A Dynamic Logic for Privacy Compliance

Guillaume Aucher, Leendert van der Torre, Guido Boella

To cite this version:

Guillaume Aucher, Leendert van der Torre, Guido Boella. A Dynamic Logic for Privacy Compliance. Artificial Intelligence and Law, Springer Verlag, 2011, 19 (2-3), pp.187-231. 10.1007/s10506-011-9114-3 . inria-00628318v2

HAL Id: inria-00628318

https://hal.inria.fr/inria-00628318v2

Submitted on 1 Sep 2013

A Dynamic Logic for Privacy Compliance

Guillaume Aucher · Guido Boella · Leendert van der Torre

Abstract Knowledge based privacy policies are more declarative than traditional action based ones, because they specify only what is permitted or forbidden to know, and leave the derivation of the permitted actions to a security monitor. This inference problem is already non trivial with a static privacy policy, and becomes challenging when privacy policies can change over time. We therefore introduce a dynamic modal logic that permits not only to reason about permitted and forbidden knowledge to derive the permitted actions, but also to represent explicitly the declarative privacy policies together with their dynamics. The logic can be used to check both regulatory and behavioral compliance, respectively by checking that the permissions and obligations set up by the security monitor of an organization are not in conflict with the privacy policies, and by checking that these obligations are indeed enforced.

1 Introduction

The UN Declaration of Human Rights (United Nations General Assembly, 1948) stipulates that "No one shall be subjected to arbitrary interference with his *privacy*, family, home or correspondence, nor to attacks upon his honour and reputation" (our emphasis). However, there is no consensus on what *privacy* precisely means. In the nineteenth century, Warren and Brandeis (1890), quoting the judge Thomas M. Cooley, defined the right

This paper corrects a minor mistake in the Definition 7 of the article published in the Journal of Artificial Intelligence and Law (2011): the reflexivity of the knowledge accessibility relation was not preserved during an update with the promulgation operation.

G. Aucher
IRISA, INRIA, Université de Rennes 1, Campus de Beaulieu, 35042 Rennes, France
E-mail: guillaume.aucher@irisa.fr

G. Boella
Dipartimento di Informatica, Università di Torino, Cso Svizzera 185 10149 Torino
E-mail: guido@di.unito.it

L. van der Torre
Computer Science and Communications (CSC), University of Luxembourg, 6, rue Richard Coudenhove - Kalergi, L-1359 Luxembourg
E-mail: leon.vandertorre@uni.lu

to privacy as “the right to be let alone”. More recently, Westin (1968) defines privacy as the ability for people to determine for themselves when, how, and to what extent, information about them is communicated to others. Despite this lack of consensus in the legislation, numerous privacy policies have recently been developped. In 2001, 68% of the Direct Marketing Association member companies appoint Chief Privacy Officers responsible for privacy issues and policies. They design their internal activities and information practices to simultaneously serve their customers effectively and manage risks from disclosure of sensitive information. A much debated example of privacy policy in the United States is the Health Insurance Portability and Accountability Act (HIPAA) (Office for Civil Rights, 2003). Personal health information must be used to provide effective health care, but it must also be protected from indiscriminate sharing to respect the privacy of patients. Other examples are the Children’s Online Privacy Protection Act (COPPA) (Federal Trade Commission, 1998) for e-business, and the Gramm-Leach-Bliley Act (GLBA) (Federal Trade Commission, 1999) for financial institutions.

In general, privacy policies can be defined either in terms of permitted and forbidden *knowledge*, or in terms of permitted and forbidden *actions*. For example, it may be forbidden to know the medical data of a person, or it may be forbidden to disclose these data. Both of these approaches have their advantages and disadvantages. Implementing a privacy policy based on permitted and forbidden *actions* is relatively easy, since we can add a filter on the system checking the outgoing messages. Such a filter is an example of a security monitor. If the system attempts to send a forbidden message, then the security monitor blocks the sending of that message. However, the price to pay for this relatively straightforward implementation is that it is difficult to determine privacy policies using permitted and forbidden actions only, in the sense that it is difficult to decide which actions are permitted or forbidden so that a piece of information is not disclose. For example, it is a well known database problem that you may be able to find out my identity without asking for it explicitly, for example by asking a very detailed question (all the people who are born in Amsterdam on September 11 1986, who drive a blue Mercedes, and who are married to a person from Paris on November 9, 2009), or by combining a number of queries on a medical database (Sweeney, 2002). In this paper we are therefore interested in privacy policies expressed in terms of permitted and forbidden knowledge.

Expressing a privacy policy in terms of permitted and forbidden knowledge is relatively easy, since it lists the situations which should not occur. These situations are typically determined by the fact that it may not be permitted to know some sensitive information. In many cases it is more efficient or natural to specify that a given piece of information may not be known, than explicitly forbidding the different ways of communicating it. The policies are more declarative, more concise and therefore easier to understand by the user. They may also cover unforeseen sequences of actions leading to forbidden situation. However, implementing a privacy policy based on permitted and forbidden knowledge is relatively difficult, since the system has to reason about the relation between permitted knowledge and actions. The challenge is that the exchange of messages changes the knowledge, and the security monitor therefore needs to reason about these changes.

To express privacy policies in terms of permitted and forbidden knowledge, we use modal logic, since both knowledge and obligations (and permissions) are traditionally and naturally modeled in branches of modal logic called epistemic and deontic logic respectively. Cuppens introduced in 1993 a modal logic for a logical formalization of secrecy (Cuppens, 1993), and together with Demolombe he developed a logic for reasoning about confidentiality (Cuppens and Demolombe, 1996) and a modal logical framework for security policies (Cuppens and Demolombe, 1997). The logic models the knowledge of the users of the system, and allows the security monitor to reason about them. It expresses formulas such as ‘the user knows the address of someone’, and epistemic norms, i.e. norms regulating what is permitted to know. The security monitor is able to foresee the inferences that the users can do by combining their knowledge. For example, if the user knows street name, number, town and state of a person, then he knows his address. Moreover, since privacy policies are specified in terms of knowledge that the recipient of information is permitted/forbidden to have, we can represent violations. This is an advantage over privacy policy languages modeling norms as strict constraints that cannot be violated, because in some situations it is necessary to cope with violations. These violations can be due for example to occasional and unintentional disclosures, or to the creation of new more restrictive norms.

The main task of a security monitor reasoning about a situation given a privacy policy is to check compliance – regardless of whether these policies are expressed in terms of permitted and forbidden actions or permitted and forbidden knowledge. In our approach, to check compliance one has therefore to be able to derive the permitted, obligatory and forbidden actions in a given context, just like a decision maker needs to know whether his alternative actions do not violate norms and may therefore be subject to sanctions. In this paper, we further distinguish between regulatory compliance and behavioural compliance. Regulatory compliance checks whether the permissions and obligations set up by the security monitor of an organization (e.g., company, web-service . . .) are compliant with respect to the privacy policies set up by the law/policy makers. Behavioural compliance checks whether these very obligations and permissions are indeed enforced in the system by the security monitor of the organization.

Despite its strengths, the Cuppens-Demolombe logic cannot express whether the situation is (regulatory or behaviourally) compliant with respect to a privacy policy. The problem is that the logic can define privacy policies in terms of the permitted and forbidden knowledge of the resulting epistemic state of the recipient of information, but it cannot derive the permitted messages nor the obligatory messages by combining and reasoning on this knowledge. Our modal logic addresses these problems and extends the Cuppens-Demolombe logic with dynamic update operators inspired from the ones of dynamic epistemic logic (van Ditmarsch et al, 2007). These dynamic operators model both the dynamics of knowledge and of privacy policies. They can add or remove norms from the policy, and we add constants expressing whether the system is regulatorily and behaviourally compliant with a policy, i.e., there is no violation.

The paper is organized as follows. In Section 2, we describe the range of phenomena under study, and we give a number of examples to provide some intuitions. In Section 3, we introduce our Dynamic Epistemic Deontic Logic (DEDL). We start with the static part, defining epistemic norms and privacy policies, and we then add dynamics, defining permitted (and obligatory) messages and enforcements of privacy policies. In Section 4, we propose a new logic based on the logic of Section 3 which is more appropriate for checking compliance.

2 Our scenario of privacy policies

In this paper, we consider a single agent (Sender) communicating information from a knowledge base to another agent (Recipient), with the effect that the Recipient knows the information. The Sender is subject to privacy policies which restrict the messages he is permitted to send to the Recipient. The Sender is therefore a security monitor. We illustrate the distinction between norms of transmission of information and epistemic norms with an example:

Example 1 Consider a Sender s, e.g., a web server, which is subject to a privacy regulation: he should not communicate the address a of a person to the Recipient r. We could write this as a norm of transmission of information, regulating the sending of a message: $\neg P_s(\mathit{Send}\ a)$, which denotes the denial that the Sender sends message a. Instead, in an epistemic norm perspective, this prohibition can be derived from the prohibition for the Sender that the Recipient comes to know the address: $K_r a$. This is expressed by a deontic operator indexed by the Sender and having as content the ideal knowledge K_r of the Recipient: $\neg P_s K_r a$.

This distinction is bridged by modelling sending actions performed by the Sender which update the knowledge of the Recipient.

Example 2 The action of sending the message, $[\mathit{Send}\ a]$, expresses that the Sender sends to the Recipient the address a. The result of this action is that the Recipient knows a: $K_r a$. Since $K_r a$ is not permitted by the epistemic norm $\neg P_s K_r a$, the Sender during his decision process derives that also the action $[\mathit{Send}\ a]$ is not permitted: $\neg P_s(\mathit{Send}\ a)$. Analogously, all other possible actions leading to the forbidden epistemic state $K_r a$, if any, are prohibited too. For example, if the address is composed by street m, number n and town t such that $(m \wedge n \wedge t) \leftrightarrow a$, then the sequence of messages $[\mathit{Send}\ m][\mathit{Send}\ n][\mathit{Send}\ t]$ leads to the forbidden epistemic state $K_r a$.

Given the assumptions made in this paper, there is an asymmetry in the modeling of what the Sender and the Recipient know. In the current model we consider only one source of information, so the Sender's point of view is the objective one and can be represented as factual knowledge, e.g., we read a as the Sender knows the address. We assume also that the Sender never lies, so we talk about "knowledge" of the Recipient: the result of Sender's actions on the epistemic state of the Recipient is knowledge rather than belief: $K_r a$ implies a, i.e., that the Sender holds a as true. If instead we allowed the Sender to lie to protect some secrets (as, e.g., Bonatti et al (1995) do), then the result of the action of sending messages would be a mere belief of the Recipient: the result of $[\mathit{Send}\ a]$ would be that the Recipient believes a, but a - from the point of view of the Sender - would not follow from this.

A logical approach to privacy provides a natural solution to the so-called inference problem (Hinke, 1988), i.e., how further permissions propagate from permitted information:

Example 3 Assume it is prohibited to know the street where some person lives. Thus, it must be prohibited to know the address of this person. If $m \wedge n \wedge t \leftrightarrow a$, then $\neg P_s K_r m$ implies $\neg P_s K_r a$. Viceversa, if it is permitted to know the address, then it must be permitted to know the street. The same kind of reasoning is transferred at the level of norms of transmission of information. For example, $\neg P_s(Send\ m)$ implies $\neg P_s(Send\ a)$: if it is prohibited to send the name of the street, it is prohibited to send the entire address.

Note that to attribute knowledge to the Recipient, it is neither necessary to have user profiles nor to have any uncertainty. This stems from the assumption that the Sender is for the Recipient the only source of information concerning the knowledge base. The only knowledge that should be considered is the one derived from the past interactions between the two agents, i.e., the information already disclosed by the Sender. Assuming for simplicity that the Sender is rational and sends only information consistent with his previous messages, there is no need to introduce some kind of belief revision.

When the forbidden state is achieved by a sequence of messages, there is the possibility that each message of the sequence is permitted while the resulting state is prohibited: this is a new kind of the Chinese wall problem (Brewer and Nash, 1989), considered from the double perspective of both knowledge of the Recipient and messages by the Sender. A Chinese wall security policy is an example of aggregation exception. The exception is given by a system that maintains two pieces of information and a user is authorized to access one or the other, but not both.

Example 4 (Website example) Consider the information about websites contacted by a user (the Recipient), which are available on a server (the Sender) logfile. The list of websites for each user is clearly a sensitive information which he would not like to disclose. However, knowing which websites have been visited is a valuable information, for example, for the configuration of a firewall, or to make statistics. Thus it has become anonym by replacing the names of the users with numbers by means of a hashcode (h). So even if one knows the list of users one cannot understand who contacted which website. However, from the association between users and numbers and between numbers and websites the original information can be reconstructed. Therefore the mappings from the users to the numbers (c) and from the numbers to the websites (e) can be distributed individually but not altogether since their association would allow to reconstruct the mapping from the users to the websites they visited (v): $c \wedge e \rightarrow v$.

