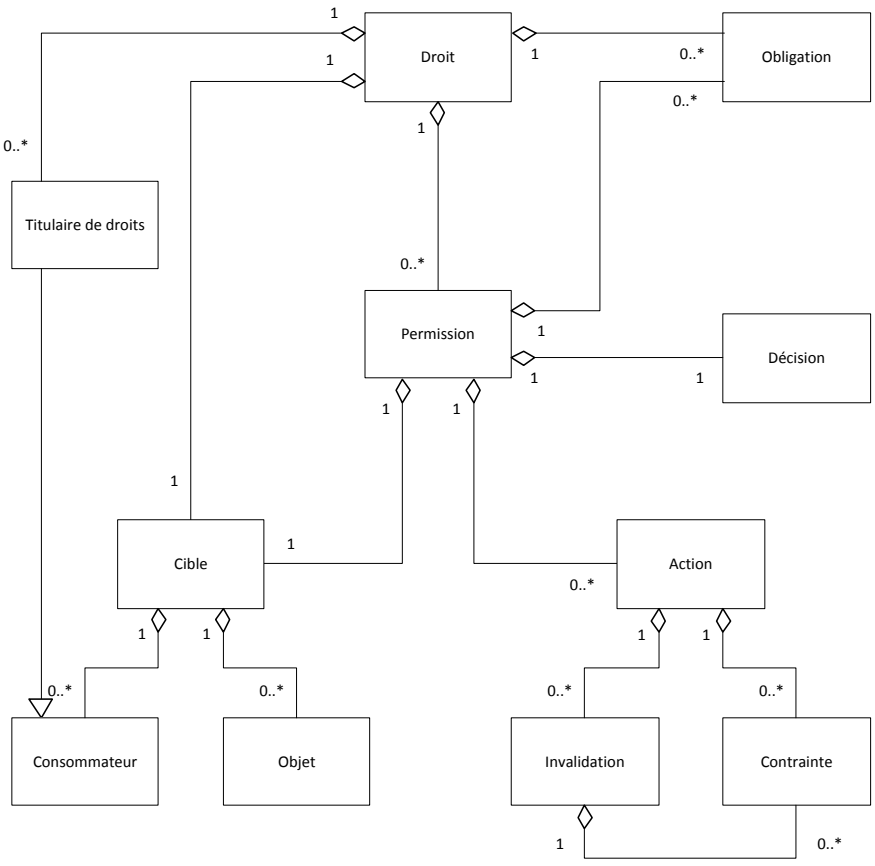

Figure 62 Modèle générique des expressions de droits/politiques

Dans ce diagramme, l'entité principale est représentée par le « Droit » au sein duquel on retrouve tout d'abord le « Titulaire de droits » à l'origine de la création de celui-ci. L'utilisation d'une « Cible » à ce niveau permet de limiter le champ d'application de ce droit à un ensemble de « Sujets » et un ensemble d' « Objets ». Ce droit regroupe de plus un ensemble d' « Obligations » s'appliquant à toutes les « Permissions » définies dans ce droit.

Ces permissions incluent également une cible afin de limiter plus en profondeur son application. En effet, on pourrait préciser dans un premier temps un groupe de sujets pour lequel le droit s'appliquerait et par la suite indiquer de manière plus détaillée le ou les sujet(s) appartenant à ce groupe pour le(s)quel(s) la ou les permission(s) entrerai(en)t en vigueur. Une permission est détaillée par des « Actions » auxquelles s'appliquent un certain nombre de « Conditions » limitant son utilisation à un cadre spécifique défini par le ou les titulaire(s) de droits. Il leur est également permis de préciser un ou des critères d' « Invalidation » qui en cas de conformité (l'ensemble des critères est satisfait) rendraient invalide l'action. Pour ce faire, le système est pourvu d'un mécanisme de « Décision » qui spécifie si une permission est autorisée ou refusée. De cette manière, un titulaire de droits a la possibilité de créer une permission s'appliquant à une cible pour laquelle une action est autorisée dans le cas où les conditions sont respectées, mais dont la décision finale est refusée. Cela signifie que pour cette cible, dans de telles conditions, cette action interdite. A l'aide de ce mécanisme, il est dès lors possible de définir des autorisations considérées comme des permissions autorisées (dans le cadre d'une politique dite fermée) mais également des interdictions, considérées comme des permissions refusées (dans le cadre d'une politique ouverte).

Pour satisfaire le modèle ODRL, il faut inclure l'entité « Contexte » au diagramme et lier celle-ci aux différentes entités du modèle. Cependant, pour alléger le diagramme et par soucis de clarté, elle n'a pas été représentée. Mais son utilité est toutefois à discuter car l'ajout des attributs présents dans cette entité dans l'ensemble des autres entités apporterait la même fonctionnalité. Cependant, la gestion en serait alourdie.

