Déléguer la protection des données personnelles à des agents cognitifs *

Guillaume Piolle [†] and Yves Demazeau [‡]

2010

Résumé

L'intégration des réglementations en matière de protection des données personnelles peut s'avérer complexe pour l'utilisateur. Malheureusement, les propositions techniques en matière de vie privée manquent à la fois d'un lien efficace avec les réglementations et de réelles capacités à aider les utilisateurs au cours de leurs interactions. C'est pourquoi nous proposons le modèle d'agent PAw, un agent cognitif autonome chargé d'assister son propriétaire humain et de protéger ses données. Il est capable d'appréhender un contexte normatif composite et d'en déduire une politique de gestion des données respectant les réglementations. Il peut ainsi procéder aux choix les plus judicieux pour protéger le droit à la vie privée de son propriétaire.

Integrating regulations regarding personal data protection can be a complex task for users. Unfortunately, technical privacy-enhancing proposals lack both an efficient bond with regulations and the actual ability to help human users dealing with their interactions. This is why we propose the model of the PAw agent, a cognitive autonomous agent whose aim is to assist its human owner and to protect her data. It is capable of perceiving and reasoning on a composite normative context, and to build upon this basis a data usage policy focusing on regulation compliance. It is thus able to make the wisest choices in order to protect the right to privacy of its owner.

Mots-clés vie privée, données personnelles, normes, logique déontique

Keywords *privacy, personal data, norms, deontic logic*

1 Introduction

La protection de la vie privée, et plus particulièrement la protection des données personnelles, est une thématique sociétale qui reçoit de plus en plus d'échos dans les média et dont le public se soucie de plus en plus. Elle présente, de par sa nature, des intrications fortes avec tous les domaines de l'informatique. C'est en effet la capacité à stocker, à traiter et à partager des informations rapidement et facilement qui favorise l'apparition de nouvelles dimensions au problème de la vie privée. Ces risques sont encore accentués par les nouveaux usages, fortement collaboratifs, de l'outil informatique. Mais si l'informatique est source de risque en matière de protection des données personnelles, elle fournit également des moyens techniques pour s'en préserver, voire même pour résoudre des problèmes de vie privée qui n'étaient pas dus à l'informatique mais qui ne peuvent être résolus sans elle.

*Ce document est une version « auteur » de l'article [41].

[†]INRIA Grenoble Rhône-Alpes, 655 avenue de l'Europe, Montbonnot, F-38334 Saint-Ismier Cedex – guillaume.piolle@inria.fr

[‡]CNRS, Laboratoire d'Informatique de Grenoble, Maison Jean Kuntzmann, 110 avenue de la Chimie, Domaine Universitaire, F-38400 Saint-Martin-d'Hères – yves.demazeau@imag.fr

Si les propositions concernant les outils techniques sont nombreuses [24], il subsiste certains besoins d'intégration spécifiques afin de pouvoir fournir à un utilisateur humain une solution complète et satisfaisante. Tout d'abord, il semble nécessaire de pouvoir adapter le choix ou le fonctionnement des outils aux diverses réglementations qui régissent le domaine. En matière de protection des données personnelles, les réglementations s'orientent suivant six axes thématiques : l'information de l'utilisateur, son consentement, son droit d'accès et de modification, la justification de la collecte et du traitement, la conservation des données et leur transmission à des tiers [42]. Ces six dimensions devraient idéalement être directement appréhendables et manipulables par les outils. De plus, les sources de telles réglementations sont multiples. L'intégration des exigences des lois nationales, des directives communautaires, des clauses contractuelles, des réglements institutionnels ou des exigences particulières d'un utilisateur est un problème délicat et une assistance automatique semble la bienvenue, si elle s'avère possible. Enfin, le choix des outils techniques, leur manipulation, les spécificités des procédures de traitement mises en place par l'utilisateur pour assurer au mieux la protection de ses données doivent être assujetties automatiquement à ces contraintes réglementaires, à ce « contexte normatif ».

Nous proposons ici des éléments de réponse à ce besoin en fournissant à l'utilisateur humain un assistant logiciel personnalisé pour l'aider à gérer ses données personnelles de manière à protéger sa vie privée. Si l'on souhaite faciliter au maximum l'expérience de l'utilisateur, le déchargeant des aspects technologiques et sécuritaires pour lui permettre de se focaliser sur le cœur applicatif de son interaction avec le système informatique, l'assistant dont on se propose de le doter doit nécessairement être doté de capacités de raisonnement et de décision efficaces. C'est pourquoi il nous paraît légitime de le représenter sous la forme d'un agent cognitif [27, 46]. En effet, ce composant logiciel doit pouvoir manipuler une représentation des contraintes issues du contexte normatif. Comme nous l'avons suggéré, les autorités constituant ce contexte sont multiples, et dans le cas général indépendantes les unes des autres. Par conséquent, l'agent devra être capable de déceler et de résoudre d'éventuelles incompatibilités entre ces contraintes, afin de bâtir, de manière dynamique, un modèle de fonctionnement cohérent. Cette fonctionnalité implique nécessairement une prise de décision, qui amène à considérer l'assistant comme une entité dotée d'autonomie. Le fonctionnement de l'assistant doit toutefois rester soumis dans ses principes directeurs à la volonté de son propriétaire humain, condition nécessaire pour engager la responsabilité de ce dernier [32].

De par les exigences fonctionnelles complexes d'un assistant chargé de protéger les droits de l'utilisateur humain en matière de vie privée, la solution de recourir à un agent cognitif autonome paraît la plus naturelle. Partant de cette perspective, nous présentons ici le modèle d'un tel agent, le *Privacy-Aware agent*, ou agent PAw, conçu comme un « agent utilisateur », un assistant personnel logiciel doté des particularités des agents autonomes.

Dans une première section, nous passons en revue les concepts et outils utiles à la conception d'un tel agent, sans revenir sur les moyens techniques utilisables pour assurer diverses propriétés utiles à la protection de la vie privée dans le cadre d'une application distribuée. Nous détaillons ensuite les fonctionnalités et l'architecture de l'agent PAw (section 3) avant de détailler plus avant la couche de raisonnement normatif qui fait sa principale particularité (section 4). Nous concluons sur des perspectives d'intégration de l'agent PAw à des environnements d'exécution réalistes.

2 Outils existants

La proposition de l'agent PAw s'appuie sur des développements déjà existants, de plusieurs ordres. Du point de vue des outils conceptuels, la notion d'agent utilisateur découle des réflexions récentes de la communauté agent sur le centrage utilisateur. Des outils plus formels, d'autre part, ont déjà été mis au point en ce qui concerne le raisonnement logique, et notamment en ce qui concerne les notions de temps et d'obligation, nécessaires à la compréhension des « normes » en matière de protection des données personnelles.

2.1 Systèmes multi-agents centrés utilisateurs

La recherche sur les systèmes multi-agents a évolué au cours des dernières années pour mettre un accent particulier sur le rôle de l'utilisateur dans la compréhension et la conception du système. La notion, essentiellement applicative, de systèmes multi-agents centrés utilisateurs [22] désigne notamment les architectures dans lesquelles les utilisateurs humains développent une interaction privilégiée avec certains agents artificiels du système, qui leur servent d'interface. Ce type de relation a parfois été modélisé à l'aide de systèmes multi-agents hybrides, comportant des agents artificiels et des agents humains considérés au même niveau d'abstraction [14]. Parmi les travaux traitant de la place de l'utilisateur humain dans le système multi-agent, on peut notamment citer l'analyse d'Yvon Haradji, Nils Ferrand et Haijiang Li [29] ou encore, sur des aspects plus techniques, les architectures multi-agents centrées sur l'utilisateur proposées par Stéphanie Riché, Gavin Brebner et Mickey Gittler [44].

Dans le cadre de ces travaux, le terme de « centrage utilisateur » est généralement entendu comme un parti pris de concevoir et de considérer un système, ses fonctionnalités et ses caractéristiques en considérant comme contraintes prioritaires les seuls bénéfices, risques et diverses répercussions sur un utilisateur humain de ce système. Le point de vue de cet utilisateur prévaut donc, dans cette conception, sur celui des autres intervenants. Dans le cadre d'un agent utilisateur chargé de traiter et de protéger des données personnelles au sein d'un système multi-agent, cela signifie entre autres que la protection des données appartenant à l'utilisateur humain propriétaire de l'agent est considérée comme plus importante que les performances d'une application ou que l'utilité (notamment stratégique ou commerciale) perçue par d'autres agents.

Le centrage utilisateur tel que nous l'entendons ici constitue donc un parti pris très fort, qui nous fait observer un système depuis un unique point de vue (qui peut ne pas être celui de tous les acteurs du système). Les intérêts de ce choix sont principalement de deux ordres. D'une part, il permet d'aborder de manière poussée des questions de libertés individuelles qui entrent en conflit avec les intérêts de la majorité des agents d'un système et qui seraient donc menacées d'être mises de côté dans le cadre d'une approche plus globale. D'autre part, il permet en quelque sorte de simuler l'égoïsme (au sens de la prévalence des buts individuels sur les intérêts collectifs) d'un agent du système, qui ne peut raisonner efficacement qu'en se fondant sur ses informations et intérêts propres et/ou en considérant que les intentions des autres agents sont potentiellement dangereuses pour lui.

Bien entendu, c'est également un point de vue partial et partiel qui reste insuffisant pour la construction d'un modèle global. Il fait écho à d'autres points de vue défendant d'autres agents et d'autres intérêts, comme la personnalisation du service fourni [7] ou la sécurité logicielle des agents [10]. Néanmoins les développements sur ces autres points de vue nous semblent pour l'instant largement plus présents dans le débat scientifique, technique et social que les analyses détaillées du centrage utilisateur tel que nous l'entendons.

2.2 Normes et systèmes normatifs

Dans le domaine de l'intelligence artificielle, le besoin de représentation et de raisonnement sur les contraintes ou les obligations, institutionnelles ou autres, a donné lieu à un certain nombre de développements théoriques. Un agent chargé de protéger les données personnelles, quel que soit son modèle, doit faire partie d'un système au sein duquel son comportement pourra être mis en regard des réglementations. Le champ scientifique traitant des **systèmes normatifs** [30, 34] étudie plus particulièrement cette correspondance entre les comportements souhaités et les comportements réels.

En protection des données personnelles comme dans d'autres domaines, les contraintes qui régissent les processus peuvent être considérées comme une pression sociale exercée sur les individus afin de les amener à un comportement ou un état conforme à une règle. Les prescriptions sociales qui s'appliquent alors aux agents sont appelées **normes**. Le dictionnaire de l'Académie française définit le terme comme un « type, état, comportement qui peut être pris pour référence ». Ce concept générique a été affiné, notamment par Raimo Tuomela (1995), qui caractérise quatre grandes familles de normes, à savoir les r-normes (ou règles), les s-normes

(normes sociales), les m-normes (normes morales) et les p-normes (normes de prudence). Celle qui s'applique le plus naturellement aux applications informatiques et à notre cadre applicatif en particulier est la classe des « r-normes », qui sont les règles qui s'imposent aux agents parce qu'il y a un accord entre eux pour les respecter. Cette classe englobe notamment les textes institutionnels et les obligations contractuelles.

À partir de la notion de norme, on peut dériver le concept de systèmes normatifs et plus particulièrement de **systèmes multi-agents normatifs** [6], systèmes sociaux présentant une interaction riche entre des ensembles de normes et des agents autonomes disposant de plus ou moins de liberté vis-à-vis de ces normes. L'étude de ces systèmes observe notamment la capacité des agents à violer ces normes (par erreur ou par choix stratégique) et la possibilité pour une autre entité (souvent simplement appelée « le système normatif ») d'observer ces violations et d'y réagir, en particulier en appliquant des sanctions. Dans le cadre spécifique de la protection des données personnelles, le concept même de cette autorité de contrôle semble délicat à manipuler, et ce pour deux raisons.