A solution to enforce this privacy policy could be to forbid the distribution of a mapping if the other one has been already distributed, using a language like the one proposed by Barth et al (2007), which is able to express policies about the flow of information referring to actions already performed. This solution, however, requires two rules corresponding to the possible permutations of messages. Moreover, this solution is not general, because there can be further ways of making the forbidden information available, for example by distributing the hash function h used. Expressing a flexible policy on all the alternative combinations of actions becomes soon unfeasible. Moreover, new ways of computing the forbidden information could be devised later, which would not be taken into account by the policy.

In this situation we have that it is permitted to know the individual pieces of information, but not what is implied by the conjunction of them:

$$P_sK_rc, P_sK_re, \neg P_sK_rv.$$

It states that it is permitted to know the mapping between users and numbers (P_sK_rc), it is permitted to know the mapping between numbers and websites visited (P_sK_re) but it is not permitted to know the mapping between users and their websites visited ($\neg P_sK_rv$). We have the same situation from the point of view of permissions concerning actions: it is permitted to send the messages c and e individually, but not their combination: $P_s(Send\ c) \wedge P_s(Send\ e)$ but $\neg P_s(Send\ (e \wedge c))$ otherwise the epistemic norm $\neg P_s K_r v$ would be violated. This means that after sending one of the two messages, the other one becomes prohibited: $[Send\ e]\neg P_s(Send\ c)$ and $[Send\ c]\ \neg P_s(Send\ e)$.

Some privacy policies prescribe not only which messages are forbidden, but also which messages are obligatory. We can lift these obligations at the level of epistemic norms:

Example 5 (Spyware Example) Assume that the list of websites mentioned in e and sent by Sender can disclose websites where it is possible to download software. If Sender knows that there is the risk that such software contains spyware (y), then the Recipient should 'know' it. This privacy policy is expressed by a single epistemic norm:

$$y \wedge K_r e \rightarrow O_s K_r y$$

We can generalize this kind of policies to situations where the Sender should inform the Recipient *whether* a piece of information has some property or not:

Example 6 When the knowledge base contains classified information and the Recipient has some specific role, then the Sender should inform the Recipient whether the message p he sent contains classified information (c) or not:

$$role(r) \wedge K_r p \rightarrow O_s(K_r c \vee K_r \neg c)$$

The policy does not contain in the antecedent the fact that Sender communicated p, but only that Recipient knows p: recall that we assume that the only source of information for the Recipient is the sender, so this epistemic state must be the result of a previous message by the Sender.

Note that in this example we have that the norm is applicable only in a context where the Recipient plays a given role. Indeed, as discussed by Barth et al (2007) in the theory of contextual integrity, privacy norms are relevant only in some context, usually defined by roles played by Sender and Recipient.

When introducing the new notions of permitted and obligatory epistemic states and the corresponding permissions and obligations to send a message, it is necessary to clarify what the notion of compliance becomes.

Example 7 A message is permitted ($P_s(Send\ a)$), if sending it does not lead to a non-compliant situation: i.e., if sending it does not violate some epistemic obligation. For example, sending the message a ($[Send\ a]$) leads to a state where $K_r a$ while at the same time $\neg P_s K_r a$ (i.e., $O_s \neg K_r a$). A message is obligatory ($O_s(Send\ a)$) when the Recipient does not already know the information ($\neg K_r a$), it is permitted to send the message ($P_s(Send\ a)$) and it is obligatory that the Recipient knows the content of the message ($O_s K_r a$)

Many privacy policy languages, like Barth et al (2006), view norms as strict constraints which cannot be violated. This reduces the flexibility since in some situations it is necessary to cope with violations.

Example 8 Assume that the Sender discloses some prohibited information (e.g., $\neg P_s K_r a$), and this leads to a non-compliant situation: $[Send\ a](K_r a \wedge \neg P_s K_r a)$. In this situation, the Sender is subject to further requirements, for example to make the Recipient know that the information is classified c: $K_r a \to O_s K_r c$.

The possibility of nesting formulas with epistemic and deontic modalities allows us to express meta-policies, i.e., policies concerning the disclosure of policies, as proposed for example by Bonatti et al (1995):

Example 9 Sometimes, informing the Recipient about the prohibition to send some information might lead him to infer something he should not know. For example, if the Recipient asks whether a person is a secret agent (p), replying "I cannot tell this to you" to the question makes the Recipient infer that the person is actually a secret agent, otherwise the answer would have been "no". To avoid this case, it should be prohibited to let the Recipient know the policy that knowing p is prohibited:

$$\neg P_s K_r \neg P_s K_r p$$

In contrast, if a policy is permitted to be known, it can even be communicated to the Recipient. If $P_s K_r P_s K_r p$ then it is permitted to send the message $P_s K_r p$: $P_s(Send\ P_s K_r p)$. This illustrates also that policies can be the content of messages.

Finally, policies have a dynamic character for several motivations. A major source of modification in policies is due to the internal IT system management, to new user requirements or to adapt the system against new possible security threats. Moreover, it is well known that law has a dynamic character since it is subject both to continuous change by the legislator and to the interpretation process of law in courts (see Boella et al (2010) for a formal model of interpretation). The two problems are interconnected, since changes of policies not due to changes in law must be assured to be compliant with laws and further changes may be due to ensure this compliance.

For these reasons in our model, we consider how to introduce new norms in policies composed by epistemic norms.

Example 10 In case of attack by some hacker, the privacy policies can be made more strict. For example, the Sender can decide to strengthen the privacy policy $\mathscr{P}$ of Example 4 to

$$P_s K_r c, \neg P_s K_r e, \neg P_s K_r v$$

where $P_s K_r e$ has been replaced by $\neg P_s K_r e$: it is now prohibited to disclose the mapping from numbers to visited web-sites. In the same way as we have message sending actions, we introduce promulgation actions which update the set of norms. The above new privacy policy can be enforced by the Sender through the update $[Prom\ \neg K_r e]$.

3 Dynamic epistemic deontic logic

The logic for privacy policy reasons about obligation, permission, knowledge, and information exchange. To deal with these notions altogether, we first extend in Section 3.1 the logic of Cuppens and Demolombe (1997), and in Section 3.2 we add dynamics.

3.1 'Static' privacy policies

3.1.1 Epistemic Deontic Logic (EDL)

We split our language into two kinds of formulas: circumstances and epistemic practitions. The former cannot be in the scope of an obligation operator O_s whereas the latter are always within the scope of a deontic operator O_s. Formulas of $\mathscr{L}_{EDL}$ are called circumstances and formulas of $\mathscr{L}^{\alpha}_{EDL}$ are called epistemic practitions. The formula $O_s\alpha$ reads as 'it is obligatory for the Sender that α.' The formula $P_s\alpha$ is an abbreviation for $\neg O_s\neg\alpha$ and reads as 'it is permitted for the Sender that α.' The formula $K_r\phi$ reads as 'the Recipient knows that ϕ.' Pure circumstances are circumstances without obligation operators $O_s\alpha$, we call them $\mathscr{L}_{EL}$. In other words, pure circumstances are just (standard) epistemic formulas. This duality between practitions and proposition is derived from Castañeda's well known approach to deontic logic, who observed the grammatical duality of expressions depending whether they are within or without the scope of deontic operators (Castañeda, 1981) (see the appendix for more details). This yields the following language.

Definition 1 (Syntax of $\mathscr{L}_{EDL}$) Let Φ^{ϕ} be a set of propositionnal letters. The language $\mathscr{L}_{EDL}$ is defined inductively as follows.

$$\mathscr{L}_{EDL} : \phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid K_r\phi \mid O_s\alpha$$
$$\mathscr{L}^{\alpha}_{EDL} : \alpha ::= K_r\phi \mid \neg\alpha \mid \alpha \wedge \alpha$$

where p ranges over Φ^{ϕ}.

We have that practitions are also epistemic propositions, in the sense that $\mathscr{L}^{\alpha}_{EDL} \subset \mathscr{L}_{EDL}$. The duality between practitions and epistemic propositions is illustrated by the following example.

Example 11 The following formulas are members of both $\mathscr{L}_{EDL}$ and $\mathscr{L}^{\alpha}_{EDL}$:

- K_ra: the receiver knows the address.
- $K_rK_ra \rightarrow K_rO_sK_ra$: if the Recipient knows that he knows the address, then he knows that it is obligatory for the Sender that he knows the address.

The following formulas are members of $\mathscr{L}_{EDL}$ but not $\mathscr{L}^{\alpha}_{EDL}$:

- $\neg P_sK_ra$: it is not permitted for the Sender that the Recipient knows the address.
- $y \wedge K_re \rightarrow O_sK_ry$: If Sender knows that there is the risk that software contains spyware (y), then the Recipient should 'know' it.
- $role(r) \wedge K_rp \rightarrow O_s(K_rc \vee K_r\neg c)$: If the Recipient has some specific role and Recipient knows p, then the Sender should inform the Recipient whether the message p he sent contains classified information (c) or not.
- $K_ra \rightarrow O_sK_rc$: if the Recipient knows the address, then the Sender is subject to further requirements, for example to make the Recipient know that the information is classified.

The following formulas are *not* members of $\mathscr{L}_{EDL}$:

- $O_sO_sK_rp$: there is no nesting of two operators O_s without an operator K_r in between.
- O_sp: only epistemic formulas are allowed in the scope of a deontic operator.

Consequently, the formula $O_s K_r p \rightarrow O_s p$ does not belong to the language $\mathscr{L}_{EDL}$. This formula is known as Åqvist's paradox (Åqvist, 1967).

Definition 2 (Semantics of $\mathscr{L}_{EDL}$) An *EDL-model* M is a tuple $M = (W, D, R, V)$, where W is a non-empty set of possible worlds, $R : W \rightarrow 2^W$ and $D : W \rightarrow 2^W$ are accessibility relations on W, D being serial, and R being reflexive[1], and $V : \Phi^\phi \rightarrow 2^W$ is a valuation. An *EDL-frame* is an *EDL*-model without valuation V. The truth conditions are defined inductively as follows.

$$\begin{array}{lll}
M, w \models p & \text{iff} & w \in V(p) \\
M, w \models \phi \wedge \psi & \text{iff} & M, w \models \phi \text{ and } M, w \models \psi \\
M, w \models \neg\phi & \text{iff} & \text{not } M, w \models \phi \\
M, w \models O_s \alpha & \text{iff} & \text{for all } v \in D(w), M, v \models \alpha. \\
M, w \models K_r \phi & \text{iff} & \text{for all } v \in R(w), M, v \models \phi
\end{array}$$

We write $M \models \phi$ when for all $w \in W$, $M, w \models \phi$. (M, w) is called a pointed *EDL*-model. If $\mathscr{P}$ is a set of formulas, we write $M, w \models \mathscr{P}$ when $M, w \models \phi$ for all $\phi \in \mathscr{P}$.

Theorem 1 (Soundness and completeness) *The logic L_{EDL} axiomatized by the following axiom schemes and inference rules is* sound and complete *for the language $\mathscr{L}_{EDL}$ with respect to the semantics of EDL-models. We write $\vdash \phi$ for $\phi \in \mathsf{L}_{EDL}$.*

$$\begin{array}{ll}
\mathsf{A}_1 & \textit{All propositional tautologies based on } \Phi^\phi \\
\mathsf{A}_2 & \vdash O_s \alpha \rightarrow P_s \alpha \\
\mathsf{A}_3 & \vdash K_r \phi \rightarrow \phi \\
\mathsf{A}_4 & \vdash O_s(\alpha \rightarrow \alpha') \rightarrow (O_s \alpha \rightarrow O_s \alpha') \\
\mathsf{A}_5 & \vdash K_r(\phi \rightarrow \phi') \rightarrow (K_r \phi \rightarrow K_r \phi') \\
\mathsf{R}_1 & \textit{If } \vdash \alpha \textit{ then } \vdash O_s \alpha \\
\mathsf{R}_2 & \textit{If } \vdash \phi \textit{ then } \vdash K_r \phi \\
\mathsf{R}_3 & \textit{If } \vdash \phi^* \rightarrow \psi^* \textit{ and } \vdash \phi^* \textit{ then } \vdash \psi^*
\end{array}$$

Proof It follows straightforwardly from the Sahlqvist correspondence theorem (Sahlqvist, 1975) because Axioms A_2 and A_3 are Sahlqvist formulas. QED

Axiom A_2 models the fact that an obligation for the Sender should always be permitted. Axiom A_3 models the fact that if the Recipient knows a fact then this fact must be true: it is an essential property of knowledge. The other axioms model the fact that obligations and knowledge are closed under logical consequences. We could perfectly add other conditions on accessibility relations and their corresponding axioms, such as knowledge introspection ($\vdash K_r \phi \rightarrow K_r K_r \phi$), and all the results of this paper would still hold. We remain unspecific about these epistemic issues because we prefer to focus in this paper on our contribution, which is of a different nature.