Les modèles XrML et MPEG-21 REL sont entièrement satisfaits par ce modèle générique. L'ensemble des entités et fonctionnalités de chaque modèle est présent.

Le modèle XACML inclus une entité supplémentaire, l' « Environnement » dont le but est d'apporter un ensemble d'informations supplémentaires sur celui-ci. Cela permet au modèle d'évaluer plus rapidement les conditions dont l'information serait déjà reprise dans cette entité ; il ne serait pas nécessaire d'en faire la demande au système (pour par exemple connaître l'heure actuelle). Cette nécessité de rapidité d'évaluation n'a pas été jugée assez importante au point d'en faire sa représentation ; l'évaluation de conditions sur l'environnement étant réalisé par l'entité « Condition » et le système.

4 Définition d'un modèle générique

4.1 Introduction

L'objectif de ce chapitre est de définir un modèle générique. Pour ce faire, une analyse des concepts présents dans les modèles de contrôle d'accès, dans les langages d'expression de droits et dans un langage politiques a été réalisée et présentée dans les chapitres précédents.

Cette analyse a permis d'aboutir à la création de deux modèles génériques :

- le modèle générique des modèles de contrôle d'accès (présenté dans le §2.3.2) ;
- le modèle générique des langages d'expression de droits et langage de politiques (présenté dans le §3.5.3).

La définition d'un modèle générique « global » consiste dès lors à :

- comparer les modèles génériques précédemment énoncés ;
- proposer un ensemble de concepts ainsi que leur définition ;
- proposer un modèle générique unifié sur base de ces concepts.

4.2 Comparaison des modèles génériques de contrôle d'accès et d'expression de droits/politiques

Le présent chapitre a pour objectif de définir les principaux concepts mis en évidence dans les modèles génériques de contrôle d'accès et d'expression de droits/politiques, ainsi que de proposer une comparaison de ceux-ci.

Des analyses précédentes, on peut en conclure aisément que le modèle générique élaboré pour les contrôles d'accès est schématiquement plus simple que le second (il est composé d'un nombre d'entités inférieur et le nombre de relations entre les entités l'est également). Dès lors, la comparaison de ces deux modèles se fera en prenant le premier comme valeur de référence : si celui-ci est satisfait par le deuxième, alors le modèle générique d'expression de droits/politiques pourra prétendre au titre de modèle générique « global ».

Le modèle générique de contrôle d'accès est composé de 4 entités, à savoir le « Sujet », l' « Objet », l' « Opération » et le « Contexte » où l' « Opération » peut être vue comme étant soit une autorisation, soit une obligation, soit une interdiction. Le contexte, quant à lui, permet de décrire l'environnement dans lequel le sujet souhaite effectuer l'opération.

L'ensemble des entités issues de ce modèle est comparé au second modèle et repris dans le Tableau 5.

Tableau 5 Comparaison des entités des modèles génériques

Modèle générique des contrôles d'accès	Modèle générique des expressions de droits/politiques
Sujet	Consommateur
Objet	Objet
Contexte	Contrainte
Opération - Autorisation - Obligation - Interdiction	- - Action et « Décision » autorisée - Obligation - Action et « Décision » refusée

On constate à l'aide de ce tableau que l'ensemble des entités ainsi que les fonctionnalités du modèle générique des contrôles d'accès est inclus dans celui des expressions de droits/politiques. En effet :

- Le sujet dans les contrôles d'accès est équivalent au consommateur dans les expressions de droits/politiques
- L'objet est présent dans les deux modèles génériques
- Le contexte défini dans l'un correspond à la contrainte de l'autre
- Les notions d'autorisation, d'obligation et d'interdiction sont regroupées dans l'entité opération dans le modèle générique des contrôles d'accès. Dans le modèle des expressions de droits/politiques, ces notions présentes mais représentées par des entités distinctes :
    - L'autorisation du modèle générique des contrôles d'accès correspond à la combinaison des entités « Action » et « Décision » autorisée ;
    - L'obligation est définie dans l'entité « Obligation » ;
    - L'interdiction correspond à la combinaison des entités « Action » et « Décision » refusée.

De plus, tous deux utilisent la notion d'attributs permettant de décrire plus précisément chaque entité.

On peut en conclure que le modèle générique des contrôles d'accès est satisfait avec la représentation du modèle générique des expressions de droits/politiques. Nous pouvons dès lors utiliser celui-ci pour proposer un modèle générique global.

4.3 Proposition d'un modèle générique

Le modèle générique est constitué des entités suivantes :

- le « Sujet » est l'entité qui dispose d'un certain nombre de permissions lui permettant de réaliser une action sur un objet dans un contexte particulier.
- l' « Objet » est la ressource sur laquelle un sujet effectue une action.
- l' « Opération/Action » correspond à un ensemble d'actions proposées au sujet sur un objet.