Tout d'abord, dans une application distribuée ouverte (comme l'interaction d'agents hétérogènes via Internet, un scénario des plus communs), les normes considérées par chacun des agents pourront être différentes (notamment pour des raisons géographiques, d'appartenance institutionnelle ou de préférences spécifiques des utilisateurs), et l'évaluation des violations peut difficilement être réalisée de manière uniforme par une autorité partagée. En effet, des informations comme l'importance relative revêtue par les différentes sources de normes, ou bien le détail des préférences de l'utilisateur (qui constituent également des normes) ne sont pas publiquement disponibles, et il serait difficile à un observateur extérieur de modéliser le raisonnement normatif d'un agent distant. De plus, les informations nécessaires à ce raisonnement de contrôle (normes et autorités respectées par l'agent, préférences de l'utilisateur) peuvent être considérées comme privées par l'agent. Quand bien même ces données seraient volontairement fournies à l'autorité de contrôle, celle-ci devrait rendre des évaluations personnalisées pour chaque agent, et ainsi la caractérisation des violations ne pourrait pas être partagée. Le problème aurait été déplacé.

En outre, la plupart des violations de normes sur la protection des données personnelles peut souvent s'effectuer de manière transparente pour les autres agents, qui ne pourront la détecter. Par exemple, si un agent conserve une donnée personnelle alors qu'en application des normes il aurait dû la détruire, et si cet agent ne commet pas « d'erreur » trahissant ce manquement, seule la connaissance de son état interne (information généralement non disponible aux autres agents interagissant avec lui) permettrait d'affirmer que la violation a bien eu lieu.

En conséquence, seule une autorité omnisciente pourrait détecter et sanctionner efficacement les violations des normes relatives à la protection des données personnelles, et une telle autorité ne saurait exister dans le cas d'une application complètement distribuée et ouverte. En effet, dans le cas d'une transaction de commerce électronique, cette autorité devrait pouvoir analyser l'état interne des agents clients comme des agents de service (les vendeurs), tout en ayant autorité pour sanctionner les uns et les autres. Cela pose des problèmes à la fois d'ordre technologique (l'analyse est délicate si le système est ouvert à tous types d'agents, et ces derniers devraient toujours s'exécuter dans un environnement contrôlé), pratique (dans le cadre d'une transaction entre des agents de plusieurs pays, comment s'accorder sur la localisation de l'autorité de contrôle ?) et philosophique (les différentes parties se soumettraient en partie à une entité de contrôle centralisée dont les capacités lui permettraient des atteintes sévères à la vie privée, à la propriété intellectuelle, à la libre concurrence...). Il paraît donc illusoire de prendre comme hypothèse que la protection des données personnelles par des agents utilisateurs autonomes peut s'appuyer sur une autorité de contrôle centralisée considérée a priori comme un tiers de confiance. Ces considérations sur les limitations de la notion de système normatif dans le cadre de la protection des données personnelles pourront être retrouvées en détail dans [18].

Pour ces raisons, il nous semble que si le concept de système normatif est essentiel à la réalisation de nos objectifs, la manière la plus réaliste (sinon la plus efficace) de l'appréhender est de considérer que l'autorité de contrôle et de sanction est complètement délocalisée dans les agents eux-mêmes, seuls à même, dans le cas général, de détecter et de pallier leurs propres manquements. Cela traduit un certain parti pris : considérant qu'il n'est pas raisonnable (dans

un cadre distribué et ouvert) de recourir à une autorité de contrôle, les agents ne pourront s'en référer qu'à leur propre évaluation de la situation. Ils devront ainsi s'assurer par eux-mêmes que leur propre comportement est conforme, et évaluer dans la mesure de leurs moyens la conformité du comportement des autres agents, sans pouvoir recourir à des sanctions autres que le refus d'interagir avec des agents soupçonnés de violer les normes. Il en résulte bien évidemment que dans ces cas d'espèces particulièrement défavorables, il est impossible de recourir au système normatif lui-même pour assurer le respect des propriétés fonctionnelles requises par les normes. Enfin, le besoin se fait d'autant plus pressant de doter les agents individuels de capacités de raisonnement leur permettant de raisonner sur les normes qui pèsent sur eux et sur leurs interlocuteurs.

2.3 Outils formels

La formalisation de la notion de système normatif et des concepts associés dans le cadre des systèmes multi-agents, et plus particulièrement des agents cognitifs, s'appuie sur des outils de nature logique et a donné lieu à un certain nombre de développements dont certains semblent particulièrement adaptés à la manipulation des réglementations en matière de protection des données personnelles.

2.3.1 Formalisation déontique

Le modèle cognitif BDI de Cohen et Levesque [17], comme nombre de ses extensions et d'autres contributions dans le domaine des agents cognitifs, sont exprimés dans le formalisme de la logique modale [13, 5]. Il paraît souvent naturel d'étendre les fonctionnalités de raisonnement d'un agent cognitif avec de nouvelles modalités ou un nouveau langage modal, pour une articulation plus aisée avec des propositions complémentaires. Lorsque l'objectif est la manipulation de la notion de norme, on introduit dans le modèle de l'agent des modalités de nature déontique (traitant des notions d'obligation, de permission et d'interdiction) et éventuellement temporelle, qui lui permettront de se représenter le caractère plus ou moins désirable des actions, des états, voire de scénarios complexes.

La plupart des représentations formelles de la notion de norme s'appuient sur la logique déontique standard (ou SDL pour *Standard Deontic Logic*), une logique modale normale où la modalité universelle Ob représente la notion d'obligation ou de devoir. Elle permet également de construire les modalités de permission (Per) ou d'interdiction (For). Elle a été introduite par Georg Henrik von Wright en 1951 [45].

2.3.2 Politiques de sécurité

Malgré les nombreux paradoxes dont souffre la logique déontique standard [33], celle-ci reste un outil privilégié à cause de sa correspondance avec les concepts philosophiques de l'obligation et du caractère intuitif et aisément représentable de sa sémantique. Toutefois, les handicaps formels la confinent souvent à un rôle de représentation et de formalisation, notamment dans le cadre des travaux sur les politiques de sécurité. SDL ou l'une de ses variantes sont en effet souvent utilisées pour représenter ces politiques, plus que pour raisonner conjointement sur les politiques et l'état réel du monde. Dans le contexte de l'agent PAw, nous avons tout intérêt à élaborer un formalisme unique pour traiter à la fois des ensembles de normes qui constituent le contexte de l'exécution de l'agent, et sa politique de sécurité interne, cohérente et directement applicable, qui constitue le résultat de son raisonnement normatif.

De nombreux auteurs [12, 19, 20, 28, 30, 35, 36, 15, 16] ont étudié les apports et les limitations de la logique déontique pour le travail sur les politiques de sécurité. L'intérêt initial et fondamental est d'apporter un formalisme mathématique dans un domaine originellement régi par des réglementations en langage naturel, donc potentiellement informelles. Les spécifications même d'une politique de sécurité peuvent ainsi être manipulées grâce à des outils mathématiques. Cette démarche de limitation des ambiguïtés du langage ne semble pas être en soi un argument donnant un avantage particulier à la logique déontique par rapport à d'autres formalismes logiques. Par contre, Rodolphe Ortalo remarque que la facilité avec laquelle les ensembles de formules

déontiques peuvent être conceptualisés en s'appuyant sur la sémantique relationnelle, les diverses représentations qui peuvent en être faites, sont apparues comme un outil de communication formidable entre les différents acteurs de la sécurité d'un système [36]. Selon Ortalo toujours, le fait d'exprimer les politiques de sécurité en logique déontique permet une grande flexibilité dans la conception du système lui-même.

Dans le cadre de la protection des données personnelles, l'ensemble des normes applicables dans un contexte donné peut être représenté de manière formelle et considéré comme une politique de sécurité pour un agent ou pour une organisation. La logique déontique peut alors être utilisée dans ce rôle de représentation, indépendamment des possibles applications de ladite politique. Néanmoins, un ensemble de normes contraignant le comportement des agents ne pourra constituer une politique acceptable que s'il est cohérent.

Il a par ailleurs été souvent remarqué que l'utilisation de la logique déontique pour la gestion des politiques nécessitait une meilleure adaptabilité du formalisme, afin de représenter l'évolution de ces politiques au cours de l'exécution du système. L'introduction de logiques temporelles dans ce formalisme est une possibilité qui a été étudiée par de nombreux auteurs, et qui nous paraît essentielle. Le second principal reproche que l'on fait à la trop simpliste logique déontique est son incapacité à résoudre nativement les dilemmes, quand bien même on pourrait les représenter par différenciation des modalités, pour arriver à un ensemble de normes cohérent pouvant constituer une politique de sécurité. Nous verrons par la suite comment l'introduction de logiques temporelles d'une part, et la mise en place de procédures de fusion d'autre part, ont été proposées pour répondre à ces deux défaillances identifiées de la logique déontique.

2.3.3 Logique temporelle linéaire

Les normes portant sur la protection des données personnelles traitent souvent de concepts liés au temps. Les exemples les plus simples en sont sans doute les limitations sur la durée de rétention, qui imposent que telle ou telle donnée soit supprimée dans un temps imparti. Il est donc nécessaire pour l'agent PAw de pouvoir raisonner sur les concepts temporels afin de pouvoir appréhender ce type de norme avec efficacité.

Les logiques temporelles constituent une famille de logiques modales dans lesquelles les modalités représentent des notions liées au passé, au futur ou à la précédence temporelle d'événements. Dans ces logiques, les mondes représentent des instants reliés par une relation d'accessibilité définissant le sens d'écoulement du temps. Dans les logiques temporelles linéaires (LTL) [43], les instants sont ordonnés suivant une unique chaîne qui va du passé vers le futur.

Les modalités temporelles universelles les plus courantes sont H et G. $H\varphi$ signifie que φ a toujours été vraie dans le passé, $G\varphi$ que φ sera toujours vraie dans le futur (présent exclu dans les deux cas). On peut leur associer les modalités existentielles P et F, respectivement. La modalité X, quant à elle, désigne l'instant successeur ($X\varphi$ signifie que φ sera vraie à l'instant suivant). On peut lui associer une modalité converse X^{-1}. Toutes ces modalités peuvent éventuellement être déduites de deux modalités dyadiques plus expressives, les versions strictes de (U) ($\ll$ *Until* $\gg$) et (S) ($\ll$ *Since* $\gg$), $\varphi \mathcal{U} \psi$ signifiant que φ sera vraie dans le futur jusqu'à ce que ψ survienne, et $\varphi \mathcal{S} \psi$ signifiant que φ a été vraie dans le passé depuis la dernière survenance de ψ. Les axiomatiques et sémantiques détaillées de ces opérateurs pourront être retrouvées dans [5].

2.3.4 Logiques déontiques et temporelles

La notion d'obligation ou d'interdiction est souvent associée à des contraintes temporelles. Dans le cadre de la protection des données personnelles, ce besoin d'associer les obligations à des notions de dates, de délais ou de précédence se fait particulièrement sentir. Si l'on prend l'exemple de la rétention de données personnelles, il faut généralement exprimer une obligation (de supprimer cette information) associée à une échéance. Dans le cas de l'information de l'utilisateur, on a une obligation sur la précédence temporelle de plusieurs formules. De nombreuses propositions ont été faites pour associer aux notions de la logique déontique les possibilités offertes par les logiques temporelles, notamment linéaires. Une telle association permet de mieux caractériser la

notion de violation d'une obligation, ou encore d'exprimer des obligations comportant des durées de validité ou des dates limites.

En 1980, Brian F. Chellas propose l'introduction de modalités temporelles dans la logique déontique standard [13]. Dans la sémantique associée, les mondes représentent des exécutions complètes du système, depuis sa création. Chellas introduit le concept de compatibilité historique à un instant donné, qui dit que deux exécutions ont des histoires passées indiscernables l'une de l'autre, et conditionne la construction d'une relation d'accessibilité déontique à cette notion. Le système de Chellas nous paraît très limité aujourd'hui. En effet, il ne prend pas en compte les développements alors récents de la logique LTL et s'interdit un raisonnement explicite sur le futur et le passé. D'autre part, dans le système proposé par Chellas, il est impossible à un agent d'avoir des obligations qu'il ne peut respecter, ce qui limite l'expressivité normative (les conflits et inconsistances pouvant toujours survenir).