Let M be a finite *EDL*-model and ϕ be a formula of $\mathscr{L}_{DEDL}$. We define $||M||$, the *size* of M, to be the sum of the number of possible worlds in M and the number of pairs in R and D. We define $|\phi|$, the *size* of ϕ, to be the number of symbols in ϕ.[2] For a given function

[1] An accessibility relation R is serial if and only if $R(w) \neq \emptyset$ for all worlds w. An accessibility relation R is reflexive if and only if $w \in R(w)$ for all $w \in W$. See Blackburn et al (2001) for details.

[2] Formally, $|\phi|$ is defined inductively as follows: $|p| = 1$, $|\neg\phi| = |K_r \phi| = 1 + |\phi|$, $|\phi \wedge \phi'| = 1 + |\phi| + |\phi'|$, $|O_s \alpha| = 1 + |\alpha|$, $|[Send\ \psi]\phi| = 1 + |\psi| + |\phi|$, $|[Prom\ \alpha]\phi| = 1 + |\alpha| + |\phi|$.

$g(n)$, we denote by $O(g(n))$ the set of functions

$$O(g(n)) = \left\{ f(n) \;\middle|\; \text{there are constants } c \text{ and } n_0 \text{ such that } O \leq f(n) \leq cg(n) \text{ for all } n \geq n_0 \right\}.$$

Thus, for example, when we say that the running time of model checking whether $M,w \models \phi$ is $O(||M|| \times |\phi|)$, this means that there is some constant c, independant of the EDL-model M and the formula ϕ, such that for all pointed EDL-models (M,w) and formulas ϕ, the time to check if $M,w \models \phi$ is at most $c \times ||M|| \times |\phi|$.

Theorem 2 (Decidability and complexity) *The satisfiability problem for $\mathscr{L}_{EDL}$ is decidable and PSPACE-complete. There is an algorithm that, given a finite pointed EDL-model (M,w) and a formula $\phi \in \mathscr{L}_{EDL}$, determines, in time $O(||M|| \times |\phi|)$, whether $M,w \models \phi$.*

Proof To prove decidability, one shows that L_{EDL} has the finite model property by adapting the selection method (Blackburn et al, 2001). The proof that the satisfiability problem is PSPACE can easily be adapted from Halpern and Moses (1992). The algorithm for the model checking problem is the same as in Fagin et al (1995). QED

3.1.2 Privacy policies and compliance in EDL.

As discussed by Barth et al (2007) in the theory of contextual integrity, privacy norms are relevant only in some context, usually defined by roles played by Sender and Recipient. This leads us to define the following notions.

Definition 3 (Epistemic norm and privacy policy) An *epistemic norm* is a formula of the form $i \rightarrow O_s\alpha$ or $i \rightarrow P_s\alpha$ where i is a pure circumstance and α is an epistemic practition. The set of epistemic norms is written $\mathscr{E}\mathscr{N}$. A *privacy policy* $\mathscr{P}$ is a L_{EDL}-consistent set of epistemic norms.

Example 12 The following formulas of $\mathscr{L}_{EDL}$ can be elements of a privacy policy:

- $O_s\neg K_r a$: it is not permitted for the Sender that the Recipient knows the address.
- $y \wedge K_r e \rightarrow O_s K_r y$: If Sender knows that there is the risk that software contains spyware (y), then the Recipient should 'know' it.
- $role(r) \wedge K_r p \rightarrow O_s(K_r c \vee K_r \neg c)$: If the Recipient has some specific role and Recipient knows p, then the Sender should inform the Recipient whether the message p he sent contains classified information (c) or not.
- $K_r a \rightarrow O_s K_r c$: if the Recipient knows the address, then the Sender is subject to further requirements, for example to make the Recipient know that the information is classified.

Note that obligations and permissions concern the knowledge of the Recipient. This fact should not let the reader think that a privacy policy concerns the behavior of the Recipient. Indeed, the beliefs of the Recipient are only modified by actions of the Sender, so these policies regulate the behavior of the Sender who might disclose information or not to the Recipient depending on whether or not this disclosure is in conflict with the privacy policy. The following formulas of $\mathscr{L}_{EDL}$ cannot be elements of a privacy policy:

- $K_r a$: no deontic operator
- $K_r K_r a \rightarrow K_r O_s K_r a$: deontic operator preceded by a knowledge operator

- $K_r O_s K_r y \wedge K_r e \rightarrow O_s K_r y$: condition is not a pure circumstance since it contains a deontic operator

Privacy policies are typically set up by a law/policy maker and are imposed to the security monitor (Sender), although they might also be set up by the Sender himself/herself. They should be enforced in any case.

The set of epistemic norms is not necessarily complete. As a result, the Sender can perfectly add other epistemic norms as long as they are consistent with the privacy policy, depending on the particular situation at stake. This leads us to define the following notions of permissive and restrictive privacy policies. Intuitively, a restrictive privacy policy is a policy where only the permissions of the security policies hold, everything else being forbidden. A permissive privacy policy is a policy where only the prohibitions of the security policy hold, everything else being permitted. These definitions are similar with the definitions of permissive and restrictive approach of Cuppens and Demolombe (1997).

Definition 4 (Permissive and restrictive privacy policy) Let $\mathscr{P}$ be a privacy policy. The set $\mathscr{E}(M,w) = \{\phi \in \mathscr{L}_{EL} \mid M,w \models \phi\}$ represents the epistemic state of the Recipient.

- The privacy policy $\mathscr{P}$ is *restrictive* if for all *EDL*-models (M,w), if $\mathscr{E}(M,w) \cup \mathscr{P} \nvdash P_s\alpha$, then $M,w \models \neg P_s\alpha$.
- The privacy policy $\mathscr{P}$ is *permissive* if for all *EDL*-models (M,w), if $\mathscr{E}(M,w) \cup \mathscr{P} \nvdash \neg P_s\alpha$, then $M,w \models P_s\alpha$.

Note that specifying whether a privacy policy $\mathscr{P}$ is restrictive or permissive specifies completely what is permitted and forbidden to know for the Recipient in the pointed *EDL*-model (M,w). As we said, in the general case, the privacy policy $\mathscr{P}$ does not specify all the obligations that should hold in a situation (M,w).

We therefore define four notions of compliance. The first notion of compliance, called *regulatory compliance*, checks whether the permissions and obligations set up by the security monitor (Sender) of an organization are consistent with the current privacy policy $\mathscr{P}$ set up by the law/policy maker. The second notion of compliance, called *behavioural compliance*, checks whether the obligations $O_s\alpha$ strictly following from the privacy policy $\mathscr{P}$ given the epistemic state $\mathscr{E}(M,w)$ are enforced. The third notion of compliance, simply called *compliance*, corresponds to the conjunction of the regulatory and behavioural compliance. The fourth notion of compliance, called *strong compliance*, checks regulatory compliance and whether *all* the obligations set up by the security monitor (Sender) are enforced.

Definition 5 (Compliance) Let (M,w) be a pointed *EDL*-model and $\mathscr{P}$ a privacy policy.

- The situation (M,w) is *regulatory compliant* with respect to the privacy policy $\mathscr{P}$ if and only if $M,w \models O_s\alpha$ for all $i \rightarrow O_s\alpha \in \mathscr{P}$ and $M,w \models P_s\alpha$ for all $i \rightarrow P_s\alpha \in \mathscr{P}$.
- The situation (M,w) is *behaviourally compliant* with respect to the privacy policy $\mathscr{P}$ if and only if $M,w \models i \rightarrow \alpha$ for all $i \rightarrow O_s\alpha \in \mathscr{P}$.
- The situation (M,w) is *compliant* with respect to the privacy policy $\mathscr{P}$ if and only if it is regulatory and behaviourally compliant with respect to $\mathscr{P}$.
- The situation (M,w) is *strongly compliant* with respect to $\mathscr{P}$ if and only if it is regulatory compliant with respect to $\mathscr{P}$ and $M,w \models O_s\alpha \rightarrow \alpha$ for all $\alpha \in \mathscr{L}^{\alpha}_{EDL}$.

When the privacy policy $\mathscr{P}$ is finite, we can express with a single formula whether the current situation (M,w) is regulatory compliant, behaviourally compliant and compliant with respect to $\mathscr{P}$.

Proposition 1 *Let $\mathscr{P}$ be a finite privacy policy. Let $RegComp = \bigwedge_{i \to O_s\alpha \in \mathscr{P}} O_s\alpha \wedge \bigwedge_{i \to P_s\alpha} P_s\alpha$, $BehComp = \bigwedge_{i \to O_s\alpha \in \mathscr{P}} (i \to \alpha)$ and $Comp = RegComp \wedge BehComp$. Then,*

- *(M,w) is regulatory compliant with respect to $\mathscr{P}$ if and only if $M,w \models RegComp$;*
- *(M,w) is behaviourally compliant with respect to $\mathscr{P}$ if and only if $M,w \models BehComp$;*
- *(M,w) is compliant with respect to $\mathscr{P}$ if and only if $M,w \models Comp$.*

The following proposition shows that for permissive privacy policies compliance is the same as strong compliance. It also gives a semantic counterpart to the syntactic notion of strong compliance: an epistemic state (represented by $(M,R(w))$) is strongly compliant if there exists a corresponding ideal epistemic state (represented by $(M,R(v))$ for some $v \in D(w)$) containing the same information (i.e. bisimilar[3]).

Proposition 2 *Let (M,w) be a finite pointed EDL-model and $\mathscr{P}$ a privacy policy.*

- *If $\mathscr{P}$ is permissive, then (M,w) is compliant with respect to $\mathscr{P}$ if and only if (M,w) is strongly compliant with respect to $\mathscr{P}$.*
- *The situation (M,w) is strongly compliant with respect to $\mathscr{P}$ if and only if there exists $v \in D(w)$ such that $(M,R(w))$ and $(M,R(v))$ are bisimilar.*

Proof

- The proof of the first item stems from the fact that if $\mathscr{P}$ is permissive, then for all pointed *EDL*-models (M,w), $M,w \models O_s\alpha$ for some $\alpha \in \mathscr{L}^{\alpha}_{EDL}$ if and only if $\mathscr{E}(M,w) \cup \mathscr{P} \vdash O_s\alpha$. The right to left direction trivially trivially holds. Assume towards a contradiction that $M,w \models O_s\alpha$ and $\mathscr{E}(M,w) \cup \mathscr{P} \nvdash O_s\alpha$. Then, because $\mathscr{P}$ is permissive $M,w \models \neg O_s\alpha$, which is impossible.
- Let (M,w) be a finite pointed *EDL*-model strongly compliant with respect to $\mathscr{P}$. Assume towards a contradiction that for all $v \in D(w)$, $R(w)$ and $R(v)$ are not bisimilar. Then there is $v' \in R(v)$ such that for all $w' \in R(w)$, (M,v') and (M,w') are not bisimilar. Therefore, because M is finite, there is $\phi^v \in \mathscr{L}_{EDL}$ such that $M,v' \models \phi^v$ and $M,w' \models K_r\neg\phi^v$. Then for all $v \in D(w)$, $M,v \models \neg K_r\neg\phi^v$ and $M,w \models K_r\neg\phi^v$. Hence, $M,w \models O_s\left(\bigvee_{v \in D(w)} \neg K_r\neg\phi^v\right)$ and $M,w \models \bigwedge_{v\in D(w)} K_r\neg\phi^v$. Let $\alpha = \bigvee_{v\in D(w)} \neg K_r\neg\phi^v$, we

[3] Two pointed models (M,v) and (M',v') are *bisimilar* if there is a relation Z on $W \times W'$ such that vZv' and satisfying the following conditions:

Base:	if wZw', then for all $p \in \Phi^{\phi}$, $w \in V(p)$ iff $w' \in V'(p)$;
Forth R:	if wZw' and $u \in R(w)$, then there is $u' \in R(w')$ such that uZu';
Back R:	if wZw' and $u' \in R(w')$, then there is $u \in R(w)$ such that uZu';
Forth D:	if wZw' and $u \in D(w)$, then there is $u' \in D(w')$ such that uZu';
Back D:	if wZw' and $u' \in D(w')$, then there is $u \in D(w)$ such that uZu'.

If two pointed Kripke models are bisimilar then the formulas true at these two pointed models are the same, i.e. then contain the same information (see Blackburn et al (2001) for more details). Two multi-pointed models (M,S) and (M',S'), where $S \subseteq M$ and $S' \subseteq M'$, are *bisimilar* if for all $w \in S$ there is $w' \in S'$ such that (M,w) and (M',w') are bisimilar, and for all $w' \in S'$ there is $w \in S$ such that (M,w) and (M',w') are bisimilar.