- l' « Obligation » peut être vue comme une pré-condition devant être satisfaite pour obtenir la permission d'exécuter l'opération/action demandée.
- la « Contrainte » correspond aux restrictions sur les opérations d'un objet.
- la « Cible » correspond à un ensemble de conditions simples s'appliquant au sujet et à l'objet
- la « Décision » est l'entité qui permet de faire basculer le résultat d'une permission soit en une permission autorisée, c'est-à-dire une autorisation, soit en une permission refusée, c'est-à-dire une interdiction
- l' « Invalidation » correspond à l'entité annulant une permission à la suite de la satisfaction de diverses contraintes
- la « Permission » n'est pas réellement une entité. Elle est utilisée comme un mécanisme d'agrégation permettant de relier entre eux un certain nombre d'entités.
- le « Droit » est à considérer comme l'entité « Permission », c'est-à-dire un mécanisme d'agrégation permettant de relier entre elles un certain nombre d'entités.

L'ensemble de ces entités est mis en relation et représenté sous forme d'un diagramme de classe à la Figure 63.

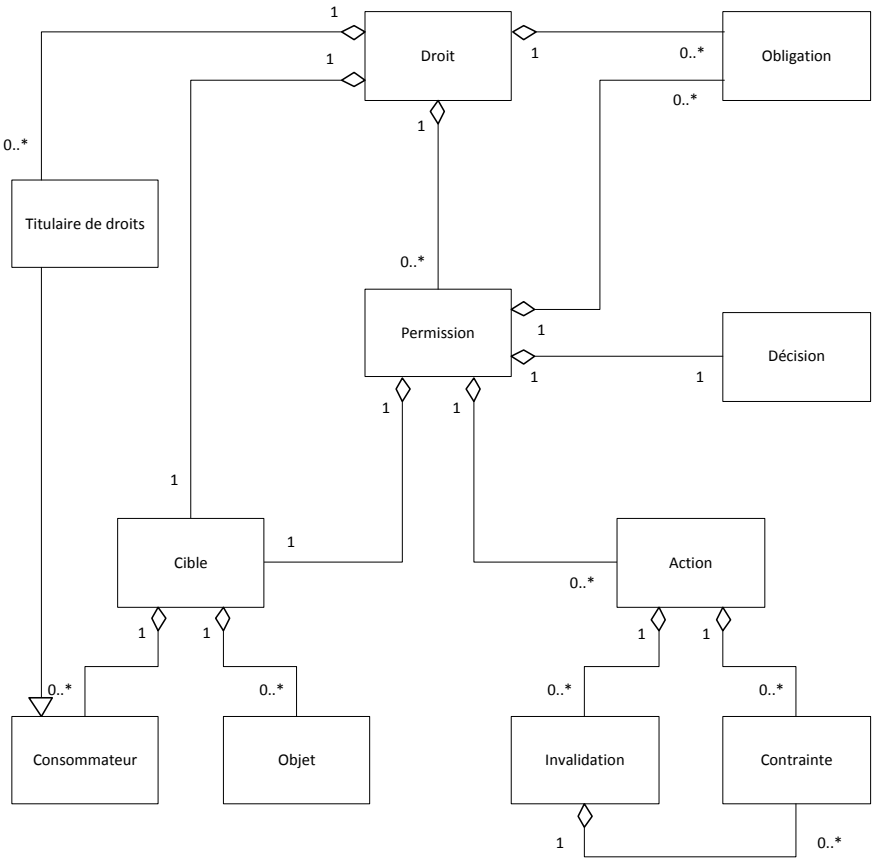

Figure 63 Modèle générique global

5 Conclusion

L'objectif du présent mémoire était de définir un modèle générique permettant à la fois de contrôler les accès et d'exprimer les droits des utilisateurs.

Pour ce faire, le travail s'est subdivisé en trois parties.

La première partie est axée sur la politique de sécurité et plus particulièrement celle des modèles de contrôle d'accès. Y ont été abordés, les modèles d'accès de base tels que le modèle de contrôle d'accès discrétionnaire (DAC), le modèle mandataire (MAC), le modèle basé sur les rôles (RBAC), le modèle contextuel basé sur les attributs (ABAC) mais également le modèle basé sur l'organisation Or-BAC. Ces modèles d'accès fournissent un langage pour l'application de politiques d'accès. Celui-ci est destiné à gérer l'autorisation sécuritaire et à connecter des sujets à des objets par le biais de déclarations de politiques formalisées. L'analyse de ces modèles a permis l'élaboration d'un modèle générique pouvant se substituer à chacun d'eux.