La *Normative Temporal Logic* (NTL) est un exemple de logique temporelle et déontique moderne, proposée en 2007 [1]. La logique NTL n'est pas fondée sur la logique temporelle linéaire, mais sur la logique temporelle arborescente (logique CTL). Les modalités temporelles utilisées dans NTL s'entendent donc par défaut pour l'ensemble des futurs possibles. Par ailleurs, l'opérateur d'obligation de la logique NTL est contextualisé en fonction du système normatif, plusieurs d'entre eux pouvant être considérés simultanément. Cette logique présente donc l'intérêt de représenter les dilemmes, à condition qu'ils soient répartis sur plusieurs systèmes normatifs. C'est un concept qui s'avère particulièrement pertinent lorsqu'il s'agit de raisonner sur un contexte normatif hétérogène comme celui de l'agent PAw. En conséquence, cette logique peut être considérée comme un premier pas vers une gestion des dilemmes entre systèmes normatifs, en permettant de les représenter mais pas de les arbitrer. Cette proposition soulève la question de l'utilisation d'une logique du temps arborescente. Il faut alors garder à l'esprit que ce sont des normes que nous souhaitons représenter, et que ces normes traduisent des réglementations qui disent ce qui doit être fait ou pas. Une représentation arborescente du temps permettrait d'analyser le comportement possible d'un agent en décrivant ses futurs possibles (certains respectant une norme, d'autres la violant), mais la norme en elle-même ne traite que de l'histoire effective, celle qui a eu et qui aura lieu, à l'exclusion de toute autre. Le fait que l'agent raisonne prioritairement sur ses obligations nous permet de nous limiter à un temps linéaire, plutôt que d'utiliser un formalisme plus adapté à la description et à l'analyse externe d'un système.

Lennart Åqvist présente en 2004 une étude détaillée des méthodes usitées pour introduire le temps dans une logique déontique [3]. Il distingue notamment les approches utilisant une indexation par le temps, à la manière de Chellas par exemple [13], des approches utilisant les opérateurs temporels comme celle utilisée plus tard par Ågotnes *et al* [1]. Åqvist caractérise les logiques modales multidimensionnelles, s'interprétant sur une structure de coordonnées à deux dimensions : les instants et les mondes (ou histoires). Afin de lier langage et sémantique et d'utiliser dans les formules les notions d'instant et de monde, Åqvist utilise dans le formalisme qu'il met en place des *constantes systématiques de cadre* représentant l'une ou l'autre coordonnée. De la proposition d'Åqvist nous retiendrons principalement le raisonnement bi-dimensionnel entre axes déontique et temporel, ainsi que la notion d'instant plus aisément manipulable que les exécutions de Chellas. Les principes introduits ici nous permettront de raisonner sur des dates (de manière absolue ou relative), concept récurrent dans le domaine de la protection des données personnelles.

2.3.5 Obligations avec échéances

On peut considérer qu'exprimer des obligations en logique déontique reste vain si l'on n'attache pas des contraintes temporelles à chaque obligation. Pour illustrer cette idée, prenons l'exemple d'un agent qui aurait l'obligation d'effacer une certaine donnée personnelle, mais sans contrainte de temps. Il y a deux manières de considérer cette situation : ou bien c'est une obligation immédiate et l'agent est en violation s'il n'obéit pas à l'instant présent (ce qui est un point de vue rarement tenable), ou bien c'est une obligation de le faire « un jour ». Dans ce second cas, l'agent ne se trouve jamais en situation de violation, car il peut toujours prétendre qu'il

prévoit de détruire les données à une date future. Ainsi, son comportement ne peut jamais lui être reproché. Ce problème est d'autant plus présent en protection des données personnelles que les réglementations font systématiquement référence à des contraintes de temps.

Pour remédier à ce problème et représenter les obligations de manière plus réaliste, plusieurs auteurs ont fait des propositions d'opérateurs mixtes, déontiques et temporels, d'obligations avec échéances [25, 23, 9, 21]. Toutes partent d'une même idée fondamentale : il faut utiliser l'opérateur temporel $\mathcal{U}$ (ou un équivalent) de manière à maintenir une obligation sur une formule temporelle jusqu'à ce qu'une échéance arrive ou que l'obligation soit respectée. Si l'obligation n'est toujours pas respectée à l'échéance, on considère alors qu'il y a une violation.

Frank Dignum, Jan Broersen, Virginia Dignum et John-Jules Meyer proposent un modèle d'obligation avec échéances qui n'est pas fondé sur une logique déontique [25]. En effet, la notion d'obligation avec échéance y est définie parallèlement avec celle de violation, ces deux concepts étant liés par des contraintes temporelles pouvant être exprimées en LTL. En conséquence, une obligation avec échéance peut être dérivée dès que les conditions sont réunies pour qu'elle soit respectée, sans qu'il y ait eu une quelconque initiative d'une quelconque autorité pour exprimer une obligation. Cet exemple illustre la nécessité de recourir à un opérateur déontique primitif dans le formalisme. Robert Demolombe propose une logique de type *stit* (pour *See To It That*, faire en sorte que), capable de représenter les obligations et utilisant des modalités temporelles classiques [23]. En transposant l'opérateur de Demolombe dans une logique déontique et temporelle, on constate qu'il ne souffre pas du même problème d'apparition « automatique » des obligations que celui de Dignum *et al.* Julien Brunel, Jean-Paul Bodeveix et Mamoun Filali étendent quant à eux le produit de SDL et de LTL, en utilisant des opérateurs indexés par des durées (plutôt que par des dates dans les autres formalismes) [9]. Ce formalisme permet de raisonner explicitement sur des durées, mais nécessite l'introduction dans le langage de quantificateurs explicites sur ces durées. L'opérateur d'obligation avec échéance correspondant ne peut donc être exprimé directement dans une logique déontique et temporelle classique. D'autres propositions, comme celle de Frédéric Cuppens, Nora Cuppens-Boulahia et Thierry Sans [21] poursuivent un but légèrement différent dans la conception de l'obligation avec échéance. Dans ce cadre, le langage logique est destiné à la spécification de politiques de sécurité, mais pas au raisonnement cognitif des agents qui y sont soumis. Les exigences formelles en sont donc moindres, elles se limitent à permettre la définition claire et distincte de la norme et l'évaluation de son respect ou de sa violation. Cette dernière proposition est construite dans une logique déontique, aléthique, temporelle et dynamique, prenant notamment en compte la durée des actions.

Afin de traiter efficacement les obligations avec échéances dans le cadre de protection des données personnelles, nous devrons, lorsque nous disposerons d'un formalisme logique, déterminer quelles sont les caractéristiques qu'un tel opérateur doit avoir dans notre contexte.

2.3.6 Gestion des conflits normatifs

De nombreux travaux de recherche se sont intéressés à la notion de conflit en logique déontique, que ce soit pour les éviter ou les résoudre. Dans une logique mêlant nécessité, obligation, temps et action [4], Philippe Balbiani aborde en 2005 le problème des obligations émises par des autorités indépendantes et propose de forcer l'absence de conflits entre celles-ci grâce à un axiome (formule (1)) liant les différentes modalités déontiques associées. Cette formule dit que l'existence d'obligations implique la permission par toute autorité de la conjonction de toutes les formules sur lesquelles portent des obligations. L'équivalent sémantique de cet axiome est la contrainte suivant laquelle l'intersection de toutes les relations d'accessibilité déontiques doit être sérielle.

$$\bigwedge_{i=1}^{n} Ob_i\ \varphi_i \rightarrow Per_{n+1} \bigwedge_{i=1}^{n} \varphi_i \tag{1}$$

On trouve dans cet axiome un outil essentiel pour la nécessaire gestion des conflits, qui peuvent rendre un ensemble de normes incohérent. Nous pourrons l'utiliser dans une logique déontique multimodale pour stigmatiser la présence ou l'absence de conflits entre différentes sources de normes. Il est utile de noter que c'est davantage un schéma axiomatique qu'un axiome, et que ses

différentes instances en rendent la structure symétrique. Ainsi, il n'y a pas une autorité $n+1$ se distinguant particulièrement des autres : s'il y a $n+1$ autorités, il y a $n+1$ instances de l'axiome, avec à chaque fois une autorité différente en partie droite.

Dans une série de propositions successives, Laurence Cholvy et Frédéric Cuppens se sont attelés à la résolution de tels conflits. Dans [15], cette résolution se fonde sur une fusion des différents rôles joués par un agent, alors que dans [16] les conflits surviennent lorsque plusieurs sources de normes entrent en conflit, ce qui correspond mieux au cas du contexte d'exécution de l'agent PAw. Comme dans la plupart des travaux similaires [31], la notion de conflit correspond à l'inconsistance d'un système SDL, et sa résolution s'opère en fusionnant des ensembles de normes conflictuels entre eux pour aboutir à un ensemble non conflictuel, en désactivant un certain nombre de formules au profit des autres. Nous retiendrons le principe de cette méthode de « fusion de normes avec préférences » (correspondant à la recherche d'un sous-ensemble maximal consistant), la relation de préférence convenant à l'agent PAw restant à identifier de manière claire.

3 L'agent *PAw*, un agent conscient de la protection des données personnelles

Les propositions en matière de protection des données personnelles mettent en exergue un certain nombre de principes et de bonnes pratiques, mais il nous semble que certains aspects demandent à être traités de manière plus spécifique. En effet, les démarches d'automatisation de la gestion sont assez rares et encore embryonnaires, alors que les travaux en matière de personnalisation et d'organisation des applications distribuées autour de l'utilisateur en montrent le besoin, comme le pointe par exemple Alfred Kobsa (2007). En réponse à ce constat, nous proposons une contribution à cet état de l'art par la conception d'outils formels et techniques pour un agent artificiel assistant (l'agent PAw, ou *Privacy-Aware Agent*), lié à son agent humain utilisateur et chargé d'interfacer ses interactions avec des applications de service tout en prenant en charge la protection de ses données personnelles. L'agent PAw est un agent cognitif conscient de son contexte normatif en matière de protection des données personnelles. L'objectif fondamental que nous poursuivons dans la conception de cet agent est de lui permettre de gérer les données personnelles qui lui sont confiées de la meilleure manière possible, en regard des diverses réglementations qui régissent son activité. Utilisant, dans sa mise en œuvre technique actuelle, le modèle d'agent BDI procédural JACK [2] (ce choix n'étant pas essentiel en soi à la réalisation du modèle proposé), l'agent PAw inclut des modules spécialisés mettant en œuvre des fonctionnalités de raisonnement logique sur les réglementations à respecter, ainsi que des outils de représentation et de raisonnement sur les outils techniques à mettre en œuvre pour cela.

Nous présentons tout d'abord les fonctionnalités que nous jugeons nécessaires, au vu de nos études et observations, pour aboutir à un agent personnel gérant correctement les données personnelles, puis nous proposons une intégration de ces fonctionnalités dans un modèle d'agent.

3.1 Les composantes de la gestion PAw

Afin que l'agent PAw puisse manipuler les données qui lui sont confiées dans le respect du cadre légal et réglementaire, il doit notamment être capable de représenter explicitement les données personnelles (appartenant à son propriétaire ou à un autre utilisateur) ainsi que les différentes réglementations qui s'appliquent à celles-ci, afin de pouvoir agir conformément au contexte normatif auquel il est soumis. Nous avons vu que ces réglementations pouvaient provenir de sources hétérogènes (directives européennes, textes de loi, réglementations, contrats, préférences de l'utilisateur) et par conséquent comporter des incohérences. Un agent PAw doit donc être capable de raisonner en dépit de ces incohérences afin de déduire un ensemble de normes cohérent à respecter, servant de politique de sécurité. Il lui faut enfin asservir sa gestion des données personnelles à ces normes, puis faire en sorte d'étendre la protection des données personnelles aux autres agents du système.

3.1.1 Perception du contexte normatif

L'agent PAw dispose tout d'abord d'outils de communication lui permettant de se procurer les réglementations émises par les diverses autorités pouvant influer sur son comportement. Nous faisons ici l'hypothèse que les autorités normatives sont directement désignées à l'agent par son propriétaire. Par le biais de requêtes à ces autorités, l'agent obtient l'expression de ces normes dans un langage formel de nature logique. Nous faisons ce choix afin de permettre à l'agent PAw de bénéficier, pour son raisonnement sur les réglementations, des mécanismes d'inférence de la logique déontique, spécifiquement adaptés à la notion de norme, détaillés plus haut. Nous utilisons ici un formalisme particulier, la logique DLP (pour *Deontic Logic for Privacy*) [39] que nous allons développer de manière plus approfondie dans la section suivante.

L'agent PAw est également capable de récupérer, depuis l'interface web d'un agent de service, une politique P3P [47] pour en extraire des informations dans le formalisme DLP, de manière à pouvoir les mettre en relation avec les normes dont il a connaissance.