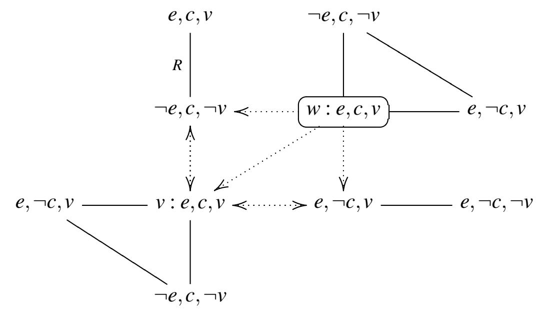

Fig. 1 Website example.

then have that $M,w \models O_s\alpha \wedge \neg\alpha$. This contradicts the assumption of strong compliance.

Let (M,w) be a finite pointed *EDL*-model regulatory compliant with respect to $\mathscr{P}$, and assume that there exists $v \in D(w)$ such that $(M,R(w))$ and $(M,R(v))$ are bisimilar. Let $\alpha \in \mathscr{L}^{\alpha}_{EDL}$, then there are $\alpha^i_0, \alpha^i_1, \ldots, \alpha^i_{n_i} \in \mathscr{L}^{\alpha}_{EDL}$ such that $\alpha \leftrightarrow \bigvee_{i\in\{1,\ldots,k\}} (K_r\alpha^i_0 \wedge \neg K_r\neg\alpha^i_1 \wedge \ldots \wedge \neg K_r\neg\alpha^i_n) \in \mathsf{L}_{EDL}$ by definition of $\mathscr{L}^{\alpha}_{EDL}$. Assume that $M,w \models \alpha$. Then there is $i \in \{1,\ldots,k\}$ such that $M,w \models K_r\alpha^i_0 \wedge \neg K_r\neg\alpha^i_1 \wedge \ldots \wedge \neg K_r\neg\alpha^i_{n_i}$. Because $(M,R(w))$ and $(M,R(v))$ are bisimilar, we have that $M,v \models K_r\alpha^i_0 \wedge \neg K_r\neg\alpha^i_1 \wedge \ldots \wedge \neg K_r\neg\alpha^i_{n_i}$. Indeed, assume towards a contradiction that $M,v \models \neg K_r\alpha^i_0$. Then there is $v' \in R(v)$ such that $M,v' \models \neg\alpha^i_0$. Therefore, by our definition of bisimilarity, there is $w' \in R(w)$ such that $M,w' \models \neg\alpha^i_0$, which contradicts the fact that $M,w \models K_r\alpha^i_0$. The same reasoning holds if we assume that $M,v \models K_r\neg\alpha^i_j$. So, $M,v \models \alpha$, and hence $M,w \models P\alpha$. Therefore, we have proved that for all $\alpha \in \mathscr{L}^{\alpha}_{EDL}$, $M,w \models \alpha \rightarrow P_s\alpha$, i.e., for all $\alpha \in \mathscr{L}^{\alpha}_{EDL}$, $M,w \models O_s\alpha \rightarrow \alpha$. So, (M,w) is strongly compliant with respect to $\mathscr{P}$.

QED

Example 13 (Website example continued) Consider Example 4, where we have the mappings from the users to the numbers (c) and from the numbers to the websites (e), the related mapping from the users to the websites they visited (v) such that $c \wedge e \rightarrow v$. We express the *privacy policy* $\mathscr{P}_1$ as:

$$\mathscr{P}_1 = \{P_sK_rc, P_sK_re, \neg P_sK_rv\}$$

It states that it is permitted to know the mapping between users and numbers (P_sK_rc), it is permitted to know the mapping between numbers and websites visited (P_sK_re) but it is not permitted to know the mapping between users and their websites visited ($\neg P_sK_rv$). In the pointed *EDL*-model (M,w) of Figure 1, the accessibility relation R is represented by plain lines and the accessibility relation D is represented by dashed lines. So for example, when we have a plain line between two possible worlds w and v, it means that $w \in R(v)$ and $v \in R(w)$, and when we have a dashed arrow from a possible world w to another possible world v, it means that $v \in D(w)$. Even if we do not represent it in the figure, we also have that $v \in R(v)$ for all worlds $v \in M$, and $v \in D(v)$ for all worlds $v \in M - \{w\}$. Besides, it holds

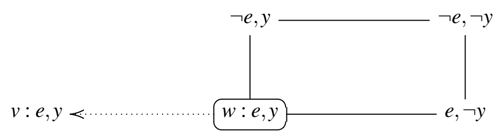

Fig. 2 Spyware example.

that $M \models c \wedge e \rightarrow v$. This pointed *EDL*-model (M,w) represents semantically a situation which is *compliant* with respect to the privacy policy $\mathscr{P}_1$. Indeed, $M,w \models \mathscr{P}_1$, so the situation is regulatory compliant with respect to $\mathscr{P}_1$. Moreover, the prohibition $\neg P_s K_r v$, or equivalently the obligation $O_s \neg K_r v$, of the privacy policy $\mathscr{P}_1$ is enforced because $M,w \models \neg K_r v$. So the situation (M,w) is also behaviourally compliant. In fact the situation (M,w) is also *strongly* compliant because, by application of Proposition 2, $(M,R(w))$ is bisimilar to $(M,R(v))$ and $v \in D(w)$.

Example 14 (Spyware example) Consider a situation where e is the list of websites mentioned and y is the fact that websites might contain risky software. The privacy policy is expressed by a unique epistemic norm:

$$\mathscr{P}_2 = \{y \wedge K_r e \rightarrow O_s K_r y\}$$

It states that if the Recipient knows a list of websites ($K_r e$) which might contain some risky softwares (y), then the Recipient should know that some of these websites might contain some risky softwares ($O_s K_r y$). Note that the condition of this epistemic norm contains an epistemic formula. In Figure 2 is depicted a situation compliant with this privacy policy. Like in Figure 1, the accessibility relation R in this pointed *EDL*-model (M,w) is represented by plain lines and the accessibility relation D is represented by dashed lines. So for example, when we have a plain line between two possible worlds w and v, it means that $w \in R(v)$ and $v \in R(w)$, and the dashed arrow stemming from the possible world w to the possible world v means that $v \in D(w)$. Reflexive arrows are omitted, which means in this *EDL*-model that for all $v \in M$, we have $v \in R(v)$ and for all $v \in M - \{w\}$, $\{v\} = D(v)$. The situation is compliant with respect to the privacy policy $\mathscr{P}_2$. Indeed, the situation is regulatory compliant because $M,w \models \mathscr{P}_2$. It is also behaviourally compliant, because $M,w \models y \wedge K_r e \rightarrow K_r y$ since $M,w \models \neg(y \wedge K_r e)$. However, it is not strongly compliant because (M,w) is not bisimilar to (M,v).

We can generalize this kind of policies to stronger policies where the Sender has to inform the Recipient *whether* a piece of information has some property or not as in Example 6.

3.2 The dynamic turn

3.2.1 Dynamic Epistemic Deontic Logic (DEDL)

We now add dynamics to the picture by means of messages sent to the Recipient and by means of promulgations that change the obligations of the Sender. These actions affect the situation respectively in two ways: either they affect the epistemic realm (represented in an *EDL*-model by the relation R) or they affect the normative realm (represented in an *EDL*-model by the relation D). This leads us to enrich the language $\mathscr{L}_{EDL}$ with two dynamic operators $[Send\ \phi]$ and $[Prom\ \alpha]$, yielding the language $\mathscr{L}_{DEDL}$. The formula $[Send\ \psi]\phi$ reads as 'after the Recipient learns ψ, ϕ holds', and $[Prom\ \alpha]\phi$ reads as 'after the Sender promulgates α, ϕ holds'.

Definition 6 (Syntax of $\mathscr{L}_{DEDL}$) The language $\mathscr{L}_{DEDL}$ is defined inductively as follows:

$$\mathscr{L}_{DEDL} : \phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid K_r\phi \mid O_s\alpha \mid [Send\ \phi]\phi \mid [Prom\ \alpha]\phi$$
$$\mathscr{L}^{\alpha}_{DEDL} : \alpha ::= K_r\phi \mid \neg\alpha \mid \alpha \wedge \alpha \mid [Send\ \phi]\alpha \mid [Prom\ \alpha]\alpha$$

where p ranges over Φ^{ϕ}.

We also have that $\mathscr{L}^{\alpha}_{DEDL} \subset \mathscr{L}_{DEDL}$, just as we had $\mathscr{L}^{\alpha}_{EDL} \subset \mathscr{L}_{EDL}$.

Example 15 The following formulas are members of $\mathscr{L}_{DEDL}$:

- $[Send\ a]K_r a$ The result of the Sender sending to the Recipient the address a is that the Recipient knows a.
- $[Send\ a](K_r a \wedge \neg P_s K_r a)$: the Sender discloses some prohibited information a (because $\neg P_s K_r a$), and this leads to a non (strongly) compliant situation.
- $[Prom\ \neg K_r e](P_s K_r c \wedge \neg P_s K_r e \wedge \neg P_s K_r v)$: the Sender enforced the new privacy policy $\mathscr{P} = \{P_s K_r c, \neg P_s K_r e, \neg P_s K_r v\}$.

The semantics of these dynamic operators is defined below, it is inspired by the logical framework of Baltag et al (1998); Baltag and Moss (2004).

In the multi-agent setting of Baltag and Moss (2004), one is interested in modeling the effect of (epistemic) actions on the beliefs of several agents. The perception of the agents concerning these actions when they occur is represented by action models. An example of action model is the one for private announcement whereby an agent B learns ψ while the other agent A believes nothing is happening. The action model of this private annoucement is the same as the EDL-frame A_ψ of Figure 3, except that the accessibility relation R has to be replaced by the epistemic accessibility relation of agent B and that the accessibility relation D has to be replaced by the epistemic accessibility relation of agent A. As a result of this private announcement, the beliefs of agent B are updated by ψ, whereas the beliefs of the agent A remain unchanged. This update is captured formally in Baltag and Moss (2004) by an update product $\otimes$ taking as argument an epistemic model and an action model, and yielding a new epistemic model. The definition of their update poduct $\otimes$ is formally the same as our definition of the update products $\oplus$ and $\odot$ below; the only difference here is our intuitive interpretation of the action model and the update product. As a result of the update $\oplus$ by the EDL-frame A_ψ, the knowledge of the Recipient is updated by ψ whereas the obligations and permissions of the Sender remain unchanged. Dually, as a result of the update $\odot$ by the action model A_α, the obligations of the Sender are updated by α whereas the knowledge of the Recipient remains unchanged. To this, we add the condition that if the information sent to the Recipient is not true then it is ignored, and the condition that if the norm to be promulgated is not true then it is also ignored.

Definition 7 (Semantics of $\mathscr{L}_{DEDL}$) Let $M = (W, D, R, V, w)$ be a pointed EDL-model, $\psi \in \mathscr{L}_{EDL}$ and $\alpha \in \mathscr{L}^{\alpha}_{EDL}$. Let $A_\psi = (W_\psi, D_\psi, R_\psi)$ be the EDL-frame depicted in Figure 3 and defined by $W_\psi = \{w_\psi, w_\top\}$, $D_\psi = \{(w_\psi, w_\top), (w_\top, w_\top)\}$ and $R = \{(w_\psi, w_\psi), (w_\top, w_\top)\}$. Let $A_\alpha = (W_\alpha, D_\alpha, R_\alpha)$ be the EDL-frame depicted in Figure 3 and defined by $W_\alpha = \{w_\alpha, w_\top\}$, $D_\alpha = \{(w_\alpha, w_\alpha), (w_\top, w_\top)\}$ and $R = \{(w_\alpha, w_\top), (w_\top, w_\top)\}$. We define the EDL-models $(M, w) \oplus \psi$ and $(M, w) \odot \alpha$ as follows.

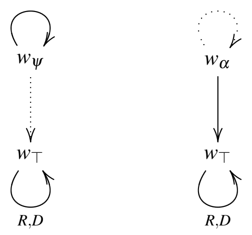

Fig. 3 *EDL*-frames A_ψ (left) and A_α (right).

- If $M,w \not\models \psi$ then $(M,w) \oplus \psi = (M,w)$, otherwise $(M,w) \oplus \psi = (M,w) \otimes A_\psi$.
- If $M,w \not\models \alpha$ then $(M,w) \odot \alpha = (M,w)$, otherwise $(M,w) \odot \alpha = (M,w) \otimes A_\alpha$.

Let (M,w) be a pointed *EDL*-model and $A_{\phi^*} = (W_{\phi^*}, D_{\phi^*}, R_{\phi^*})$ be an *EDL*-frame of the form A_ψ or A_α. The update product $(M,w) \otimes A_{\phi^*} = (W', D', R', V')$ is defined as follows.