La deuxième partie de ce mémoire s'est intéressée aux technologies de gestion de droits, appelées DRM (Digital Rights Management). Celles-ci se basent sur un langage d'expression de droits (REL – Rights Expression Language), qu'il soit propriétaire ou bien ouvert. Dès lors, plusieurs langages d'expression de droits tels que ODRL, MPEG-21 REL ont été étudiés. La politique de contrôle de gestion XACML a également été étudiée dans cette deuxième partie car il s'agit d'un modèle générique, extensible et indépendant des modèles de contrôle d'accès. L'analyse de ces modèles a permis, comme pour les modèles de contrôle d'accès, l'élaboration d'un modèle générique applicable à tous.

La définition du modèle générique, réalisée dans la troisième partie, a été possible suite à une analyse de ces deux modèles génériques. L'objectif était de définir un modèle générique global pouvant s'appliquer à chacun des modèles considérés dans ce mémoire.

En conclusion, le modèle générique global défini dans le cadre du présent mémoire est donc applicable à chacun des modèles étudiés. Sa définition ouvre la perspective à des études plus avancées sur l'élaboration d'un modèle générique intégrant le modèle de protocole. Il pourrait également être complété en précisant l'ensemble des attributs de chaque entité. De plus, notre analyse s'est basée sur un certain nombre de modèles existants. D'autres modèles d'accès ou d'expressions de droits pourraient être analysés de manière à intégrer de nouveaux concepts dans notre modèle générique. Il serait également intéressant de démontrer l'efficacité de ce modèle en mettant à disposition un outil capable de convertir tout type de licence (ODRL, XrML, …) vers une licence compatible au modèle générique. Cela appuierait la définition de ce modèle en prouvant sa capacité de substitution.

6 Bibliographie

Abou El Kalam, Anas, et al. 2003. *Or-BAc : un modèle de contrôle d'accès basé sur les organisations.* 2003. 12 p.

Anderson, James P. 1972. *Computer security technology planning study.* 1972. Vol. 1, 33 p.

Bell, D.E. et La Padula, L.J. 1976. *Secure computer system: unified exposition and multics interpretation.* 1976. 29 p.

Biba, K.J. 1975. *Integrity considerations for secure computer systems.* 1975. 38 p.

Chong, Cheun Ngen, Etalle, Sandro et Hartel, Pieter H. 2003. *Comparing Logic-based and XML-based Rights Expression Languages.* 2003. 40 p.

Commission of the European communities. 2002. *Commission staff working paper: Digital Rights.* 2002. 17 p.

ContentGuard. 2000-2010. About XrML. *XrML.* [En ligne] 2000-2010. [Citation : 7 Avril 2011.] http://www.xrml.org/about.asp.

—. 2001a. *eXtensible rights Markup Language (XrML) 2.0 Specification - Part I : Primer.* 2001a. 20 p.

—. 2001b. *eXtensible rights Markup Language (XrML) 2.0 Specification - Part II : Core Schema.* 2001b. 46 p.

—. 2001c. *eXtensible rights Markup Language (XrML) 2.0 Specification - Part III : Standard Extension Schema.* 2001c. 43 p.

—. 2001d. *eXtensible rights Markup Language (XrML) 2.0 Specification - Part IV : Content Extension Schema.* 2001d. 35 p.

—. 2002. *XrML 2.0 Technical Overview - Version 1.0.* 2002. 24 p.

—. 2002. *XrML 2.1 Technical overview - draft version 0.1.* 2002. 16 p..

—. 2001. *XrML Software Development Kit User's Guide - Release 2.0.* 2001. 68 p.

—. 2000-2010. XrML...eXtensible rights Markup Language. *XrML.* [En ligne] 2000-2010. [Citation : 7 Avril 2011.] http://www.xrml.org/.

Cuppens, Frédéric et Miège, Alexandre. 2003. *Modelling Contexts in the Or-BAC Model.* GET/ENST Bretagne. 2003. 10 p.

—. sd. *Or-BAC Organization Based Access Control.* ENST Bretagne, Campus de Rennes. sd. 13 p.

Equipe SERES. 2008. OrBAC.org. [En ligne] 17 Novembre 2008. [Citation : 7 Avril 2011.] http://orbac.org/index.php?page=orbac&lang=fr.

European Commission. 2010. Europe's Information Society. *European Commission.* [En ligne] 27 Janvier 2010. [Citation : 28 Mars 2011.] http://ec.europa.eu/information_society/eeurope/2005/all_about/digital_rights_man/index_en.htm.

Ferraiolo, David F. et Kuhn, D. Richard. 1992. *Role-Based Access Control.* 1992. 11 p.

Gallagher, Patrick R. 1987. *A guide to understanding discretionary access control in trusted systems.* 1987. 32 p.

Harrison, Michael A., Ruzzo, Walter L. et Ullman, Jeffrey D. 1976. Protection in operating systems. *Communications of the ACM.* Août 1976, Vol. 19, 8, pp. 461-470.