3.1.2 Raisonnement sur le contexte normatif

Disposant de ces ensembles de normes multiples, l'agent PAw met en œuvre des procédures particulières pour vérifier certaines propriétés (notamment de syntaxe et de cohérence individuelle) sur les normes reçues et les autorités individuelles qui les ont émises. Puis les divers ensembles de normes sont assemblés pour former un tout cohérent. En effet, les sources de normes dans un environnement ouvert sont multiples, et un ensemble de normes devrait être rendu logiquement consistant afin de constituer une politique de sécurité acceptable (en ceci qu'elle ne se contredit pas elle-même et qu'elle permet à l'agent de prendre des décisions saines). Pour cela, l'agent PAw doit détecter les éventuels conflits qui existent entre ces ensembles et les arbitrer de manière à obtenir au final une collection de normes qui lui permette de diriger son comportement de manière non ambiguë. Par la suite, l'agent devra être capable de décider si une ou plusieurs de ces normes concernent les actions qu'il souhaite mener, en fonction de celles qu'il a déjà effectuées ou observées.

3.1.3 Protection locale des données personnelles

Une fois établie une base de normes cohérente sur la base des réglementations, l'agent PAw doit avoir les moyens de s'assurer que les données personnelles qu'il est amené à traiter n'échappent pas à ces normes. C'est le cœur de notre proposition : l'agent PAw, de par ses spécifications, doit respecter au mieux les réglementations et la politique qu'il en a tirée. Nous supposons ici que l'agent est d'ores et déjà capable de distinguer les informations à caractère personnel des autres données informatiques (l'inférence automatique du caractère personnel d'un ensemble de données dépassant de beaucoup le cadre de notre travail).

Nous choisissons de placer les informations personnelles dans une structure de données à part au sein de l'agent. En effet, ces données doivent être traitées différemment des autres, car les éléments de politique s'y appliquant doivent être au préalable examinés. Cette structure de données doit être conçue de telle manière que l'agent lui-même ne puisse y accéder sans qu'une vérification des normes ait lieu. Une telle organisation permet ainsi d'assurer une protection *locale* des données personnelles, limitée à l'enceinte logicielle de l'agent PAw.

3.1.4 Raisonnement sur les protocoles applicatifs

Les fonctionnalités présentées pour l'instant ne permettent à l'agent PAw que de protéger les données personnelles dans le cadre local de ses propres actions. Or, pour assurer la protection de la vie privée de l'utilisateur, la protection *étendue* des données personnelles est une nécessité lorsque ces données sont partagées au sein d'une application distribuée. Dans ce contexte, il faut donc que l'agent PAw soit capable de garantir que les normes sont respectées non seulement par lui mais également par tous les agents qui seront amenés à manipuler les données dont il a la garde. Dans cet objectif d'amélioration de la protection étendue des données personnelles, nous

dotons l'agent PAw de connaissances factuelles et spécifiques sur divers protocoles et architectures d'applications. Ces connaissances, associées à l'ensemble de normes auquel il se soumet, lui permettent de prendre des décisions éclairées lorsque la nécessité survient de communiquer des informations personnelles à d'autres agents afin qu'elles soient traitées à l'échelle d'une application répartie. L'agent PAw a ainsi par exemple la possibilité de déterminer si une architecture présente davantage de garanties qu'une autre sur le respect des normes ou si la transaction dans son ensemble est inacceptable.

3.2 Architecture de l'agent PAw

Nous choisissons pour la réalisation de l'agent PAw d'utiliser le modèle d'agent JACK [2], qui met en œuvre les principes des agents BDI [8] de manière procédurale, en s'appuyant sur le modèle dMars [26]. Bien que n'importe quel modèle d'agent cognitif suffisamment expressif pourrait a priori lui être substitué, nous justifions ce choix technique par la présence dans la plate-forme de développement de fonctionnalités utiles à la mise en œuvre de la protection locale des données personnelles ainsi qu'au maintien de bases de croyances cohérentes.

Les fonctionnalités de l'agent PAw sont organisées en couches, suivant une architecture guidée par le modèle d'agent sous-tendu par la plate-forme JACK. On dispose nativement d'une couche de **croyances**, dans laquelle sont stockées toutes les informations non strictement statiques nécessaires au fonctionnement de l'agent, une couche de **buts** BDI, qui sont des événements spéciaux auxquels l'agent fait en sorte de répondre d'une manière ou d'une autre, et une couche de **plans** qui constituent les procédures d'action spécifiques de l'agent. La figure 1 représente l'architecture de l'agent PAw, que nous allons détailler plus avant.

3.2.1 Couche de raisonnement normatif

Au-dessus des trois couches préexistantes, nous ajoutons artificiellement une couche de **raisonnement normatif** à laquelle nous confions les fonctionnalités de perception et de raisonnement sur le contexte normatif de l'agent PAw. Cette couche fournit en sortie des ensembles de croyances « normatives » qui alimentent la couche de croyances de l'agent. Nous choisissons ainsi d'isoler les fonctionnalités de raisonnement sur le contexte applicatif que nous venons d'introduire, car elles ont pour objet la production d'une politique qui sera consommée par les autres fonctionnalités de l'agent PAw. La couche de raisonnement normatif fait appel à des mécanismes natifs du modèle d'agent JACK (croyances, buts, plans, communication) que nous représentons à part et qui interagissent avec une machine virtuelle Prolog utilisée pour le raisonnement logique. C'est cette couche de raisonnement qui met en œuvre la logique DLP que nous allons présenter plus loin.

3.2.2 Couche de croyances

Les croyances manipulées par l'agent PAw sont de plusieurs ordres :

- Les **croyances normatives** sont celles qui sont fournies par la couche de raisonnement normatif. Elles sont notamment constituées d'un ensemble de normes cohérent issu de l'arbitrage des conflits (la politique de sécurité applicable par l'agent), de l'ensemble des normes désactivées par ce même mécanisme ainsi que d'autres informations annexes que nous détaillerons par la suite. Nous représentons ces informations dans l'agent PAw sous la forme de croyances à cause de leur caractère dynamique et de la nécessaire interprétation des normes effectuée par l'agent, qui fait de la politique de sécurité calculée une vision du monde qui n'est pas forcément partagée par les autres agents. Nous isolons toutefois ces informations des autres croyances car elles doivent être considérées séparément, et notamment avant chaque manipulation de données à caractère personnel.
- Les **croyances métier** sont les données nécessaires au fonctionnement quotidien de l'agent PAw, les croyances que devrait manipuler de toute manière un agent dans la même situation applicative, même sans s'inquiéter de la protection des données personnelles. Parmi ces croyances, certaines sont identifiées comme ayant un caractère personnel. Ces dernières

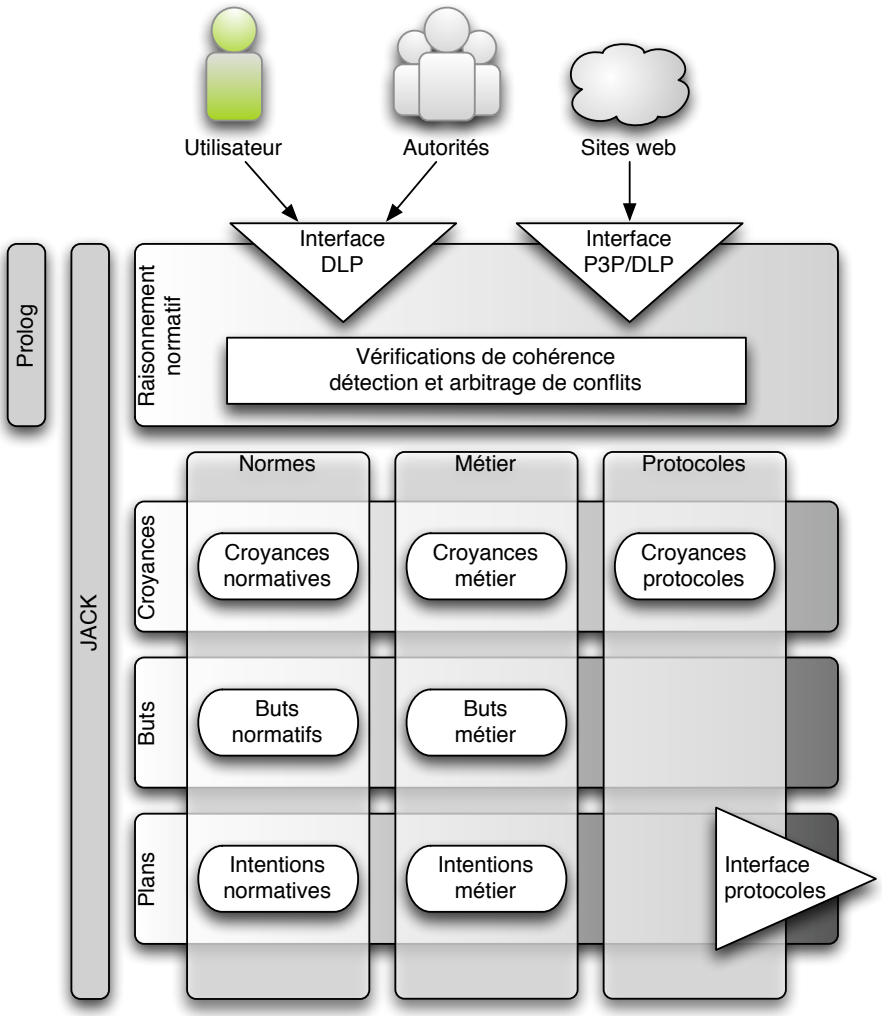

FIGURE 1 – Architecture de l'agent PAw

sont donc isolées et traitées différemment, comme déjà signifié, mais rentrent tout de même par définition dans la catégorie des croyances « métier ».

– Les **croyances protocoles**, statiques car fournies à l'agent à son démarrage, constituent sa base de connaissances relative aux protocoles et architectures d'application qu'il est susceptible d'utiliser pour collaborer avec d'autres agents.

3.2.3 Couche de buts

Les buts de l'agent PAw sont représentés par des événements (messages internes ou externes) d'un type particulier, représentant chacun un objectif à atteindre. L'agent est capable de répondre à ces buts en leur associant un plan. Nous choisissons de diviser les buts de l'agent en deux familles : les **buts métier** qui traitent de ses missions applicatives en tant qu'agent personnel et les **buts normatifs**, engendrés par une analyse des croyances normatives sur la protection des données personnelles. Les buts normatifs sont générés automatiquement par l'agent lui-même lorsqu'il observe, par l'analyse de ses croyances normatives et de ses croyances métier, qu'une obligation devient active. Nous permettons ainsi de distinguer les activités métier de l'agent, qui peuvent être déclenchées par des événements extérieurs, du contrôle normatif qu'il s'impose, ce dernier étant subordonné à des événements essentiellement internes, créés en fonction des

liens entre croyances normatives et exigences applicatives. Les buts normatifs vont influer sur la manière dont l'agent traite les buts métier, en conditionnant les actions menées par les plans associés pour faire en sorte que les normes actives soient respectées.

3.2.4 Couche de plans

Les plans JACK de l'agent représentent, lorsqu'ils sont activés, les intentions d'un agent BDI. C'est la couche la plus basse en termes d'abstraction cognitive, elle décrit les méthodes de fonctionnement concrètes de l'agent. Ici encore les intentions se divisent entre **intentions normatives** et **intentions métier**, les premières contraignant les secondes. Des ensembles de plans spécialisés, les plans « protocoles », sont également en charge de la communication avec l'extérieur, lorsque cette communication met en jeu des données personnelles. Ces plans sont appelés en relation avec les connaissances de l'agent sur la caractérisation des protocoles et les prises de décision qui en découlent, mais en eux-mêmes ils constituent davantage un mécanisme d'interface qu'un concept cognitif.

4 Le composant de raisonnement normatif

La fonctionnalité clé de l'agent PAw est sans doute sa capacité à raisonner directement sur son contexte normatif de manière à influencer son propre mode de fonctionnement. Le composant de raisonnement normatif de l'agent s'appuie sur un langage dédié permettant de représenter les normes et de travailler dessus. Il autorise notamment le raisonnement sur les conflits d'obligations, pour lesquels nous proposons des méthodes de résolution, et est suffisamment expressif pour permettre la conception d'opérateurs déontiques et temporels riches comme les obligations avec échéances ou les interdictions maintenues sur une période.