- $W' = \{(w, w_{\phi^*}), (v, w_\top) \mid w, v \in W, M, w \models \phi^*\}$;
- $(v, v_*) \in D'(w, w_*)$ iff $v \in D(w)$ and $v_* \in D_{\phi^*}(w_*)$;
- $(v, v_*) \in R'(w, w_*)$ iff $v \in R(w)$ and $v_* \in R_{\phi^*}(w_*)$;
- $(w, w_*) \in V'(p)$ iff $w \in V(p)$.

The truth conditions of the dynamic operators $[Send\ \psi]$ and $[Prom\ \alpha]$ are defined as follows.

$$\begin{array}{lcl} M,w \models [Send\ \psi]\phi & \text{iff} & M \oplus \psi, w_\oplus \models \phi \\ M,w \models [Prom\ \alpha]\phi & \text{iff} & M \odot \alpha, w_\odot \models \phi. \end{array}$$

where $w_\oplus = (w, w_\psi)$ if $M,w \models \psi$, and $w_\oplus = w$ otherwise; and $w_\odot = (w, w_\alpha)$ if $M,w \models \alpha$, and $w_\odot = w$ otherwise.

Example 16 In Figure 4 is given an example of update with a formula of $\mathscr{L}_{EDL}$ (i.e. a circumstance) and in Figure 5 is given an example of update with a formula of $\mathscr{L}^\alpha_{EDL}$ (i.e. an epistemic practition). As in the previous representation of Figure 1, we omit in all these *EDL*-model the accessibility relations $s \in R(s)$, for all $s \in M$, and $s \in D(s)$ for all $s \in M - \{w\}$.

In Figure 4, the update by e just removes the world accessible from w by R where e does not hold, namely the world v. The rest of the model remains unchanged because all the other worlds are accessible from w by a deontic accessibility relation D.

In Figure 5, the update by $\neg K_r e$ just removes the world accessible from w by D where $\neg K_r e$ does not hold, namely the worlds u and t. The rest of the model remains unchanged because all the other worlds are either accessible from w via a world accessible by the relation D and making $\neg K_r e$ true, or they are accessible from w by the accessibility relation R.

Theorem 3 (Soundness and completeness) *The logic* L_{DEDL} *axiomatized by the following axiom schemes and inference rules is* sound and complete *for* $\mathscr{L}_{DEDL}$ *with respect to the semantics of EDL-models. The symbol* $\Box$ *below stands either for* $[Send\ \psi]$ *or* $[Prom\ \alpha]$.

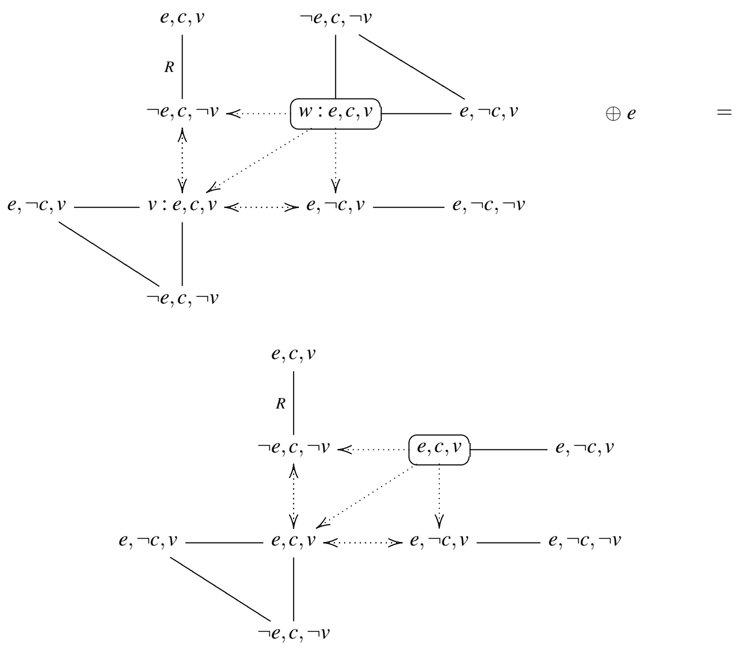

Fig. 4 Website example of Figure 1 updated by e.

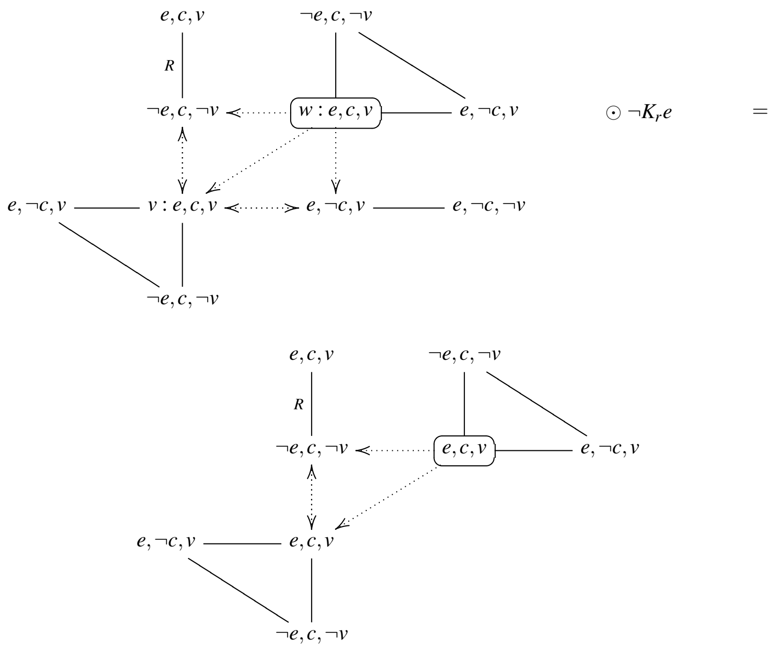

Fig. 5 Website example of Figure 1 updated by the epistemic practition $\neg K_r e$.

We write $\vdash \phi$ *for* $\phi \in \mathsf{L}_{DEDL}$.

L_{EDL}	*All the axiom schemes and inference rules of* L_{EDL}
A_6	$\vdash \psi \to ([Send\ \psi]K_r\phi \leftrightarrow K_r(\psi \to [Send\ \psi]\phi))$
A_7	$\vdash \neg\psi \to ([Send\ \psi]\phi \leftrightarrow \phi)$
A_8	$\vdash [Send\ \psi]O_s\alpha \leftrightarrow O_s\alpha$
A_9	$\vdash [Prom\ \alpha]K_r\phi \leftrightarrow K_r\phi$
A_{10}	$\vdash \alpha \to ([Prom\ \alpha]O_s\alpha' \leftrightarrow O_s(\alpha \to \alpha'))$
A_{11}	$\vdash \neg\alpha \to ([Prom\ \alpha]\phi \leftrightarrow \phi)$
A_{12}	$\vdash \Box p \leftrightarrow p$
A_{13}	$\vdash \Box\neg\phi^* \leftrightarrow \neg\Box\phi^*$
A_{14}	$\vdash \Box(\phi^* \to \psi^*) \to (\Box\phi^* \to \Box\psi^*)$
R_4	*If* $\vdash \phi^*$ *then* $\vdash \Box\phi^*$

Proof We first prove a lemma.

Lemma 1 *For all* $\phi \in \mathscr{L}_{DEDL}$ *there is* $\phi' \in \mathscr{L}_{EDL}$ *such that* $\vdash \phi \leftrightarrow \phi'$. *For all* $\alpha \in \mathscr{L}^{\alpha}_{DEDL}$ *there is* $\alpha' \in \mathscr{L}^{\alpha}_{EDL}$ *such that* $\vdash \alpha \leftrightarrow \alpha'$.

Proof of Lemma First, note that if ψ is a formula with dynamic operator then one shows by induction on ψ using A_6 to A_{14} that $\Box\psi$ is provably equivalent to a formula ψ' without dynamic operator. Now, if ϕ is an arbitrary formula with n dynamic operators, it has a subformula of the form $\Box\psi$ where ψ is without dynamic operators which is equivalent to a formula ψ' without dynamic operators. So we just substitute $\Box\psi$ by ψ' in ϕ and we get a provably equivalent formula thanks to A_{14} and R_4 with $n-1$ dynamic operators. We then iterate the process. QED

As usual in dynamic epistemic logic, we use the previous key lemma to prove the theorem. The soundness part is routine. Let $\phi \in \mathscr{L}_{DEDL}$ such that $\models \phi$. Then there is $\phi' \in \mathscr{L}_{EDL}$ such that $\vdash \phi \leftrightarrow \phi'$ by Lemma 1, and therefore $\models \phi \leftrightarrow \phi'$ by soundness. But $\vdash \phi'$ by Theorem 1, so $\vdash \phi$ as well. QED

Axioms A_9 to A_{11} model the fact that the operation of promulgation does not change the knowledge of the Recipient but only the obligations and permissions of the Sender. Axioms A_6 to A_8 model the fact that sending a message changes only the actual knowledge of the Recipient. In particular, if we focus on the propositional knowledge of the Recipient, then the combination of Axioms A_6 and A_7 boils down to

$$\vdash [Send\ \psi]K_r\phi \leftrightarrow ((\psi \to K_r(\psi \to \phi)) \wedge (\neg\psi \to K_r\phi)) \tag{1}$$

In other words, in this propositional case, if the message sent is not true then the knowledge of the Recipient remains the same as before the sending, so the message is just ignored. But if the message sent is true, then the knowledge of the Recipient is just conditionalized with this new message. In particular, we can derive the following theorem for all propositional formula ψ:

$$\vdash \psi \to [Send\ \psi]K_r\psi \tag{2}$$

This theorem means that after the Sender sends any truthful message to the Recipient, the Recipient knows this message. The same line of reasoning holds for Axiom A_{10} and A_{11}: if the imperative α is not permitted then it is just ignored, but if α is permitted then the obligations of the Sender are conditionalized by this imperative. These axioms entail that sending a message does not change the privacy policy (as a consequence of A_8) in case the antecedent of epistemic norms is propositionnal. They also entail that sending a message does not change regulatory compliance. This is as expected: regulatory compliance checks that the obligations and permissions set up by an organization (e.g. web-service, institution) are compliant with respect to the privacy policy set up by a law/policy maker. Therefore, this compliance should be independent from the context and the particular state of the organization at a particular point in time, and therefore also independent from the knowledge of the recipient at a particular point in time. The following proposition illustrates it.

Proposition 3 *For all epistemic norms $\chi = i \to O_s\alpha$ or $\chi = i \to P_s\alpha$ where i is propositionnal, and for all $\psi \in \mathscr{L}_{DEDL}$, we have that*

$$\vdash \chi \to [Send\ \psi]\chi \tag{3}$$

Besides,

$$\vdash RegComp \leftrightarrow [Send\ \psi]RegComp \tag{4}$$

Proof We prove this property using the axioms of L_{DEDL}.

1	$\vdash \chi \to \chi$	
2	$\vdash \chi \to (i \to O_s\alpha)$	
3	$\vdash \chi \to ([Send\ \psi]i \to [Send\ \psi]O_s\alpha)$	by Axioms A_8, A_{12} and A_{13}
4	$\vdash \chi \to (\neg[Send\ \psi]i \vee [Send\ \psi]O_s\alpha)$	
5	$\vdash \chi \to (\neg[Send\ \psi]\neg(\neg i) \vee [Send\ \psi]O_s\alpha)$	
6	$\vdash \chi \to (\neg[Send\ \psi]\neg(\neg i) \vee \neg[Send\ \psi]\neg O_s\alpha)$	by Axiom A_{13}
7	$\vdash \chi \to \neg[Send\ \psi]\neg(\neg i \vee O_s\alpha)$	by normality of $[Send\ \psi]$
8	$\vdash \chi \to [Send\ \psi](\neg i \vee O_s\alpha)$	by Axiom A_{13}
7	$\vdash \chi \to [Send\ \psi](i \to O_s\alpha)$	
8	$\vdash \chi \to [Send\ \psi]\chi$	

QED

Dually, promulgating a norm does not change the knowledge of the Recipient (as a consequence of axiom A_9).

The following theorem tells us that the model checking problem of our logic is tractable and the same as for the language $\mathscr{L}_{EDL}$ without the dynamic operators $[Send\ \psi]$ and $[Prom\ \alpha]$ (see Theorem 2).

Theorem 4 (Decidability and complexity) *The satisfiability problem for $\mathscr{L}_{DEDL}$ is decidable. There is an algorithm (Algorithm 1) that, given a finite pointed EDL-model (M,w) and a formula $\phi \in \mathscr{L}_{DEDL}$, determines, in time $O(||M|| \times |\phi|)$, whether $M,w \models \phi$.*

Proof Decidability (just as completeness of the axiomatization) comes from Lemma 1 because L_{EDL} has the finite model property.