IPR Systems Pty Ltd. 2002. *ODRL Initiative.* [En ligne] 8 Août 2002. [Citation : 7 Avril 2011.] 68 p. http://odrl.net/1.1/ODRL-11.pdf.

Ku, William et Chi, Chi-Hung. 2004. Survey on the technological aspects of the Digital Rights Management. *ISC.* 2004, pp. 391-403.

Latham, Donald C. 1985. *Department of Defense trusted computer system evaluation criteria.* 1985. 116 p.

Moving Picture Experts Group. sd.. MPEG Home Page. *MPEG.* [En ligne] sd. [Citation : 25 Avril 2011.] http://mpeg.chiariglione.org/.

Multimedia Description Schemas (MDS) Group. 2005. *Introducing MPEG-21 REL - an Overview.* Poznan : s.n., 2005. ISO/IECJTC1/SC29/WG11/N7427.

—. 2005. *Introduction to ISO/IEC 21000-6 Rights Data Dictionary.* [éd.] Chris Barlas. Poznan : s.n., 2005. ISO/IEC JTC1/SC29/WG11/N7433.

—. 2002. *MPEG-21 Overview v.5.* 2002. ISO/IECJTC1/SC29/WG11/N5231.

OASIS. 2005. *eXtensible Access Control Markup Language (XACML) Version 2.0.* 2005. 141 p.

—. 2011. OASIS eXtensible Access Control Markup Language (XACML) TC. *OASIS Advancing open standards for the information society.* [En ligne] 2011. [Citation : 20 Avril 2011.] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.

ODRL Initiative. 2000-2011. ODRL Initiative. [En ligne] 2000-2011. [Citation : 7 Avril 2011.] http://odrl.net/.

Parrott, David. 2001. *Requirements for a rights data dictionary and rights expression language. Technical Report version 1.0.* Reuters Ltd. 2001.

Red Hat, Inc. 2011. Multi-mevel security (MLS). *Product documentation.* [En ligne] 13 Janvier 2011. [Citation : 12 Avril 2011.] http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5/html/Deployment_Guide/sec-mls-ov.html.

Wang, Xin, et al. 2004. *The MPEG-21 Rights Expression Language ans Rights Data Dictionary.* 2004. 9 p.

Wikipedia. 2011. Dépôt (informatique). *Wikipedia L'encyclopédie libre.* [En ligne] 18 Mars 2011. [Citation : 24 Mai 2011.] http://fr.wikipedia.org/wiki/Repository.

WWF. 2010. Save as WWF, Save a tree. [En ligne] 2010. [Citation : 17 Avril 2011.] http://www.saveaswwf.com/en/.

Yuan, Eric et Tong, Jin. 2005. *Attributed Based Access Control (ABAC) for Web Services.* 2005. 9 p.

Fairplay (2003)

Le système DRM Fairplay a été développé par la société Apple afin de protéger les contenus digitaux disponibles sur iTunes, une application permettant en autre aux utilisateurs d'écouter ou d'acheter de la musique en ligne.

Fonctionnement général

Selon R. Venkataramu[7], le fonctionnement général de iTunes est le suivant :

- L'utilisateur choisit une chanson dans la base de données iTunes ;
- Le « Client iTunes »[8] envoie la requête et les informations de l'utilisateur au « Serveur iTunes »[9] ;
- Le Serveur iTunes envoie une URL et une clé de téléchargement au client iTunes ;
- Ce dernier télécharge alors le fichier et le décrypte en utilisant la clé. Le fichier décrypté correspond à la chanson protégée. Elle est stockée finalement sur le pc de l'utilisateur ;
- Le Client envoie un message au Serveur indiquant que la transaction a été effectuée avec succès.

Protection de contenu

Pour rappel, le procédé de protection de contenu a été présenté à la **Error! Reference source not found.** du §3.2.2.2. Le système Fairplay est doté des protections suivantes[10] :

- différentes clés de cryptage permettent de protéger le contenu ;
- tatouage numérique (les fichiers non « tatoués » ne peuvent par exemple pas être utilisés par iTunes) ;
- il utilise le format de fichier Quicktime en tant que conteneur sécurisé[11] permettant de lire de l'audio, de la vidéo, des images, du texte

[7] Ramya Venkataramu, *"Analysis and enhancement of Apple's Fairplay digital rights management"*, 2007, p10.
[8] Correspondant au "Client DRM" du schéma de fonctionnement général présenté à la **Error! Reference source not found.** du §3.2.2
[9] Correspondant au "Gestionnaire de licences" et à la « distribution » du schéma de fonctionnement général présenté à la **Error! Reference source not found.** du §3.2.2
[10] Ramya Venkataramu, *"Analysis and enhancement of Apple's Fairplay digital rights management"*, 2007, p8-11.
[11] Ramya Venkataramu, *"Analysis and enhancement of Apple's Fairplay digital rights management"*, 2007, p5.