4.1 Représentation des réglementations

L'agent PAw utilise, pour raisonner sur les normes, une logique déontique et temporelle nommée DLP (pour *Deontic Logic for Privacy*), dont les spécifications complètes pourront être retrouvées dans [37, chapitre 4].

4.1.1 Construction du langage

Cette logique s'appuie sur un langage de prédicat que l'on appelle $\mathcal{L}_{DLP}$ et qui permet de représenter les caractéristiques d'un traitement portant sur des données personnelles. Ces prédicats ont été conçus pour décrire les six axes réglementaires de la protection des données personnelles, à savoir l'information, le consentement, le droit d'accès et de modification, la justification, la conservation et la transmission [42]. La figure 2 liste les prédicats qui, avec les connecteurs de la logique propositionnelle, forment le langage $\mathcal{L}_{DLP}$. Les prédicats **informatifs** représentent les actes de langage de l'agent responsable du traitement (nous l'appelons par la suite « agent de service »), informant l'agent propriétaire des données (l'agent utilisateur) de diverses informations relatives au traitement : sa nature, les éventuels agents destinataires des données, les durées de conservation des données et les informations de contact. Les prédicats **performatifs** représentent les actions essentielles au traitement, hors les actes de langage informatifs : la demande de données, la demande de consentement, le consentement, la transmission des données, la réalisation du traitement, les demandes de mise à jour des données, les mises à jour effectives, les demandes de suppression des données, les suppressions effectives et la transmission de données à des agents tiers. Les prédicats **d'état** sont des prédicats « d'intendance », en cela qu'ils sont destinés à enregistrer certaines informations et à assurer la permanence de leur représentation dans le temps (via des axiomes appropriés). Ils permettent à l'agent PAw de mémoriser les caractéristiques statiques relatives à un traitement : sa nature, le type des données associées, la listes des agents destinataires des données, les durées de conservation des données, le responsable du traitement, les informations de contact obtenues et le propriétaire des données.

$$
\begin{aligned}
\text{PREDICATE} &= \text{INFORMATIVE} \mid \text{PERFORMATIVE} \mid \text{STATE} \ ; \\
\text{INFORMATIVE} &= \text{“}\mathit{informActionType}\text{(”}\ \text{A, A, ID, ACTIONTYPE}\ \text{“)”} \\
&\mid \text{“}\mathit{informForward}\text{(”}\ \text{A, A, ID, DATAID, FW}\ \text{“)”} \\
&\mid \text{“}\mathit{informDuration}\text{(”}\ \text{A, A, ID, DATAID, DURATION}\ \text{“)”} \\
&\mid \text{“}\mathit{informContact}\text{(”}\ \text{A, A, ID, A}\ \text{“)”}\ ; \\
\text{PERFORMATIVE} &= \text{“}\mathit{request}\text{(”}\ \text{A, A, ID, DATATYPE, DATAID}\ \text{“)”} \\
&\mid \text{“}\mathit{consentRequest}\text{(”}\ \text{A, A, ID}\ \text{“)”} \\
&\mid \text{“}\mathit{consent}\text{(”}\ \text{A, A, ID}\ \text{“)”} \\
&\mid \text{“}\mathit{tell}\text{(”}\ \text{A, A, ID, DATAID}\ \text{“)”} \\
&\mid \text{“}\mathit{perform}\text{(”}\ \text{A, ID}\ \text{“)”} \\
&\mid \text{“}\mathit{updateRequest}\text{(”}\ \text{A, A, ID, DATAID, UPDATEID}\ \text{“)”} \\
&\mid \text{“}\mathit{update}\text{(”}\ \text{A, ID, DATAID, UPDATEID}\ \text{“)”} \\
&\mid \text{“}\mathit{forgetRequest}\text{(”}\ \text{A, A, ID, DATAID}\ \text{“)”} \\
&\mid \text{“}\mathit{forget}\text{(”}\ \text{A, ID, DATAID}\ \text{“)”} \\
&\mid \text{“}\mathit{forward}\text{(”}\ \text{ID, DATAID, ID, DATAID}\ \text{“)”}\ ; \\
\text{STATE} &= \text{“}\mathit{actionType}\text{(”}\ \text{ID, ACTIONTYPE}\ \text{“)”} \\
&\mid \text{“}\mathit{dataType}\text{(”}\ \text{ID, DATAID, DATATYPE}\ \text{“)”} \\
&\mid \text{“}\mathit{forwardList}\text{(”}\ \text{ID, DATAID, A}\ \text{“)”} \\
&\mid \text{“}\mathit{duration}\text{(”}\ \text{ID, DATAID, DURATION}\text{“)”} \\
&\mid \text{“}\mathit{responsible}\text{(”}\ \text{A, ID}\ \text{“)”} \\
&\mid \text{“}\mathit{contact}\text{(”}\ \text{A, ID}\ \text{“)”} \\
&\mid \text{“}\mathit{owner}\text{(”}\ \text{ID, DATAID, A}\ \text{“)”}\ ; \\
\text{FW} &= \text{“[”}\{\text{A}\}\text{“]”}\ ;
\end{aligned}
$$

FIGURE 2 – Prédicats du langage $\mathcal{L}_{DLP}$

Le langage de la logique DLP est construit en appliquant des modalités déontiques et temporelles aux prédicats de $\mathcal{L}_{DLP}$. La classe de modalités Ob^{ν} est constituée de modalités d'obligation SDL, une pour chaque autorité normative ν reconnue par l'agent PAw. Les modalités temporelles utilisées en SDL sont $\mathcal{U}$, H et X^{-1}. On peut ainsi également utiliser les modalités abrégées X, G, F et P, ainsi que la classe de modalités X^n ($n \in \mathbb{Z}$) permettant d'atteindre un instant à une distance déterminée dans le passé ou le futur. On pourra également utiliser des versions affaiblies des opérateurs (comme F^- ou $\mathcal{U}^-$), incluant le présent. On choisit une expressivité plus grande dans le futur que dans le passé, la logique DLP devant certes permettre de raisonner sur des événements passés, mais étant destinée en premier lieu à assister l'agent PAw dans la planification de ses actions futures. Cette intuition s'avérera justifiée au vu de la forme que prendront les réglementations temporelles exprimées en DLP.

On notera que ce langage est destiné à être utile à la fois à un agent de service et à un agent utilisateur, l'agent PAw pouvant, suivant le contexte, assumer les deux rôles. Utilisé par un agent de service, il permet de mettre en relation les normes sur la protection des données personnelles avec les caractéristiques des traitements et avec les actes de langages effectués ou planifiés, de manière à s'assurer que les agissements de l'agent restent respectueux de la vie privée des utilisateurs. Du point de vue d'un agent utilisateur, il permet de vérifier que les actes

de langages et les informations déclarées semblent compatibles avec les normes, et donc que les traitements mis en œuvre par les agents distants semblent respectueux des données personnelles.

On peut également observer que le langage ne propose pas en soi de méthode de gradation du caractère privé des informations, permettant d'étiqueter les données comme plus ou moins sensibles. L'intention ici est de permettre ce type de raisonnement en le fondant non pas sur une annotation nécessairement manuelle, mais sur les caractéristiques essentielles des données et des traitements. Le langage DLP permet par exemple de construire des normes différentes pour les données ayant trait à l'identité bancaires, pour les informations de profil communiquées à des services commerciaux, ou encore pour les types de données listées par la loi au titre de la protection contre la discrimination. En d'autres termes, les métadonnées manipulées par DLP sont plus riches qu'une simple gradation, et la spécialisation du raisonnement se fait directement sur les aspects sémantiques.

4.1.2 Aspects sémantiques

Les formules de la logique DLP s'interprètent sur des structures sémantiques de type Kripke qui correspondent à des grilles à deux dimensions. La figure 3 est une représentation graphique d'un modèle DLP. La première dimension, représentée horizontalement, est le temps, indexé par des **dates** (t_i). Cela signifie qu'une ligne horizontale est le déroulement d'une seule exécution dans le temps du système. Verticalement, on retrouve les différentes **histoires** (h_i) ou exécutions possibles du système. En d'autres termes, sur une colonne on retrouve les différents états possibles du systèmes à une date donnée, les différentes lignes correspondant aux différentes exécutions possibles. Les mondes (nœuds du graphe) sont appelés des **instants** (notés par des i) et correspondent à des couples (histoire, date). Entre ces mondes on retrouve une relation d'accessibilité temporelle (en trait plein sur le schéma) et une relation d'accessibilité déontique (en trait pointillé) pour chaque modalité d'obligation. La relation d'accessibilité temporelle relie chaque instant (et l'état du système associé) à son successeur dans le temps, pour chacune des exécutions possibles du système. Une relation d'accessibilité déontique permet de désigner, à partir d'un instant donné (état d'une des exécutions possibles du système à une date donnée), dans lequel certaines obligations sont vraies, les instants du modèle où ces obligations sont respectées. Ici, par exemple, l'obligation $Ob\ \psi$ est vraie à l'instant (h_1, t_2), donc les instants qui y sont reliés par la relation d'accessibilité déontique sont ceux auxquels ψ est vraie. La flèche brisée représente quant à elle une violation des normes à l'instant (h_0, t_1).

4.2 Conflits d'obligations

Les travaux existants en matière de gestion des conflits normatifs s'appuient, comme nous l'avons vu, sur la notion d'inconsistance d'un système SDL. Cette notion peut être formalisée comme suit en DLP :

Définition 1 (Conflits normatifs en DLP) *Un ensemble de formules DLP contient un conflit normatif si et seulement si :*

- *Il permet de dériver $\perp$ à l'aide de l'axiomatique de la logique DLP ;*
- *Dans toute dérivation possible de $\perp$ à partir de cet ensemble, l'un des axiomes $Ob - K$ ou $Ob - D$ spécifiques à SDL est impliqué.*

Cette inconsistance peut découler de deux obligations sur des formules contradictoires $(Ob\ \varphi, Ob\ \neg\varphi)$ ou encore d'une interdiction et d'une permission portant sur la même formule $(Ob\neg\varphi, Per\varphi)$. Cependant, lorsqu'un agent est confronté à ce dernier cas, il a toujours la possibilité de respecter l'interdiction, auquel cas aucune des deux formules n'est violée. Par conséquent, en dépit de la contradiction formelle, la conjonction de ces deux normes laisse tout de même à l'agent la possibilité d'un comportement sain. Si de telles normes sont émises par une même autorité normative, on peut considérer que cette autorité n'est pas suffisamment cohérente avec elle-même, mais ce type de conflit survenant entre deux autorités différentes doit pouvoir être accepté par l'agent PAw. C'est pourquoi nous introduisons spécifiquement la notion de conflit d'obligations [39], qui est un conflit normatif n'impliquant pas de permission.