The model checking algorithm 1 works as follows. We label the worlds of the *EDL*-model M with subformulas of ϕ in a particular order. We start with the formulas χ and α appearing in the subformulas of ϕ of the form $[Send\ \chi]\psi$ and $[Prom\ \alpha]\psi$. Then, we label the other subformulas of ϕ. In particular, we use the reduction axioms A_6 to A_{13} to label formulas involving dynamic operators $[Send\ \chi]\psi$ or $[Prom\ \alpha]\psi$. This step is made possible

Algorithm 1 Model-Check($(M,w),\phi$)

Input: A pointed *EDL*-model (M,w), $\phi \in \mathscr{L}_{DEDL}$
Output: **True** if $M,w \models \phi$, **False** otherwise

```
        Stack S = Empty Stack
        $Add_1(S,\phi)$
        $Add_2(S,\phi)$
 5:     while not $Empty(S)$ do
          $\psi \leftarrow Pop(S)$
          for all $w \in M$ do
            if $\psi = p$ then
              if $w \in V(p)$ then
10:             $Label(w,\psi)$
              end if
            else if $\psi = \neg\psi'$ then
              if $\psi' \notin Label(w)$ then
                $Label(w,\psi)$
15:           end if
            else if $\psi = \psi' \wedge \psi''$ then
              if $\psi' \in Label(w)$ and $\psi'' \in Label(w)$ then
                $Label(w,\psi)$
              end if
20:         else if $\psi = K_r\psi'$ then
              Boolean b $\leftarrow$ True
              for all $(w,v) \in R$ do
                if $\psi' \notin Label(v)$ then
                  b $\leftarrow$ False
25:             end if
              end for
              if $b =$ True then
                $Label(w,\psi)$
              end if
30:         else if $\psi = O_s\alpha$ then
              Boolean $b \leftarrow$ True
              for all $(w,v) \in R$ do
                if $\alpha \notin Label(v)$ then
                  b $\leftarrow$ False
35:             end if
              end for
              if $b =$ True then
                $Label(w,O_s\alpha)$
              end if
40:         else if $\psi = \Box p$ then
              if $p \in Label(w)$ then
                $Label(w,\psi)$
              end if
            else if $\psi = \Box\neg\psi'$ then
45:           if $\Box\psi' \notin Label(w)$ then
                $Label(w,\psi)$
              end if
            else if $\psi = \Box(\psi' \wedge \psi'')$ then
              if $\Box\psi' \in Label(w)$ and $\Box\psi'' \in Label(w)$ then
50:             $Label(w,\Box(\psi' \wedge \psi''))$
              end if
            else if $\psi = [Send\ \chi]K_r\psi'$ then
              if $\chi \in Label(w)$ then
                Boolean b $\leftarrow$ True
55:             for all $(w,v) \in R$ do
                  if $\chi \in Label(v)$ and $[Send\ \chi]\psi' \notin Label(w)$ then
                    b = False
                  end if
                end for
60:             if b = True then
                  $Label(w,[Send\ \chi]K_r\psi')$
                end if
              else if $K_r\psi' \in Label(w)$ then
                $Label(w,[Send\ \chi]K_r\psi')$
65:           end if
            else if $\psi = [Send\ \chi]O_s\alpha$ then
              if $O_s\alpha \in Label(w)$ then
                $Label(w,[Send\ \chi]O_s\alpha)$
              end if
70:         else if $\psi = [Prom\ \alpha]K_r\psi'$ then
              if $K_r\psi' \in Label(w)$ then
                $Label(w,[Prom\ \alpha]K_r\psi')$
              end if
            else if $\psi = [Prom\ \alpha]O_s\alpha'$ then
75:           if $\alpha \in Label(w)$ then
                Boolean b $\leftarrow$ True
                for all $(w,v) \in D$ do
                  if $\alpha \in Label(v)$ and $[Prom\ \alpha]\alpha' \notin Label(w)$ then
                    b = False
80:               end if
                end for
                if b = True then
                  $Label(w,[Prom\ \alpha]O_s\alpha')$
                end if
85:           else if $O_s\alpha' \in Label(w)$ then
                $Label(w,[Prom\ \alpha]O_s\alpha')$
              end if
            end if
          end for
90:     end while
        if $\phi \in Label(w)$ then
          Return True
        elseReturn False
        end if
```

The symbol $\Box$ in lines 43-51 stands for $[Send\ \psi]$ or $[Prom\ \alpha]$. We use this notation in order to avoid repeating the same instructions twice.

Algorithm 2 $Add_1(S,\phi)$

```
Input: A Stack $S$ and a formula $\phi$
Output: The stack $S$ added with the subformulas of $\phi$,
    ignoring the subformulas $\chi$ and $\alpha$ of $\phi$ appearing
    in dynamic operators $[Send\ \chi]$ and $[Prom\ \alpha]$ .

    $Push(S,\phi)$
    if $\phi = \psi \wedge \psi'$ then
      $Add_1(S,\psi)$
      $Add_1(S,\psi')$
    else if $\phi = K_r\psi, \neg\psi$ then
      $Add_1(S,\psi')$
    else if $\phi = O_s\alpha$ then
      $Add_1(S,\alpha)$
    else if $\phi = [Send\ \psi]p$ then
      $Add_1(S,p)$
    else if $\phi = [Send\ \psi]\psi' \wedge \psi''$ then
      $Add_1(S,[Send\ \psi]\psi')$
      $Add_1(S,[Send\ \psi]\psi'')$
    else if $\phi = [Send\ \psi]\neg\psi'$ then
      $Add_1(S,[Send\ \psi]\psi')$
    else if $\phi = [Send\ \psi]O_s\alpha$ then
      $Add_1(S,O_s\alpha)$
    else if $\phi = [Send\ \psi]K_r\psi'$ then
      $Add_1(S,[Send\ \psi]\psi')$
      $Add_1(S,K_r\psi')$
    else if $\phi = [Prom\ \alpha]p$ then
      $Add_1(S,p)$
    else if $\phi = [Prom\ \alpha]\psi \wedge \psi'$ then
      $Add_1(S,[Prom\ \alpha]\psi)$
      $Add_1(S,[Prom\ \alpha]\psi')$
    else if $\phi = [Prom\ \alpha]\neg\psi$ then
      $Add_1(S,[Prom\ \alpha]\psi)$
    else if $\phi = [Prom\ \alpha]O_s\alpha'$ then
      $Add_1(S,[Prom\ \alpha]\alpha')$
      $Add_1(S,O_s\alpha')$
    else if $\phi = [Prom\ \alpha]K_r\psi$ then
      $Add_1(S,K_r\psi)$
    end if
```

Algorithm 3 $Add_2(S,\phi)$

```
Input: A stack $S$ and a formula $\phi$
Output: The stack $S$ added with the subformulas
    $\chi$ and $\alpha$ of $\phi$ appearing in dynamic operators
    $[Send\ \chi]$ or $[Prom\ \alpha]$

    if $\phi = \psi \wedge \psi'$ then
      $Add_2(S,\psi)$
      $Add_2(S,\psi')$
    else if $\phi = O_s\alpha$ then
      $Add_2(S,\alpha)$
    else if $\phi = K_r\psi, \neg\psi$ then
      $Add_2(S,\psi)$
    else if $\phi = [Send\ \psi]\psi'$ then
      $Push(S,\psi)$
      $Add_1(S,\psi)$
      $Add_2(S,\psi)$
      $Add_2(S,\psi)$
    else if $\phi = [Prom\ \alpha]\psi$ then
      $Push(S,\alpha)$
      $Add_1(S,\alpha)$
      $Add_2(S,\alpha)$
      $Add_2(S,\psi)$
    end if
```

because we already labelled the worlds of the *EDL*-model with the formulas χ and α. This order of labelling is implemented by a stack S in algorithms 2 and 3 ($Add_1(S,\phi)$ and $Add_2(S,\phi)$). Algorithm $Add_1(S,\phi)$ adds on the stack the subformulas of ϕ by ignoring the content of messages χ and promulgation α. Besides, to make use of the reduction axioms, dynamic formulas $[Send\ \chi]\psi$ and $[Prom\ \alpha]\psi$ are decomposed according to the reduction axioms A_6 to A_{13}. Then the algorithm $Add_2(S,\phi)$ adds on the stack the content of message χ and promulgation α, which are themselves decomposed into subformulas by means of $Add_1(S,\chi)$ and $Add_1(S,\alpha)$.

Termination of algorithms 1, 2 and 3 is ensured by the fact that within each of these algorithms, calls to other algorithms are made with a formula of degree strictly smaller. Correctness of these algorithms can be proved by induction using the truth conditions of Definition 2 and the soundness of Axioms A_6 to A_{13}. As for complexity, we first prove by induction that the running time of $Add_1(S,\phi)$ is in $O(|\phi|_1)$, and that the running time of $Add_2(S,\phi)$ is in $O(|\phi_2|)$. Informally, $|\phi|_1$ is the size of ϕ ignoring the dynamic operators $[Send\ \chi]$ and $[Prom\ \alpha]$.[4] Dually, $|\phi|_2$ is the size of ϕ considering only the formulas χ and α in the dynamic operators $[Send\ \chi]\psi$ and $[Prom\ \alpha]\psi$ in the sub-

[4] Formally, $|\phi|_1$ is defined inductively as follows: $|p|_1 = 1$, $|\neg\phi|_1 = |K_r\phi|_1 = 1 + |\phi|_1$, $|\phi \wedge \phi'|_1 = 1 + |\phi|_1 + |\phi'|_1$, $|O_s\alpha|_1 = 1 + |\alpha|_1$, $|[Send\ \psi]\phi|_1 = 1 + |\phi|_1$, $|[Prom\ \alpha]\phi|_1 = 1 + |\phi|_1$.

formulas of ϕ.[5] One can show by induction on ϕ that $|\phi| = |\phi|_1 + |\phi|_2$. We only prove the case $[Send\ \chi]K_r\psi$ for algorithm 2. By definition of Add_1 and by induction hypothesis, the running time of $Add_1(S, [Send\ \chi]K_r\psi)$ is in $O(|[Send\ \chi]\phi|_1) + O(|K_r\psi|_1) + 1$, i.e. $O(|\psi|_1) + O(|K_r\psi|_1) + 1$, i.e. $O(|K_r\psi|_1) + 1$, i.e. $O(|[Send\ \chi]K_r\psi|_1)$. We only prove the case $[Send\ \chi]\phi$ for algorithm 3. By definition of Add_2 and by induction hypothesis, the running time of $Add_2(S, [Send\ \chi]\phi)$ is in $1 + O(|\chi|_1) + O(|\chi_2|) + O(|\phi|_2)$, i.e. $1 + O(|\chi|) + O(|\phi|_2)$, i.e. $O(|\chi|) + O(|\phi|_2)$, i.e. $O(|[Send\ \chi]\phi|_2)$.

We show that the running time of $Model - Check(M, \phi)$ is in $O(||M|| \times |\phi|)$. We show the following invariant I:

Invariant I: at the start of each iteration of the **while** loop, of lines 5-90, the total running time of the **while** loop is in $O(||M|| \times x)$, where x is equal to the size of the formula removed from the stack S in the previous loop and equal to 0 if none has been removed yet.

We prove the Invariant by induction on the size of the previously removed formula. The base case holds trivially. We prove the induction step only for the case where the previously removed formula is of the form $\psi = [Send\ \chi]K_r\psi'$. One can easily show that for every pair of formulas ρ, ρ' in stack S such that ρ' is on the top of ρ, we have that $|\rho'| < |\rho|$. Therefore, the total running time of the **while** loop before the removal of ψ is in $O(||M|| \times |\psi'|)$, and therefore also in $O(||M|| \times (|\psi| - 1))$. Now, the worlds of M have already been labelled by the formulas χ, $[Send\ \chi]\psi$ and $K_r\psi$, because these formulas were placed on top of $[Send\ \chi]K_r\psi$ in the stack S by definition of Add_1 and Add_2 (lines 3 and 4). Therefore, we have to determine the complexity of the part of the algorithm between the lines 53 and 69. The running time of this part of algorithm 1 is in $O(|R|)$, i.e. $O(||M||)$. Hence, the total running time of the **while** loop is in $O(||M|| \times (|\psi| - 1)) + O(||M||) = O(||M|| \times |\psi|)$. This proves the induction step.