Selon R. Venkataramu[12], le DRM Fairplay est, de plus, doté des restrictions suivantes :

- le nombre de copies sur CD ou dans une playlist est limité. L'utilisation du contenu est également limitée à un certain nombre de périphériques ;
- les contenus digitaux achetés peuvent être écoutés sur les systèmes d'exploitation Windows et sur tout périphérique de la marque Apple.
- la plupart des programmes d'édition de chansons ne sont pas fonctionnels avec les contenus iTunes.

[12] Ramya Venkataramu, *"Analysis and enhancement of Apple's Fairplay digital rights management"*, 2007, p4-5.

Helix (2003)

Le système DRM Helix a été développé par la société RealNetworks. En 2002, l'entreprise a annoncé Helix :

> Une plateforme de livraison de médias internet ouverte et complète et une Communauté permettant d'aider à la création de produits numériques et d'applications pour tout format, système d'exploitation ou périphérique[13]. [Notre traduction]

L'entreprise a dévoilé une partie du code source Helix via la Communauté que, selon RealNetworks[14], de nombreuses entreprises de technologie ont rejoint de manière à pouvoir intégrer les parties libres du code dans leurs produits.

Fonctionnement général

Selon RealNetworks[15], le système DRM Helix est constitué des composants de base[16] :

- un « Packager » qui s'occupe de la protection de contenu ;
- un « Serveur de licence » qui permet de gérer, d'autoriser et de déclarer les transactions effectuées sur un contenu. Ce Serveur joue plusieurs rôles :
    - vérifier les demandes de licence de contenus ;
    - délivrer des licences aux Clients DRM sécurisés et authentifiés (tel que RealOne) ;
    - fournir des informations d'audit afin de faciliter le payement des droits d'auteur.
- Un « Client DRM » qui permet le téléchargement et la lecture en continu de contenus sécurisés dans un environnement inviolable basé sur l'utilisation de règles spécifiées par le propriétaire de contenu.

En plus de ces trois composants, il existe également un plug-in permettant le streaming[17] de contenus protégés à partir du serveur universel Helix.

---

13 RealNetworks Inc, *"2003 Annual Report"*, sd., p2.
14 RealNetworks Inc; *"2003 Annual Report"*, sd., p2.
15 RealNetworks, *"Helix DRM – The first multi-format digital rights management platform for secure delivery of media to any device"*, 2002, p1.
16 Voir schéma de principe de fonctionnement présenté à la **Error! Reference source not found.** du §3.2.2
17 Le "streaming" permet aux utilisateurs de commencer la lecture d'un contenu multimedia sans devoir attendre la fin de son téléchargement

Protection de contenu

> Le Packager Helix utilise des algorithmes d'encryption efficaces et un conteneur sécurisé afin de prévenir l'utilisation non autorisée de contenu et de préparer le contenu pour la distribution via streaming, téléchargement ou d'autres méthodes de diffusion. Le contenu sécurisé et les règles d'utilisation associées pour l'utilisation de ce contenu sont stockés séparément de telle manière qu'un grand nombre de règles d'utilisation puissent être appliquée à un fichier unique[18]. [Notre traduction]

[18] RealNetworks, *"Helix DRM – The first multi-format digital rights management platform for secure delivery of media to any device"* ,2002, p1

OMA (2004)

Le système DRM OMA a été publié par l' « Open Mobile Alliance ». Celle-ci a été formée en 2002 par près de 200 entreprises de par le monde actives notamment dans les secteurs de la téléphonie mobile et dans la technologie de l'information. « Actuellement, le DRM OMA est reconnu comme standard dans l'industrie mobile. Deux standards OMA ont été publiés : la version 1.0 (septembre 2002) et la version 2.0 (mars 2006)[19]. » [Notre traduction]

Fonctionnement général

Différentes entités fonctionnelles entrent en jeu dans le système OMA DRM 2.0[20] :

- l' « Agent DRM » qui joue le rôle de Client DRM défini au §3.2.2.1. Il permet aux utilisateurs d'accéder à un contenu sécurisé ;
- l' « Emetteur de contenu » qui délivre le contenu sécurisé à l'Agent DRM. Il joue un rôle similaire à celui du Gestionnaire de licences défini au §3.2.2.1 ;
- l' « Emetteur de droits » (similaire à l'Editeur défini au §3.2.2.1) qui attribue les permissions et les contraintes au contenu DRM et génère des « Objets Droits ». Un Objet Droits est défini par l'OMA comme un document XML qui exprime les permissions et contraintes associées à un contenu DRM.