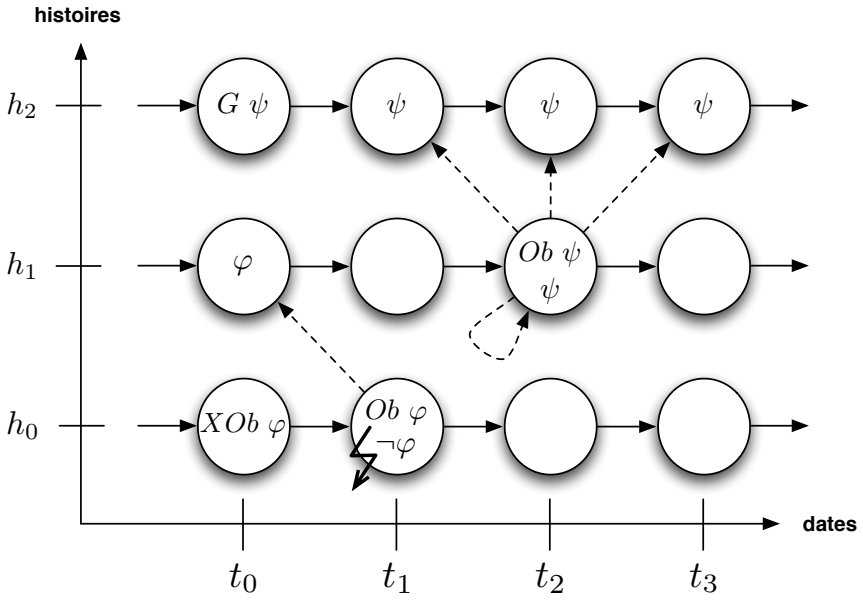

FIGURE 3 – Exemple de modèle DLP (avec une seule modalité d'obligation)

Pour cela, nous considérons la logique DLP+, qui est la logique DLP augmentée de l'axiome (2) (équivalent à celui utilisé par Balbiani en 2005), qui a pour caractéristique d'interdire les conflits entre obligations tout en laissant possibles les incohérences inter-autorités impliquant des permissions.

$$\bigwedge_{i=1}^{n} Ob^{\nu_i}\ \varphi_i \rightarrow Per^{\nu_{i+1}} \bigwedge_{i=1}^{n} \varphi_i \tag{2}$$

C'est ensuite par rapport à ce nouveau système que nous définissons le conflit d'obligations :

Définition 2 (Conflits d'obligations en DLP) *Un ensemble de formules DLP exempt de conflits normatifs comporte des conflits d'obligation si et seulement si ce même ensemble de formules est inconsistant au sens de la logique DLP+.*

Cette définition permet une détection formelle des conflits d'obligations dans les ensembles de normes dont dispose l'agent. Reste ensuite à arbitrer ces conflits, ce qui est fait ici via une procédure de fusion avec préférences. La relation de préférence utilisée est le concept de **prévalence normative**, local à l'agent et représenté par une relation de préordre total $\lhd$, $\nu_1 \lhd \nu_2$ signifiant que l'autorité normative ν_1 prévaut sur l'autorité ν_2, pour l'agent considéré. Cette relation ayant pour objet de faire correspondre le raisonnement de l'agent avec les choix de son propriétaire, elle est spécifiée par ce dernier dans la configuration de l'agent. La procédure d'arbitrage vise donc à désactiver un certain nombre de normes, les moins nombreuses possibles et correspondant à des autorités de prévalence normative la plus faible possible. L'algorithme d'arbitrage que nous avons proposé dans [39] consiste à choisir un sous-ensemble conflictuel minimal (ensemble de normes contenant un conflit d'obligations et tel que le retrait de n'importe laquelle des normes supprime le conflit) et à désactiver une des normes de prévalence minimale, puis à répéter l'opération jusqu'à ce que l'ensemble des normes soit exempt de conflits.

Cet algorithme comporte principalement deux points de choix : le choix du sous-ensemble conflictuel à traiter et le choix de la norme à désactiver. Nous avons identifié plusieurs stratégies pour le premier point de choix. Il s'avère notamment qu'il est impossible de mettre en lumière une domination entre deux des plus intéressantes, à savoir la stratégie sélectionnant l'ensemble dont la norme à désactiver est présente dans le plus grand nombre de sous-ensembles conflictuels minimaux, et celle sélectionnant l'ensemble dont la norme à désactiver est de prévalence la plus

élevée parmi tous les sous-ensembles conflictuels minimaux. Si toutes les stratégies proposées permettent d'obtenir en sortie un ensemble de normes cohérent, certaines d'entre elles permettent de récupérer plus de normes, alors que d'autres fournissent des normes moins nombreuses mais de prévalence plus élevée. Si ce problème d'optimisation multi-critère reste pour l'instant ouvert, il paraîtrait naturel que le choix de la stratégie à adopter soit guidé, au cas par cas, par la sémantique du problème à résoudre par l'agent PAw.

4.3 Normes datées

Nous avons déjà mis en avant la nécessité d'un opérateur apte à attacher une échéance à une obligation donnée. Pour travailler sur le concept, nous introduisons dans le langage des **propositions-dates**, des propositions atomiques (notées δ_i et caractérisées par un prédicat $date/1$) qui surviennent une et une seule fois dans le flot de temps (correspondant ainsi aux constantes systématiques de cadre d'Åqvist), comme spécifié en (3).

$$date(\delta) \stackrel{def}{=} \begin{cases} \delta \wedge G\neg\delta \wedge H\neg\delta \\ \vee F(\delta \wedge G\neg\delta \wedge H\neg\delta) \\ \vee P(\delta \wedge G\neg\delta \wedge H\neg\delta) \end{cases} \tag{3}$$

L'analyse des propositions existantes en matière d'obligations avec échéances ([25, 23, 9, 21]) nous a amenés à lister huit propriétés fondamentales qu'un opérateur $Ob(\varphi, \delta)$ (signifiant l'obligation de satisfaire φ avant l'échéance δ) devrait satisfaire, dans le cadre du formalisme DLP, pour représenter efficacement le concept [1]. Nous prenons comme hypothèse que la spécification de φ est suffisamment précise pour rendre unique une obligation sur φ, permettant ainsi de distinguer les obligations les unes des autres.

1. **La date d'échéance doit être une proposition-date** : pour qu'une obligation assortie d'une échéance ait un sens pour l'agent PAw, il faut que cette échéance soit définie de manière claire et distincte. La définition que nous avons donnée des propositions-dates fait en sorte qu'une telle échéance existe vraiment dans le flot de temps et qu'elle ne survienne qu'une seule fois.
2. **L'échéance doit être située strictement dans le futur** : c'est une évidence pour un interprète humain, une échéance dans le passé (ou même dans le présent) ne laisse aucune marge de manœuvre à un agent, son respect est indépendant de ses actions et de ses capacités.
3. **Les obligations sur $\top$ doivent être des tautologies** [2] : nous introduisons ce principe par mesure de cohérence avec l'obligation déontique immédiate, pour laquelle $Ob\top$ est un théorème. De la même manière, $Ob(\top, \delta)$ devrait être un théorème.
4. **Les obligations sur $\bot$ doivent être des antilogies** (*i.e.* être toujours fausses) : de la même manière que précédemment et pour les mêmes raisons, $\neg\, Ob(\bot, \delta)$ devrait être un théorème.
5. **Les obligations non respectées doivent être abandonnées à l'échéance** : le choix de ce principe est davantage d'ordre philosophique que purement logique. Nous estimons que si l'agent PAw n'a pas pu ou voulu se conformer dans les temps à l'obligation, alors, le mal étant fait, cette obligation ne doit plus pouvoir être dérivée. Ceci n'exclut pas l'existence éventuelle d'obligations *contrary-to-duties* qui pourraient survenir suite à la violation d'une obligation avec échéance. Ce critère permet à l'agent de ne pas avoir à tenir compte indéfiniment d'obligations datées qu'il ne pourra jamais respecter et qui sont peut-être caduques pour des raisons liées à leur signification dans le monde réel. Il est de plus une

1. La multiplicité des autorités normatives n'étant pas exploitée ici, nous n'utiliserons pas l'exposant associé à l'opérateur Ob, dans un souci de lisibilité.

2. Il faut être attentif à ne pas confondre la valeur de vérité (ou la dérivabilité) des obligations avec le fait qu'elles soient respectées ou non. Dans le cas d'une modalité normale, dotée de la règle de nécessitation, les deux notions correspondent pour les obligations sur des théorèmes. Cependant, l'obligation avec échéance n'est a priori pas une modalité normale.

conséquence logique de notre deuxième exigence, suivant laquelle une échéance doit être située dans le futur. L'alternative qui consisterait à maintenir l'obligation après l'échéance, éventuellement jusqu'à ce qu'elle soit respectée, relèverait de la concession d'une extension au délai initial, comme cela a pu être discuté par d'autres auteurs [11]. Si nous n'excluons pas la possibilité de demander que l'obligation soit de nouveau considérée avec une nouvelle échéance, nous préférons considérer séparément le premier échec.

6. **Les violations doivent être définies comme étant ponctuelles** : dans certains formalismes logiques, on considère que l'agent est dans un « état » de violation dès lors qu'une obligation avec échéance n'a pas été respectée. Étant donné l'expressivité qui nous est donnée par la logique DLP, et notamment la possibilité de raisonner sur le passé, il nous paraît suffisant de pouvoir considérer que l'agent a violé telle obligation à une date donnée du passé. Cela revient à considérer la violation comme un événement plutôt que comme un état, ce qui est à la fois un débat sémantique et une conséquence de l'expressivité du formalisme choisi.
7. **Le principe de propagation doit être respecté** : ce principe dit que l'obligation avec échéance doit être maintenue d'une date à la suivante, tant que l'obligation n'a pas été respectée ou que l'échéance n'est pas atteinte.

$$Ob(\varphi, \delta) \wedge \neg(\varphi \vee \delta) \wedge \neg X\delta \rightarrow XOb(\varphi, \delta) \tag{4}$$

8. **Le principe de monotonie doit être respecté** : ce principe dit que s'il existe une obligation avec une échéance donnée, alors on peut dériver la même obligation pour une échéance plus lointaine (5). Par exemple, si l'agent PAw doit détruire une information avant le premier juillet courant, alors il est normal de considérer qu'il doit également la détruire avant le premier août.

$$Ob(\varphi, \delta) \wedge F(\delta \wedge F^{-}\delta') \rightarrow Ob(\varphi, \delta') \tag{5}$$

Les six premiers critères ont été débattus par Dignum *et al.*, avec cependant des conclusions différentes. En effet, le formalisme utilisé étant assez différent du langage DLP, les violations y sont notamment considérées comme permanentes (à cause de l'impossibilité d'un raisonnement sur le passé). Sur certains aspects comme la nature des échéances, les auteurs engagent un débat sans prendre délibérément position. Notamment, ils acceptent l'éventualité d'utiliser $\bot$ comme échéance pour signifier une obligation sur le futur sans échéance particulière, ce qui reste possible en DLP avec $Ob\ F\varphi$ mais qui ne nous semble pas pertinent. Concernant les obligations sur $\top$, les auteurs (utilisant un opérateur de type *stit* impliquant la notion de responsabilité des agents) récusent la possibilité pour un agent de satisfaire à une obligation sur une tautologie. Dignum *et al.* nous rejoignent cependant sur l'inconsistance nécessaire des obligations portant sur des contradictions. Les deux critères de monotonie et de propagation ont quant à eux été introduits par Brunel *et al.* et nous les avons adoptés en les adaptant à notre notion plus stricte d'échéance.

L'opérateur proposé en (6) a été conçu pour respecter ces huit critères,avec cette exception que le principe de monotonie est remplacé par celui de semi-monotonie : la monotonie n'est respectée que lorsque l'obligation est respectée. Cette adaptation permet d'éviter la dérivation automatique d'une infinité de violations pour la même obligation.

$$Ob(\varphi, \delta) \stackrel{def}{=} \begin{cases} date(\delta) \\ F\delta \\ Ob(F^{-}(\varphi \wedge F\delta))\ \mathcal{U}^{-}(\varphi \vee \delta) \end{cases} \tag{6}$$

L'opérateur (7) correspond à la caractérisation d'une violation d'une telle obligation avec échéance.

$$violOb(\varphi, \delta) \stackrel{def}{=} \begin{cases} date(\delta) \\ \delta \\ P(Ob(\varphi, \delta) \wedge \neg\varphi\ \mathcal{U}^{-}\delta) \end{cases} \tag{7}$$

Par une méthodologie similaire, huit propriétés fondamentales, un opérateur $For(\varphi, \delta)$ et sa violation associée $violFor(\varphi, \delta)$ ont également été proposés pour traduire le concept d'interdiction maintenue sur une période [38]. Ces deux opérateurs vont constituer la clef de voûte de la traduction en DLP des réglementations en matière de protection des données personnelles.

4.4 Mise en œuvre

4.4.1 Expression d'une norme en DLP

Nous allons maintenant illustrer avec un exemple comment une clause réglementaire peut être transcrite en langage DLP. On considère la clause hypothétique suivante (portant sur le premier axe réglementaire, l'information) : *Les clients doivent être informés au moins une semaine à l'avance de la nature de tout traitement utilisant leurs données personnelles.* Comme pour la plupart des normes, on peut considérer tout naturellement une structure conditionnelle, avec un contexte déclencheur d'une clause déontique (ici, une obligation). Cette conditionnalité pourra être traduite par une implication entre une formule de contexte et une formule déontique. Cet exemple, quoique simple à comprendre pour l'utilisateur humain, et d'un type que l'on pourra retrouver dans des textes légaux ou des règlements intérieurs, est d'une interprétation relativement complexe pour un agent artificiel. La norme peut en effet être considérée soit comme une interdiction, soit comme une obligation.