Since the last formula removed from the stack is ϕ, we have that the total running time of the **while** loop at the end of algorithm 1 is in $O(||M|| \times |\phi|)$. Since the total running time of Add_1 and Add_2 in lines 3 and 4 is in $O(|\phi|)$, we have that the total running time of algorithm 1 is in $O(||M|| \times |\phi|)$. QED

3.2.2 Permitted and obligatory messages

Now that we have defined the notion of compliance and the dynamic operation $[Send\ \psi]$ of message sending and the way this message affects the knowledge of the recipient, we can naturally derive and define the notions of permitted, prohibited and obligatory messages. Obviously, given a privacy policy and a situation, some messages might not be permitted by the privacy policy, because they might lead to a non-compliant situation. Moreover, because we assumed that the Sender sends only truthfull message, a message permitted by the privacy policy should be truthfull. This leads us to the following definition.

Definition 8 (Permitted message) Let $\phi \in \mathscr{L}_{DEDL}$, $\mathscr{P}$ be a privacy policy and (M, w) an *EDL*-model representing a given situation. We say that *it is permitted for the* Sender *to send message* ϕ according to $\mathscr{P}$ in (M, w), written $M, w \models P_s(Send\ \phi)$, if $M, w \models \phi$ and $(M \oplus \phi, w)$ is compliant with respect to $\mathscr{P}$.

[5] Formally, $|\phi|_2$ is defined inductively as follows: $|p|_2 = 0$, $|\neg\phi|_2 = |K_r\phi|_2 = |\phi|_2$, $|\phi \wedge \phi'|_2 = |\phi|_2 + |\phi'|_2$, $|O_s\alpha|_2 = |\alpha|_2$, $|[Send\ \psi]\phi|_2 = |\psi| + |\phi|_2$, $|[Prom\ \alpha]\phi|_2 = |\alpha| + |\phi|_2$.

Example 17 The following formulas illustrate permissions to send.

- $\neg P_s(Send\ a)$: the Sender's action $[Send\ a]$ is not permitted.
- $\neg P_s(Send\ m) \rightarrow \neg P_s(Send\ a)$: if it is prohibited to send the name of the street, then it is prohibited to send the entire address.
- $[Send\ e]\neg P_s(Send\ c)$ and $[Send\ c]\ \neg P_s(Send\ e)$:after sending one of the two messages, the other one becomes prohibited.
- A message is permitted ($P_s(Send\ a)$), if sending it does not lead to a non-compliant situation: i.e., if sending it does not violate some epistemic obligation (of the privacy policy). For example, sending the message a ($[Send\ a]$) leads to a state where $K_r a$ while at the same time $\neg P_s K_r a$ (i.e., $O_s \neg K_r a$).

The definition of an obligatory message is a bit more tricky. Typically, the Sender is obliged to send a message when the situation is not compliant with respect to the privacy policy and sending this message restores compliance. Besides, the amount of information that the Sender should send in this context has to be minimal. For example, if it is obligatory for the editor of the Journal of Artificial and Law that we know that the submission deadline of the paper is the first of June, then he can either tell us that "the submission deadline is the first of June" or "the submission deadline is the first of June *and* there might be an extension". The first announcement is obligatory but the second one is not because the amount of information sent is not minimal, unless we should also know that there might be an extension. This leads us to define the following notions.

Definition 9 (Informativeness of a formula) Let $\phi, \phi \in \mathscr{L}_{DEDL}$ and (M,w) an *EDL*-model representing a given situation. We say that ϕ *is more informative than* ϕ' for the Recipient in the situation (M,w), written $M,w \models \phi \geq \phi'$, if $M,w \models K_r(\phi \rightarrow \phi')$. ϕ *is strictly more informative than* ϕ' for the recipient in the situation (M,w), written $M,w \models \phi > \phi'$, when $M,w \models \phi \geq \phi'$ but not $M,w \models \phi' \geq \phi$.

Equipped with this notion of minimal information, we can now define formally the notion of obligation to send.

Definition 10 (Obligatory message) Let $\phi \in \mathscr{L}_{DEDL}$, $\mathscr{P}$ be a privacy policy and (M,w) an *EDL*-model representing a given situation. We say that *it is obligatory for the Sender to send message* ϕ according to $\mathscr{P}$ in (M,w), written $M,w \models O_s(Send\ \phi)$, if the situation (M,w) is not compliant, sending ϕ restores compliance, and sending a message ϕ' strictly less informative than ϕ does not restore compliance. Formally, $M,w \models O_s(Send\ \phi)$ if and only if $M,w \models \neg Comp \wedge P_s(Send\ \phi)$ and for all $\phi' \in \mathscr{L}_{DEDL}$, if $M,w \models \phi > \phi'$ then $M,w \not\models [Send\ \phi']Comp$.

Note that the notions of permitted and obligatory message in Definitions 8 and 10 are both based on the notion of compliance. However, because of Proposition 3, this notion could be replaced by the notion of behavioural compliance.

Remark 1 In fact, one could also replace this notion of compliance by the notion of strong compliance. This would yield a stronger and a weaker notion of permitted and obligatory message respectively. In the same line, we could also define the notions of *permitted promulgation* and *obligatory promulgation* by replacing in these definitions the sending of message $[Send\ \phi]$ with a promulgation $[Prom\ \alpha]$ and by replacing the notion of (behavioural) compliance with the more appropriate notion of regulatory compliance.

Example 18 (Website example continued) In Example 13, we have:

$$M,w \models P_s(Send\ c) \wedge P_s(Send\ e).$$

So it is permitted to send the mappings from the users to the numbers (c) and it is permitted to send the mapping from the numbers to the web-sites (e). However, we also have

$$M,w \models [Send\ e]\neg P_s(Send\ c) \text{ and } M,w \models [Send\ c]\neg P_s(Send\ e)$$

which means that after sending the mapping from the numbers to the web-sites (e) it is *not* permitted to send the mapping from the users to the numbers (c), and vice versa for the second conjunct. This is because in both cases we would violate the epistemic norm $\neg P_s K_r v$:

$$M,w \models [Send\ e][Send\ c](K_r v \wedge \neg P_s K_r v) \text{ and}$$
$$M,w \models [Send\ c][Send\ e](K_r v \wedge \neg P_s K_r v).$$

We also have

$$M,w \models \neg P_s(Send\ (e \wedge c)).$$

Our approach is flexible because it is applicable in infinitely many other contexts than the one of the above example, once the privacy policy is fixed. For example, assume that the hash function computing the mapping from users to numbers is now available (h) and that the Recipient is able to apply it to get the mapping from numbers to users (c):

$$M \models h \to c.$$

Applying the same reasoning, we would get:

$$M,w \models [Send\ e]\neg P_s(Send\ h)$$
$$M,w \models \neg P_s(Send\ (e \wedge h))$$

and so we derive the forbidden messages without having to introduce explicitly new prohibitions or permissions on h.

Privacy policies do not only concern which information are permitted to be disclosed, but also which information *should* be disclosed. Example 15 illustrates that we can express such policies due to the fact that our epistemic deontic logic can express obligations about knowledge, unlike the one of Cuppens and Demolombe.

Example 19 (Spyware Example continued) After sending the message e in the previous situation represented by the pointed *EDL*-model (M,w) of Figure 2 we obtain the pointed *EDL*-model $(M \oplus e, w)$ depicted in Figure 6. The corresponding situation $(M \oplus e, w)$ is still regulatory compliant with respect to $\mathscr{P}_2 = \{y \wedge K_r e \to O_s K_r y\}$, because $M \oplus e, w \models \mathscr{P}_2$. However, it is not behaviorally compliant with respect to $\mathscr{P}_2$, because $M,w \models (y \wedge K_r e) \wedge \neg K_r y$. Therefore, it was forbidden to disclose e:

$$M,w \models \neg P_s(Send\ e)$$

But it is now obligatory (with respect to $\mathscr{P}_2$) to disclose y:

$$M \oplus e, w \models O_s(Send\ y)$$

So we have that

$$M,w \models [Send\ e]O_s(Send\ y)$$
$$M,w \models \neg P_s(Send\ e) \wedge P_s(Send\ (e \wedge y)).$$

As it turns out, after sending the message y we reach a compliant situation.

$$\begin{tikzcd} e,y & \boxed{e,y} \arrow[l, dotted] \arrow[r, dash] & e,\neg y \end{tikzcd}$$

Fig. 6 Spyware example updated.

The above example suggests that even if it is prohibited to send message e, it might still be permitted to send message e as long as it is followed by another message y. We leave the investigation of the permissibility of iterative messages for future work.

In privacy policies, the permission to disclose the names of users also allows to disclose their family names (which are part of their name). This problem, discussed in Example 3, is known as the inference problem, and is in general difficult to model (see for instance Barth et al (2006)). In our logical framework it follows easily from the fact that the Recipient has reasoning capabilities. The following proposition illustrates this point. In this proposition, a *propositional privacy policy* is a privacy policy where all antecedent i of epistemic norms are propositional and all α in $i \to O_s\alpha$ are of the form $K_r\psi$ or $\neg K_r\psi$, where ψ is a propositional formula.

Proposition 4 *Let $\mathscr{P}$ be a propositional privacy policy and $\phi, \phi' \in \mathscr{L}_{DEDL}$,*

$$\vdash Comp \to \big((\phi \geq \phi') \to \big(P_s(Send\ \phi) \to P_s(Send\ \phi')\big)\big) \tag{5}$$

Therefore,

$$If \vdash \phi \to \phi' \ then\ \vdash Comp \to \big(P_s(Send\ \phi) \to P_s(Send\ \phi')\big) \tag{6}$$

Proof Assume that $M,w \models \phi$ and that for all $i \to O_s\alpha \in \mathscr{P}$, $M,w \models i \to \alpha$ and $M,w \models [Send\ \phi](i \to \alpha)$, i.e. $M,w \models i \to [Send\ \phi]\alpha$ by Axioms $\mathsf{A}_{12} - \mathsf{A}_{14}$. We have to show that for all $i \to O_s\alpha \in \mathscr{P}$, $M,w \models [Send\ \phi'](i \to \alpha)$, i.e. $M,w \models i \to [Send\ \phi']\alpha$ by Axioms $\mathsf{A}_{12} - \mathsf{A}_{14}$.

- If $M,w \models \neg i$ then trivially $M,w \models i \to [Send\ \phi']\alpha$.
- If $M,w \models i$ then we have to show that $M,w \models [Send\ \phi']\alpha$.
  - if $\alpha = K_r\psi$, then $M,w \models K_r\psi$ by assumption. Therefore, $M,w \models K_r(\phi' \to \psi)$. Therefore, $M,w \models [Send\ \phi']K_r\psi$ by Axiom A_6, i.e. $M,w \models [Send\ \phi']\alpha$.
  - if $\alpha = \neg K_r\psi$, then $M,w \models [Send\ \phi]\neg K_r\psi$ by assumption. Therefore, $M,w \models \hat{K}_r(\phi \wedge \neg\psi)$ by Axioms A_6 and A_{13}. However, because $M,w \models \phi \geq \phi'$, we also have that $M,w \models \hat{K}_r(\phi' \wedge \neg\psi)$. Then, $M,w \models [Send\ \phi']\neg K_r\psi$ by Axioms A_6 and $\mathsf{A}_{12} - \mathsf{A}_{14}$. Therefore, $M,w \models [Send\ \phi']\alpha$.

In any case, we have proved that for all $i \to O_s\alpha \in \mathscr{P}$, $M,w \models [Send\ \phi'](i \to \alpha)$. Therefore $M,w \models [Send\ \phi']BehComp$.

Now, because the privacy policy $\mathscr{P}$ is propositional, we also have that $M,w \models [Send\ \phi']$ $RegComp$ because $M,w \models RegComp$. So, finally, $M,w \models [Send\ \phi']Comp$, and so $M,w \models P_s(Send\ \phi')$. QED

Example 20 (Website example continued) Assume we have a situation modeled by an *EDL*-model M such that $M \models v \to v'$: the association between the users' name and the web-sites they visited (v) induces the association between the users' *family* name and the web-sites they visited (v'). So if $M,w \models P_s(Send\ v)$ then $M,w \models P_s(Send\ v')$: if it is permitted to disclose the name of the users in association with the websites they visited, it is also permitted to disclose their family name in association with the websites they visited. Dually, if $M \models v \rightarrow v'$, then $M, w \models \neg P_s(\mathit{Send}\ v')$ implies $M, w \models \neg P_s(\mathit{Send}\ v)$: if it is prohibited to disclose their family names in association with the websites they visited then it is also prohibited to disclose their names in association with the websites they visited.