Protection de contenu

La protection de contenu se déroule en plusieurs étapes[21] :

- le « Packaging » : le contenu est intégré à un conteneur sécurisé. Le contenu DRM est encrypté à l'aide d'une clé d'encryption symétrique.
- l' « Authentification de l'Agent DRM » : tous les Agents DRM sont associés à un identifiant unique (paire clé publique/clé privée) et à un certificat. Ce dernier contient des informations complémentaires (telles que le nom du créateur, la version du programme,…). Cela permet aux Emetteurs de contenus et de droits d'authentifier l'Agent DRM de manière sécurisée.
- La « génération de l'Objet Droits » : en plus des permissions et des contraintes, l'Objet Droits peut aussi contenir la clé d'encryption symétrique. Cela permet de s'assurer que le contenu DRM ne peut être utilisé dans Objet Droits associé.
- La « protection de l'Objet Droits » : avant de transmettre l'Objet Droits, il est lié à l'Agent DRM cible (à l'aide d'une clé d'encryption par exemple). Cela permet de

19 Motodev (The Motorola developer network), *"Introduction of basics concepts in OMA DRM v1.0"*, 2007, p4.
20 OMA, *"DRM Architecture – Approved version 2.0.1"*, 2008, p8.
21 OMA, *"DRM Architecture – Approved version 2.0.1"*, 2008, p13.

s'assurer que seul l'Agent DRM cible peut accéder à l'Objet Droits et donc au contenu DRM. De plus l'Emetteur de droits signe l'Objet Droits.

- Livraison : l'Objet Droits ainsi que le conteneur sécurisé peuvent dès lors être transmis à l'Agent DRM cible.

Adobe Flash Access 2.0 (2009)

Le système DRM Adobe Flash Access a été développé par la société Adobe afin de protéger les contenus disponibles sur le web (via téléchargement ou streaming).

Fonctionnement général

Le DRM Adobe Flash Access 2.0 contient différents composants[22] :

- un « Packager » qui permet de protéger le contenu ;
- un « Serveur de licence » qui publie les licences liées à un ou plusieurs périphériques (similaire au Gestionnaire de licences présenté au §3.2.2.1)
- un « Client DRM » (Flash Player par exemple) qui acquiert les licences de contenu à partir du serveur de licence en utilisant un protocole sécurisé.

Pour acquérir ou visionner un contenu, l'utilisateur se connecte sur le site internet du fournisseur et choisit un contenu dans le catalogue. Le Client DRM charge le lecteur vidéo et se connecte au réseau de distribution de contenu pour amorcer le téléchargement ou le streaming du contenu. Le Client examine les métadonnées du contenu et les fournit au Serveur de licence qui va pouvoir délivrer une licence liée de manière cryptée au Client DRM.

Protection de contenu

Le mécanisme de protection de contenu est le suivant[23] :

- le DRM Flash Access protège le contenu dans un conteneur sécurisé (MPEG4(FLV)) ;
- le Packager insère ensuite des métadonnées dans le contenu et le crypte. Le titulaire de droits a également la possibilité de demander l'insertion de règles de sécurité avec le contenu. Le packager inclut ces politiques au contenu et crypte le fichier en utilisant une clé d'encryption de contenu générée aléatoirement.
- Le contenu sécurisé peut alors être transféré au réseau de distribution.

[22] Adobe, *"Adobe Flash Access 2.0 : Realize new sources of revenue and extend your market reach with a robust video content protection solution*", 2010, p3.
[23] Adobe, *"Adobe Flash Access 2.0 : Realize new sources of revenue and extend your market reach with a robust video content protection solution"*, 2010, p3.

PlayReady (2007)

Le DRM PlayReady a été développé par Microsoft pour les dispositifs portables.

Fonctionnement général

Les principales entités fonctionnelles entrant en jeu dans le système PlayReady sont les suivantes[24] :

- les « Périphériques » PlayReady et les clients PC utilisés pour la lecture de contenu sont capables d'acquérir un contenu protégé, d'interpréter une licence et de mettre en application les règles contenues dans la licence (ils ont un fonctionnement similaire au client DRM défini au §3.2.2.1) ;
- les « Serveurs de Packaging » de contenu s'occupent de la protection de contenu. Une fois protégé, le contenu est alors envoyé vers un serveur de distribution et les informations de licence sont transférées vers le Serveur de licence ;
- les « Serveur de distribution » qui sont utilisés pour stocker et distribuer le contenu ;
- les « Serveurs de licence » (similaire au Gestionnaire de licences) qui stockent les informations de protection de contenu et les droits pour l'utilisation du contenu. Avant qu'un client DRM ne puisse lire un contenu protégé, il doit acquérir une licence. Cela se fait à partir du « Serveur de licence ».