La norme peut être vue comme une interdiction dans le cas où l'agent n'a pas encore informé le client, ou bien si l'information a eu lieu moins d'une semaine dans le passé. Dans le premier cas, l'agent a interdiction de procéder au traitement avant une semaine, ce qu'exprime la formule (8). L'expression de contexte en partie gauche signifie, en DLP, que ActionType est le type du processus ProcessID, que le client Client n'a jamais été informé de la nature de ce traitement, qu'il est le propriétaire de données impliquées dans ce traitement, et que δ est une date située une semaine dans le futur. Le conséquent est pour l'agent une interdiction de procéder au traitement ProcessID, maintenue pendant au moins une semaine.

$$\left.\begin{array}{l} \mathit{actionType}(\text{ProcessID}, \text{ActionType}) \\ H\neg \mathit{informActionType}(\texttt{self}, \text{Client}, \\ \quad \text{ProcessID}, \text{ActionType}) \\ \mathit{owner}(\text{ProcessID}, \text{DataID}, \text{Client}) \\ \mathit{date}(\delta) \wedge X^{7*24}\delta \end{array}\right\} \rightarrow \begin{array}{l} For(\mathit{perform}(\texttt{self}, \\ \quad \text{ProcessID}), \delta) \end{array} \tag{8}$$

Le fait de considérer le deuxième cas (le client n'a pas été informé plus d'une semaine en amont) permet éventuellement d'ajouter une formule redondante, un garde-fou en plus de la première formule, une interdiction immédiate de procéder au traitement.

Cette même réglementation peut également être vue comme une obligation pour l'agent de prévenir le client à temps si un traitement est planifié dans le futur. C'est ce qu'exprime la formule (9). Ici l'antécédent nous dit que δ est une date située dans le futur, et néanmoins une semaine avant un traitement programmé. On retrouve également le fait que le client n'a pas encore été informé.

$$\left.\begin{array}{l} \mathit{actionType}(\text{ProcessID}, \text{ActionType}) \\ \mathit{date}(\delta) \\ F(\delta \wedge X^{7*24-1}\mathit{perform}(\texttt{self}, \text{ProcessID})) \\ H\neg \mathit{informActionType}(\texttt{self}, \text{Client}, \\ \quad \text{ProcessID}, \text{ActionType}) \\ \mathit{owner}(\text{ProcessID}, \text{DataID}, \text{Client}) \end{array}\right\} \begin{array}{l} \rightarrow Ob(\mathit{informActionType}(\texttt{self}, \\ \quad \text{Client}, \text{ProcessID}, \text{ActionType}), \delta) \end{array} \tag{9}$$

Ces deux visions d'une même réglementation sont complémentaires. La première correspond à des éléments de contraintes immédiats sur le fonctionnement de l'agent, alors que la seconde lui fournit des éléments pour l'assister dans la planification de ses actions. Lorsque l'on parcourt les six axes réglementaires, on se rend compte que les exemples de normes peuvent être appréhendés suivant cette même philosophie, débouchant sur des formules de même structure : un antécédent de contexte, formule essentiellement non déontique, une implication logique et une formule déontique (obligation ou interdiction, avec ou sans composante temporelle).

4.4.2 Implémentation

Cette dernière constatation a guidé le développement du composant de raisonnement normatif de l'agent PAw, écrit en Prolog, qui tire parti de ces simplifications syntaxiques et de la structure des obligations avec échéances et des interdictions maintenues (notamment les principes de propagation et de monotonie, qui permettent de dériver de nouvelles formules facilitant la recherche des conflits d'obligations). En l'état actuel du développement, le composant de raisonnement normatif s'intègre dans un agent PAw de démonstration, ne mettant en œuvre que des traitements très simples, mais illustrant le principe des mécanismes de protection.

Le composant de raisonnement normatif a pour objectif principal la détection et l'arbitrage des conflits d'obligations. Il est mis à contribution à chaque modification du contexte normatif (apparition d'une nouvelle autorité, par exemple), afin de fournir en sortie un ensemble de croyances normatives exempt de conflits d'obligations, qui va constituer la politique de sécurité de l'agent. Ce composant est construit autour de l'hypothèse que les normes émises par les autorités sont, comme dans l'exemple précédent, sous la forme d'un contexte d'activation et d'un opérateur déontique (daté dans le cas général). La détection des sous-ensembles conflictuels minimaux, nécessaire à la résolution des conflits d'obligations, s'effectue alors uniquement sur la partie normative de la formule, dans laquelle les échéances sont traitées comme des intervalles de variables contraintes. Les intersections d'intervalles entre les normes présentant des obligations et celles présentant des interdictions permet alors de déterminer les possibles conflits. Une fois les sous-ensembles conflictuels détectés, le choix de la norme à désactiver revient au code JACK de l'agent, qui dispose de la prévalence normative des différentes autorités et peut donc constituer la politique de sécurité de l'agent en désactivant les normes nécessaires.

Parmi les croyances normatives de l'agent PAw, les contextes d'activation des différentes normes de cette politique de sécurité sont stockées à part, et une tâche de fond les parcourt en permanence pour déterminer s'ils sont applicables. Lorsque c'est le cas, la partie déontique de la norme correspondante est déplacée dans une nouvelle base de normes actives. À cette étape, les obligations vont donner lieu à la création de buts normatifs, requérant une action spécifique de la part de l'agent par le biais des mécanismes cognitifs natifs du modèle JACK. Les interdictions, quant à elles, sont examinées par les plans normatifs, activés par le biais des mécanismes de rappel automatique des plans JACK, à chaque fois qu'un plan métier souhaite accéder à des croyances à caractère personnel. Si les actions que le plan annonce devoir entreprendre vont à l'encontre d'une des interdictions actives, l'accès aux données lui est refusé.

Il faut toutefois noter que ces modes de contrôle d'usage s'appuient sur l'hypothèse d'un codage correct de l'agent. En particulier, on suppose que les plans métier déclarent convenablement les traitements qu'ils mettent en œuvre, de manière à ne pas tromper les mécanismes de contrôle. De plus, plusieurs fonctionnalités intéressantes resteraient encore à intégrer, comme le déclenchement d'une interaction avec l'utilisateur lorsqu'une norme (même désactivée) doit être violée.

4.5 Limitations formelles

La logique DLP semble raisonnablement adaptée à la représentation de divers types de réglementations sur la protection des données personnelles. Toutefois, certains types de normes, moins courantes dans ce contexte particulier, ne peuvent être prises en compte dans ce modèle.

Tout d'abord, il n'est pas nativement possible de représenter des obligations périodiques ou récurrentes, c'est-à-dire une seule obligation associée à un ensemble de plusieurs échéances. Les opérateurs que nous avons proposés imposent d'associer chaque obligation à une échéance au plus. Cette limitation pourrait éventuellement être contournée en traduisant de telles normes en règles de plus haut niveau (toujours exprimées en DLP) produisant des ensembles finis ou même infinis d'obligations avec échéances « classiques », de manière récursive. Cette solution pourrait être encapsulée dans une abréviation syntactique plus simple.

De la même manière, la possibilité d'octroyer un délai supplémentaire à l'échéance n'est pas prévu dans DLP. Il est bien sûr possible de construire une nouvelle abréviation capturant cette notion, les mécanismes mis en œuvre n'étant pas fondamentalement plus riches, mais la notion

d'extension ne peut être que simulée au sens où elle constituera une nouvelle obligation, et non une continuation de la précédente.

Par ailleurs, DLP ne permet pas d'utiliser la notion de *liveline* (à opposer à *deadline*, pour échéance) utilisée par Henrique Lopes Cardoso et Eugenio Oliveira [11]. Si une échéance est la date avant laquelle l'obligation doit être remplie, la *liveline* est la date après laquelle elle doit être remplie : une obligation remplie trop tôt peut alors être considérée comme une violation. Néanmoins, nous n'avons pour l'instant pas retrouvé cette notion dans les réglementations sur la protection des données personnelles.

5 Conclusion

Nous avons présenté les principes de l'architecture de l'agent PAw, destiné à interfacer les interactions d'un utilisateur humain avec un système distribué en se chargeant de la protection de ses données personnelles. Nous avons particulièrement porté l'accent sur son composant de raisonnement normatif, qui lui permet d'appréhender les diverses réglementations qui s'appliquent afin d'adapter son comportement en conséquence. La conception de ce composant a donné lieu à de nouveaux développements théoriques et techniques, notamment dans les domaines des conflits normatifs et des obligations avec échéances. Il est maintenant intéressant d'observer comment l'agent PAw pourrait être utilisé dans un cadre applicatif.

5.1 Intégration au sein d'architectures distribuées

Les fonctionnalités et composants que nous avons décrits ici ne permettent à l'agent PAw d'assurer la protection des données personnelles que de manière locale, limitée à ses propres actions. Néanmoins, les risques les plus significatifs pour la vie privée de l'utilisateur surviennent lorsque ces données sont partagées avec d'autres agents, par exemple au cours d'une nécessaire collaboration visant à fournir un service. Les possibilités d'interactions offertes par le composant d'interface avec les protocoles, non détaillé ici, lui permettent d'interagir avec un certain nombre d'architectures applicatives distribuées et de protocoles d'interaction prédéfinis. De manière statique, on fournit à l'agent PAw un certain nombre d'informations factuelles concernant ces architectures, traduisant leur capacité à respecter une politique d'usage donnée et la confiance que l'agent peut avoir dans le respect de cette politique ou dans la réalisation de propriétés distantes plus générales. Dans cette optique, nous avons proposé [40] une catégorisation des différentes architectures possibles en fonction du niveau de confiance. Nous avons également réalisé une analyse des architectures fondées sur le *Trusted Computing*, qui sont à même de fournir des garanties très fortes, et avons proposé une architecture complémentaire visant à éliminer les risques inhérents à cette technologie.

L'agent PAw dispose donc d'une bibliothèque de protocoles et d'architectures. En fonction des méthodes supportées par ses interlocuteurs, de sa politique de sécurité et du niveau de sensibilité des données impliquées, l'agent peut alors prendre une décision informée d'utiliser telle ou telle architecture, ou encore tenter de décourager l'utilisateur de procéder à la transaction si celle-ci s'avère trop risquée au vu des moyens techniques mis en œuvre. Cependant, l'articulation entre le raisonnement normatif complexe, interne à l'agent PAw, et l'utilisation des architectures distribuées est encore embryonnaire. En effet, les possibilités se limitent actuellement à mettre en lien chacun des six axes réglementaires avec des niveaux de confiance correspondants pour chaque architecture. L'importance de chacun des six axes dans un cas donné (cette évaluation étant le fruit du raisonnement normatif en DLP) permet alors de déterminer des préférences sur les architectures à utiliser. Néanmoins, en l'état actuel des développements la politique d'usage des données résultant du raisonnement de l'agent PAw n'est pas directement réutilisée dans les diverses architectures disponibles.

5.2 Perspectives

Comme nous l'avons suggéré, l'intérêt majeur de l'agent PAw est d'assister l'utilisateur humain dans son interaction avec une application informatique distribuée. L'agent PAw ayant la garde des données de l'utilisateur et la connaissance du contexte normatif dans lequel il évolue, il est en mesure de prendre des décisions visant à protéger son propriétaire tout en participant à l'application, ou tout du moins de l'avertir d'une situation à risque. Divers contextes applicatifs peuvent être imaginés dans lesquels les fonctionnalités de l'agent PAw seraient utiles à un utilisateur humain.

Dans le domaine de la santé par exemple, la protection du caractère privé des données du patient est particulièrement importante, même si elle doit souffrir les exceptions nécessaires à la santé de celui-ci. Entre les lois générales sur la protection des données personnelles, les réglementations particulières à l'exercice des professions médicales et les règles internes des institutions hospitalières, il est facile pour un patient de perdre pied et de confier aveuglément ses données à la profession médicale dans son ensemble. Si cette confiance est dans son principe justifiée, il est envisageable que parmi l'ensemble des acteurs (médecins, auxiliaires médicaux, personnels administratifs...), il s'en trouve qui ne soient pas non plus en mesure d'appréhender la complexité du contexte normatif. Le choix visant à confier à un programme informatique les décisions concernant l'accès aux données médicales doit bien évidemment être considéré avec la plus grande prudence, cependant les fonctionnalités développées dans le cadre de l'agent PAw semblent à même de traiter l'aspect hétérogène du problème ainsi que la nécessaire et vitale hiérarchisation des différents systèmes réglementaires en interaction.

La nature de l'agent PAw lui réserve sans doute également une place dans les scénarios d'intelligence ambiante. Les applications correspondantes sont caractérisées par la multitude des acteurs et des interactions. Les équipements de l'utilisateur sont constamment en collaboration avec les artéfacts de l'environnement et avec des équipements appartenant à d'autres utilisateurs ou organisations. La personnalisation des environnements et des services, au cœur du concept d'intelligence ambiante, nécessite de nombreuses transmissions de données personnelles, qui doivent, pour être efficaces, être transparentes à l'utilisateur. La capacité de l'agent PAw à prendre des décisions raisonnées et autonomes en la matière en fait un modèle adapté pour des équipements d'intelligence ambiante disposant de ressources suffisantes pour mettre en œuvre ces fonctionnalités.

Le commerce électronique, enfin, constitue un terrain de choix pour l'amélioration de la protection des données personnelles. La communication de ces dernières est vitale à la fourniture des services ou des produits, et les risques comme les abus en matière de vie privée sont nombreux. Le caractère hétérogène du contexte normatif est souvent incarné par la multiplicité des ressorts législatifs dont dépendent les transactions les plus complexes, et le déroulement de l'opération nécessite bien souvent des interactions entre des partenaires multiples : fournisseurs de services ou de produits, banques, sociétés de livraison ou de manutention, régies publicitaires, tiers de confiance, banques de données, intermédiaires d'agrégation... Les applications de commerce électronique, et plus généralement les interactions ouvertes sur Internet, constituent probablement le contexte d'expression privilégié de l'agent PAw.

En conclusion, l'utilisation d'agents utilisateurs cognitifs pour assister les utilisateurs humains dans la gestion de leurs données personnelles peut s'avérer d'une aide particulièrement utile lorsque l'ensemble des normes, droits, devoirs et interdictions pesant sur l'utilisateur et sur les autres acteurs du systèmes deviennent trop complexes à appréhender. L'agent PAw, qui constitue notre réponse à ce problème, a la capacité d'absorber cette complexité et cette hétérogénéité pour en tirer, au profit de son propriétaire humain, une ligne de conduite claire, qu'il est capable d'expliquer et de justifier et qui lui permettra de protéger au mieux sa vie privée et celle de ses interlocuteurs.

Références

[1] T. Ågotnes, W. van der Hoek, J. A. Rodríguez-Aguilar, C. Sierra, and M. Wooldridge. On the logic of normative systems. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, Hyderabad, India, January 2007.

[2] AOS Autonomous Decision-Making Software. JACK. http ://www.aosgrp.com/products/jack/.

[3] L. Åqvist. Combinations of tense and deontic modalities. In A. Lomuscio and D. Nute, editors, *Proceedings of the 7th International Workshop on Deontic Logic in Computer Science (DEON 2004)*, volume 3065 of *LNCS*, pages 3–28, Madeira, Portugal, May 2004. Springer.

[4] P. Balbiani. Constitution et développement d'une logique des modalités aléthiques, déontiques, dynamiques, et temporelles en vue de la formalisation du raisonnement sur les actions et sur les normes. In A. Herzig, Y. Lespérance, and A.-I. Mouaddib, editors, *Troisièmes journées francophones sur les modèles formels de l'interaction (MFI'05)*, Caen, France, 2005. Cépaduès éditions.

[5] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic.* Cambridge University Press, 2001.

[6] G. Boella, L. van der Torre, and H. Verhagen. Introduction to normative multiagent systems. *Computational & Mathematical Organization Theory*, 12(2-3) :71–79, 2006.

[7] M. Bouzeghoub and D. Kostadinov. Data personalization : a taxonomy of user profiles knowledge and a profile management tool. Technical report, Accès Personnalisé à des Masses de Données (ACI APMD, project MD-33), June 2006.

[8] M. E. Bratman. *Intentions, plans and practical reason.* Harvard University Press, Cambridge, Massachusetts, USA, 1987.

[9] J. Brunel, J.-P. Bodeveix, and M. Filali. A state/event temporal deontic logic. In *Eighth International Workshop on Deontic Logic in Computer Science (DEON'06)*, number 4048 in LNCS, 2006.

[10] A. Bürkle, A. Hertel, W. Müller, and M. Wieser. Evaluating mobile agent platform security. In K. Fischer, I. J. Timm, E. André, and N. Zhong, editors, *Proceedings of the 4th German Conference on Multiagent System Technologies (MATES 2006)*, volume 4196 of *LNCS*, pages 159–171, Erfurt, Germany, September 2006. Springer.

[11] H. L. Cardoso and E. Oliveira. Directed deadline obligations in agent-based business contracts. In *International Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems (COIN'09)*, Budapest, Hungary, 2009.

[12] J. Carmo and A. J. I. Jones. Deontic database constraints and the characterisation of recovery. In *Second international workshop on deontic logic in computer science (DEON'94)*, pages 56–85, Amsterdam, The Netherlands, 1994.

[13] B. F. Chellas. *Modal Logic, an Introduction.* Cambridge University Press, 1980.

[14] G. Chicoisne. *Dialogue entre agents naturels et agents artificiels. Une application aux communautés virtuelles.* PhD thesis, Institut National Polytechnique de Grenoble, 2004.

[15] L. Cholvy and F. Cuppens. Analyzing consistency of security policies. In *Proceedings of the 18th IEEE Symposium on Research in Security and Privacy*, Oakland, CA, USA, May 1997.

[16] L. Cholvy and F. Cuppens. Reasoning about norms provided by conflicting regulations. In H. Prakken and P. McNamara, editors, *Norms, Logics and Information Systems : New Studies in Deontic Logic and Computer Science*, pages 247–264, Amsterdam, the Netherlands, 1998. IOS Press.

[17] P. R. Cohen and H. J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42(2-3) :213–261, 1990.

[18] L. Crépin, G. Piolle, O. Boissier, and Y. Demazeau. Des systèmes normatifs comme outils de protection de la vie privée. In *Intelligence Artificielle et Web Intelligence (IAWI'07)*, Grenoble, France, July 2007. AFIA.

[19] F. Cuppens. A logical analysis of authorized and prohibited information flows. In *IEEE Symposium on Research in Security and Privacy*, pages 100–109, Oakland, California, USA, 1993. IEEE Computer Society Press.

[20] F. Cuppens. A logical formalization of secrecy. In *Computer Security Foundations Workshop VI*, pages 53–62, Franconia, USA, 1993. IEEE Computer Society Press.

[21] F. Cuppens, N. Cuppens-Boulahia, and T. Sans. Nomad : a security model with non atomic actions and deadlines. In *18th IEEE Computer Security Foundations Workshop (CSFW'05)*, 6 2005.

[22] Y. Demazeau. Créativité emergente centrée utilisateur. In *11èmes Journées Francophones sur les Systèmes Multi-Agents (JFSMA'03)*, pages 31–36, Hammamet, Tunisia, 11 2003.

[23] R. Demolombe. Formalisation de l'obligation de faire avec délais. In *Troisièmes journées francophones des modèles formels de l'interaction (MFI'05)*, Caen, France, may 2005.

[24] Y. Deswarte and C. Aguilar-Melchor. Current and future privacy enhancing technologies for the internet. *Annals of Telecommunications*, 61(3-4) :399–417, 2006.

[25] F. Dignum, J. Broersen, V. Dignum, and J.-J. Meyer. Meeting the deadline : Why, when and how. In M. G. Hinchey, J. L. Rash, W. Truszkowski, and C. Rouff, editors, *Third International Workshop on Formal Approaches to Agent-Based Systems (FAABS'04)*, number 3228 in LNCS, pages 30–40, Greenbelt, MD, USA, april 2004. Springer Verlag.

[26] M. d'Inverno, D. Kinny, M. Luck, and M. Wooldridge. A formal specification of dMars. In *Intelligent Agents IV : Proceedings of the Fourth International Workshop on Agent Theories, Architectures and Languages (ATAL'97)*, number 1365 in LNAI. Springer-Verlag, 1997.

[27] J. Ferber. *Les systèmes multi-agents, vers une intelligence collective.* InterEditions, Paris, France, 1995.

[28] J. I. Glasgow, G. H. MacEwen, and P. Panangaden. A logic for reasoning about security. In *Computer security foundations workshop*, pages 2–13, Franconia, USA, 1990. IEEE Computer Society Press.

[29] Y. Haradji, N. Ferrand, and H. Li. *Systèmes Multi-Agents*, chapter Relations à l'Utilisateur et Nouveaux Usages, pages 215–260. Observatoire Français des Techniques Avancées, 2004.

[30] A. J. I. Jones and M. J. Sergot. *Deontic Logic in Computer Science : Normative System Specification.*, chapter On the Characterisation of Law and Computer Systems : The Normative Systems Perspective, pages 275–307. John Wiley and Sons, Chichester, England, 1993.

[31] M. J. Kollingbaum, W. W. Vasconcelos, and N. T. J. García-Camino, Andrés. Conflict resolution in norm-regulated environments via unification and constraints. In M. Baldoni, T. Cao Son, M. B. van Riemsdijk, and M. Winikoff, editors, *Declarative Agent Language and Technologies 2007 (DALT'07)*, number 4897 in LNCS, Honolulu, HI, USA, May 2007. Springer.

[32] M. Martinez Ribas. Towards legal programming : Some legal problems of agent-based mobile commerce (legal issues of e-commerce oriented intelligent agents). European Project Grocer.

[33] P. McNamara. Deontic logic. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy.* Stanford University, 2006.

[34] J.-J. C. Meyer and R. J. Wieringa. *Deontic logic in computer science : normative system specification.* John Wiley and Sons Ltd., 1993.

[35] P. Morris and J. A. McDermid. *Database Security, VI : Status and Prospects*, chapter Formalising and Validating Complex Security Requirements, pages 113–124. Vancouver, Canada, 1992.

[36] R. Ortalo. Using deontic logic for security policy specification. LAAS report 96380, LAAS-CNRS, Toulouse, France, october 1996.

[37] G. Piolle. *Agents utilisateurs pour la protection des données personnelles : modélisation logique et outils informatiques.* PhD thesis, Université Joseph Fourier - Grenoble I, Grenoble, France, 6 2009.

[38] G. Piolle and Y. Demazeau. Obligations with deadlines and maintained interdictions in privacy regulation frameworks. In *Proceedings of the 8th IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'08)*, pages 162–168, Sidney, Australia, December 2008. IEEE Computer Society.

[39] G. Piolle and Y. Demazeau. Une logique pour raisonner sur la protection des données personnelles. In *16e congrès francophone AFRIF-AFIA sur la Reconnaissance de Formes et l'Intelligence Artificielle (RFIA'08)*, Amiens, France, January 2008. AFRIF-AFIA.

[40] G. Piolle and Y. Demazeau. Délégation d'agents pour la protection étendue des données personnelles. In Z. Guessoum and S. Hassas, editors, *17èmes journées francophones sur les systèmes multi-agents (JFSMA'09)*, pages 35–46, Lyon, France, 10 2009. Cépaduès.

[41] G. Piolle and Y. Demazeau. Déléguer la protection des données personnelles à des agents cognitifs. *Revue d'Intelligence Artificielle*, 24(3/2010) :357–390, June 2010.

[42] G. Piolle, Y. Demazeau, and J. Caelen. Privacy management in user-centred multi-agent systems. In G. O'Hare, M. O'Grady, O. Dikenelli, and A. Ricci, editors, *Proceedings of the 7th Annual International Workshop on Engineering Societies in the Agents World (ESAW'06)*, volume 4457/2007 of *LNCS*, pages 354–367, Dublin, Ireland, September 2006. Springer Verlag.

[43] A. Pnueli. The temporal logic of programs. In *Proceedings of the 18th IEEE Symposium on the Foundations of Computer Science (FOCS-77)*, pages 46–57, Providence, Rhode Island, USA, 1977. IEEE Computer Society Press.

[44] S. Riché, G. Brebner, and M. Gittler. Client-side profile storage. In *NETWORKING 2002 Workshops on Web Engineering and Peer-to-Peer Computing*, pages 127–133, Pisa, Italy, May 2002.

[45] G. H. von Wright. Deontic logic. *Mind*, 60 :1–15, 1951.

[46] M. Wooldridge. *Introduction to Multiagent Systems*, chapter Logics for Multiagent Systems, pages 267–302. John Wiley & Sons, 2001.

[47] World Wide Web Consortium. Platform for Privacy Preferences specification 1.1., 2006. `http://www.w3.org/P3P/`.