En plus de ces éléments de base, PlayReady dispose également de :

- « Contrôleurs de Domaine » qui servent à authentifier les périphériques. L'utilisateur doit enregistrer son périphérique et en échange, il reçoit un certificat de domaine qui prouve son authenticité et lui permet par la suite d'obtenir des licences et de lire des contenus ;
- « Serveurs de mesure » : les périphériques enregistrent le nombre de fois qu'un fichier est lu. Lorsqu'on les connecte à un pc ou à internet, les valeurs enregistrées sont uploadées vers le fournisseur de contenu. Cela permet à ce dernier d'évaluer les droits d'auteur.

Protection de contenu

Le DRM PlayReady protège le contenu dans un conteneur sécurisé.

Chaque licence contient les droits et restrictions, définissant exactement comment un contenu peut être utilisé et sous quelles conditions.

[24] Microsoft, *"Microsoft PlayReady Content Access Technology – White Paper"*, July 2008, p2.

DRM-X (2006)

Le DRM-X a été développé par la société Haihaisoft qui est active dans le domaine des DRM depuis 2006. La version la plus récente est le DRM-X 3.0 qui a été lancées en janvier 2011[25].

Fonctionnement général

Le système DRM-X 3.0 consiste en 3 parties[26] :

- un « Serveur » jouant plusieurs rôles :
    - il sert de panneau de contrôle en ligne permettant de gérer les permissions, les utilisateurs et les groupes ;
    - il fournit une interface XML que les fournisseurs peuvent intégrer à leurs propres systèmes ;
    - il permet d'authentifier les utilisateurs et de générer des licences en temps réel pour les utilisateurs finaux ;
- un « Outil de cryptage en ligne » contenant un Packager de contenu audio et vidéo et un Packager pour les médias en format pdf ;
- une « Plateforme de lecture » de contenus digitaux et une plateforme de lecture et d'impression de documents pdf.

Protection de contenu

Le DRM-X 3.0 utilise les mécanismes de protection suivants[27] :

- encryption puissante qui fournit une protection persistante ;
- il bloque le contenu avec une clé de licence, ce qui permet de maintenir la protection de contenu. Chaque licence est associée à un ordinateur. A chaque ouverture de contenu, les plateformes de lecture vérifient si l'ordinateur utilisé dispose de la licence.

[25] News & Event – DRM Digital Rights Management – Content protection – Copyright. 2004-2011. En ligne. www.haihaisoft.com/Eventlist.aspx?CID=2. Consulté le 16 mai 2011.
[26] Haihaisoft, *"Haihaisoft DRM-X 3.0 White Paper"*, 2011, p4.
[27] Haihaisoft, *"Haihaisoft DRM-X 3.0 White Paper"*, 2011, p7.

BD+ (2005)

« Le système DRM BD+ a été développé par le *Cryptography Research Inc.* (CRI), il se base sur son concept de *Self-Protecting Digital Content* (SPDC) »[28]. « Son objectif était d'empêcher les copies non autorisées de disques Blu-Ray et la lecture de ceux-ci sur des périphériques non autorisés.[29] » [Notre traduction]

Selon Wikipedia[30], le système BD+ est une machine virtuelle qui est intégrée dans les lecteurs autorisés et qui permet aux fournisseurs de contenus d'inclure des programmes exécutables sur les disques Blu-Ray.

Selon Dell[31], chaque lecteur doté du DRM BD+ est associé à des clés de sécurité et à un certificat. La vérification de sécurité faite par le lecteur permet de faire correspondre la clé de sécurité du lecteur à son certificat. Cela permet de s'assurer de l'authenticité de la clé. De plus, selon Dell[32], les lecteurs sont fournis avec une empreinte mémoire qui permet à la machine virtuelle d'identifier leur environnement de lecture et de confirmer l'intégrité de l'environnement de protection de contenu. Une fois ces vérifications réalisées, la lecture peut commencer. Lors de la lecture, la machine virtuelle extrait le code spécifique du contenu du disque ; cela permet d'identifier les sections du contenu qui ont été brouillées lors du packaging et de les décoder.

[28] BD+ - Wikipedia. 2010. En ligne. BD+. http://fr.wikipedia.org/wiki/BD%2B. Consulté le 16 mai 2011.
[29] BD+ - Wikipedia. 2010. En ligne. BD+. http://en.wikipedia.org/wiki/BD%2B. Consulté le 16 mai 2011.
[30] BD+ - Wikipedia. 2010. En ligne. BD+. http://en.wikipedia.org/wiki/BD%2B. Consulté le 16 mai 2011.
[31] Dell, *"White Paper – Blu-Ray Disc next generation optical storage: protecting content on the BD-ROM."* 2006. p4-5
[32] Dell, *"White Paper – Blu-Ray Disc next generation optical storage: protecting content on the BD-ROM."* 2006. p5