The following property connects the notions of permitted and obligatory messages: for all $\phi, \phi' \in \mathscr{L}_{DEDL}$,

$$\vdash \phi > \phi' \rightarrow \big(O_s(\mathit{Send}\ \phi) \rightarrow \neg P_s(\mathit{Send}\ \phi')\big) \tag{7}$$

This proposition illustrate the minimality feature of our definition of obligatory message: if ϕ is strictly more informative that ϕ', then the obligation to send ϕ entails that sending ϕ' will not lead to a compliant situation. Moreover, note that $O_s(\mathit{Send}\ \phi)$ and $P_s(\mathit{Send}\ \phi)$ are not dual operators:

$$\nvdash O_s(\mathit{Send}\ \phi) \leftrightarrow \neg P_s(\mathit{Send}\ \neg\phi) \tag{8}$$

This is intuitively correct: in Example 19 it is prohibited to disclose e but it does not entail that it is obligatory to disclose $\neg e$. We also have the following property:

$$\nvdash P_s(\mathit{Send}\ \phi) \wedge P_s(\mathit{Send}\ \psi) \rightarrow P_s(\mathit{Send}\ (\phi \wedge \psi)) \tag{9}$$

Indeed, in Example 18 we had $M, w \models P_s(\mathit{Send}\ e) \wedge P_s(\mathit{Send}\ c) \wedge \neg P_s(\mathit{Send}\ (e \wedge c))$. The next example describes the interaction between meta-policy and actions.

Example 21 Consider the meta-policy of Example 9, where p should not be known and it is prohibited to let the Recipient know the policy that knowing p is prohibited:

$$\mathscr{P}_4 = \{\neg P_s K_r p, \neg P_s K_r \neg P_s K_r p\}$$

In this situation, not only sending the message $[\mathit{Send}\ p]$ but also sending the message $[\mathit{Send}\ \neg P_s K_r p]$ lead to a violation:

$$\vdash \mathscr{P}_4 \rightarrow \neg P_s(\mathit{Send}\ P_s K_r p).$$

As shown in Proposition 3, the privacy policy persists during the sending of the message. However, our logic L_{DEDL} does not allow us to derive that sending to the Recipient the piece of information whereby he should not know that p implies that as a result of this sending he actually knows p:

$$\nvdash \mathscr{P}_4 \rightarrow [\mathit{Send}\ \neg P_s K_r p] K_r p.$$

This is nevertheless a theorem in an extension of our logic spelled out in the appendix.

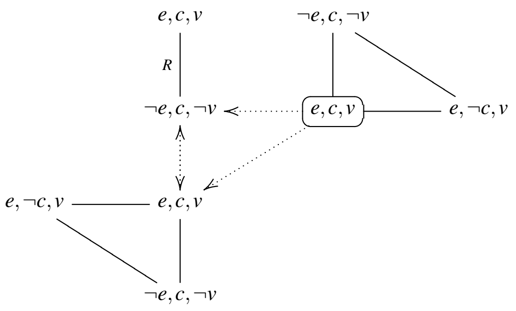

Fig. 7 Website example of Figure 1 updated by $\neg K_r e$.

3.2.3 Enforcing privacy policies: $[Prom\ \alpha]$

The hierarchical superior of the Sender or the Sender himself might decide to change the policy privacy from $\mathscr{P}$ to $\mathscr{P}'$. As a result, the sender needs to enforce this new privacy policy $\mathscr{P}'$. This enforcement is captured in our formalism by $[Prom\ \alpha]$.

Example 22 (Website Example) In case of attack by some hacker, the privacy policies can be made more strict. For example, the Sender can decide to strengthen the privacy policy $\mathscr{P}_1$ of Example 13 to

$$\mathscr{P}_5 = \{P_s K_r c, \neg P_s K_r e, \neg P_s K_r v\}$$

where $P_s K_r e$ has been replaced by $\neg P_s K_r e$: it is now prohibited to disclose the mapping from numbers to visited websites. This new privacy policy $\mathscr{P}_5$ can be enforced by the Sender through the update $[Prom\ \neg K_r e]$. We get the *EDL*-model $(M \odot \neg K_r e, w)$ visualized in Figure 7 which is compliant with respect to $\mathscr{P}_5$.

4 A privacy logic for security monitors

If the Sender, viewed as a security monitor, wanted to use our logic in real situations to decide which actions to execute in order to enforce and maintain a privacy policy, then he could implement an *EDL*-model representing the current epistemic and deontic state of affairs. He could then check compliance with respect to a given policy and determine which actions can and should be done to enforce and maintain this privacy policy by model checking this *EDL*-model. However, with the current language $\mathscr{L}_{DEDL}$, the Sender would have to face some problems. For example, he could not check whether a situation is regulatory or behaviourally compliant with respect to a given privacy policy $\mathscr{P}$ because privacy policies are not represented in (the syntax of) the language $\mathscr{L}_{DEDL}$. This language $\mathscr{L}_{DEDL}$ would not allow him to express that an epistemic norm is added or removed to/from the privacy policy by the law/policy maker. This language would not allow him to decide which actions he needs to do so that the new situation is regulatory compliant. It would not allow him to express that under the current privacy policy he is permitted to disclose ϕ. These kinds of statements are all needed if we want the security monitor (Sender) to be able to enforce and maintain a privacy policy in real situations. So we need to define a new language based on $\mathscr{L}_{DEDL}$ more appropriate in the context of privacy policy. This language should allow the security monitor to refer explicitly to the current privacy policy, which was not explicitly introduced in the previous language. We propose the following language $\mathscr{L}_{PL}$.

Definition 11 (Syntax of $\mathscr{L}_{PL}$) The language $\mathscr{L}_{PL}$ is defined inductively as follows:

$$\mathscr{L}_{PL} : \phi ::= \psi \mid (\chi \in \mathscr{P}) \mid RegComp \mid BehComp \mid Comp \mid P_s(Send\ \psi) \mid$$

$$[Send\ \psi]\phi \mid [Prom\ \alpha]\phi \mid [+\chi]\phi \mid [-\chi]\phi \mid \neg\phi \mid \phi \wedge \phi$$

where ψ ranges over $\mathscr{L}_{DEDL}$, α ranges over $\mathscr{L}^{\alpha}_{DEDL}$ and χ ranges over $\mathscr{E}\mathscr{N}$. We assume here that the set of epistemic norms $\mathscr{E}\mathscr{N}$ is finite.

So we have seven new kinds of formulas referring each of them directly or indirectly to privacy policies: $(\chi \in \mathscr{P})$, $RegComp$, $BehComp$, $Comp$, $[+\chi]\phi$, $[-\chi]\phi$ and $P_s(Send\ \psi)$. The formula $(\chi \in \mathscr{P})$ reads as 'χ is an epistemic norm of the current privacy policy'. The formula $RegComp$ reads as 'the current situation is regulatory compliant with respect to the current privacy policy'. The formula $BehComp$ reads as 'the current situation is behaviourally compliant with respect to the current privacy policy'. The formula $Comp$ reads as 'the current situation is compliant with respect to the current privacy policy'. Note that this constant $Comp$ is similar in spirit to the violation constant v of Anderson (1958): $Comp$ is somehow equivalent to $\neg\mathrm{v}$. The formula $[+\chi]\phi$ reads as 'after adding the epistemic norm χ to the current privacy policy, ϕ holds'. The formula $[-\chi]\phi$ reads as 'after removing the epistemic norm χ from the current privacy policy, ϕ holds'. The formula $P_s(Send\ \psi)$ reads as 'sending the message ψ is permitted'. This language allows to express all the new kinds of statement we wanted to express above. For example, $\neg Comp \wedge [Send\ \psi]Comp$ means that the current situation is not compliant with respect to the privacy policy but if ψ is disclosed then the situation becomes compliant with this privacy policy.

The semantics for this language is a bit different from the semantics of $\mathscr{L}_{DEDL}$ because we have to refer explicitly in the language to privacy policies. Intuitively, $\{(M,w),\mathscr{P}\}$ in the definition below is the situation (M,w) which has to comply with the privacy policy $\mathscr{P}$.

Definition 12 (Semantics of $\mathscr{L}_{PL}$) A (pointed) privacy model is a pair $\{M,\mathscr{P}\}$ (resp. $\{(M,w),\mathscr{P}\}$) composed of an EDL-model M (resp. (M,w)) together with a privacy policy $\mathscr{P}$. The truth conditions are defined inductively as follows:

$$\begin{array}{lll}
\{(M,w),\mathscr{P}\} \models \psi & \text{iff} & M,w \models \psi \\
\{(M,w),\mathscr{P}\} \models (\chi \in \mathscr{P}) & \text{iff} & \chi \in \mathscr{P} \\
\{(M,w),\mathscr{P}\} \models RegComp & \text{iff} & M,w \models P_s\alpha \text{ for all } i \to P_s\alpha \in \mathscr{P} \text{ and} \\
 & & M,w \models O_s\alpha \text{ for all } i \to O_s\alpha \in \mathscr{P} \\
\{(M,w),\mathscr{P}\} \models BehComp & \text{iff} & M,w \models i \to \alpha \text{ for all } i \to O_s\alpha \in \mathscr{P} \\
\{(M,w),\mathscr{P}\} \models Comp & \text{iff} & \{(M,w),\mathscr{P}\} \models RegComp \wedge BehComp \\
\{(M,w),\mathscr{P}\} \models P_s(Send\ \psi) & \text{iff} & M,w \models \psi \text{ and } \{(M,w),\mathscr{P}\} \models [Send\ \psi]Comp \\
\{(M,w),\mathscr{P}\} \models [Send\ \psi]\phi & \text{iff} & \{(M \oplus \psi, w_{\oplus}),\mathscr{P}\} \models \phi \\
\{(M,w),\mathscr{P}\} \models [Prom\ \alpha]\phi & \text{iff} & \{(M \odot \alpha, w_{\odot}),\mathscr{P}\} \models \phi \\
\{(M,w),\mathscr{P}\} \models [+\chi]\phi & \text{iff} & \{(M,w),\mathscr{P} \cup \{\chi\}\} \models \phi \\
\{(M,w),\mathscr{P}\} \models [-\chi]\phi & \text{iff} & \{(M,w),\mathscr{P} - \{\chi\}\} \models \phi
\end{array}$$

Note that the semantics of *RegComp*, *BehComp* and *Comp* correspond exactly to the definitions of these notions we gave in Definition 5. Likewise, the semantics of $P_s(Send\ \psi)$ corresponds to the definition of this notion we gave in Definition 8.

Theorem 5 (Soundness and completeness) *The logic* PL *axiomatized by the following axiom schemes and inference rules is* sound and complete *for the language* $\mathscr{L}_{PL}$ *with respect to the semantics of privacy models. The symbol* $[\pm\chi]$ *below stands either for* $[+\chi]$ *or* $[-\chi]$. *The symbol* $\Box$ *stands either for* $[Send\ \psi]$ *or* $[Prom\ \alpha]$.

L_{DEDL}	*All the axioms schemes and inference rules of* L_{DEDL}	
P_1	$\vdash BehComp \leftrightarrow \bigwedge\limits_{\chi=i\to O_s\alpha\in\mathscr{E}\mathscr{N}} ((\chi\in\mathscr{P})\to(i_\chi\to\alpha_\chi))$	
P_2	$\vdash RegComp \leftrightarrow \bigwedge\limits_{\chi=i\to O_s\alpha\in\mathscr{E}\mathscr{N}} ((\chi\in\mathscr{P})\to O_s\alpha)\wedge \bigwedge\limits_{\chi=i\to P_s\alpha\in\mathscr{E}\mathscr{N}} ((\chi\in\mathscr{P})\to P_s\alpha)$	
P_3	$\vdash Comp \leftrightarrow (RegComp\wedge BehComp)$	
P_4	$\vdash P_s(Send\ \psi)\leftrightarrow(\psi\wedge[Send\ \psi]Comp)$	
P_5	$\vdash \Box(\chi\in\mathscr{P})\leftrightarrow(\chi\in\mathscr{P})$	
P_6	$\vdash \Box\neg\phi\leftrightarrow\neg\Box\phi$	
P_7	$\vdash \Box(\phi\to\phi')\to(\Box\phi\to\Box\phi')$	
P_8	$\vdash [+\chi](\chi\in\mathscr{P})$	
P_9	$\vdash [-\chi]\neg(\chi\in\mathscr{P})$	
P_{10}	$\vdash [\pm\chi](\chi'\in\mathscr{P})\leftrightarrow(\chi'\in\mathscr{P})$	*for all* $\chi'\in\mathscr{E}\mathscr{N}-\{\chi\}$
P_{11}	$\vdash [\pm\chi]\psi\leftrightarrow\psi$	
P_{12}	$\vdash [\pm\chi]\neg\phi\leftrightarrow\neg[\pm\chi]\phi$	
P_{13}	$\vdash [\pm\chi](\phi\to\phi')\to([\pm\chi]\phi\to[\pm\chi]\phi')$	
R_P	*If* $\vdash\phi$ *then* $\vdash[\pm\chi]\phi$	

Proof (sketch) Soundness is routine, we only prove completeness. We use the same method as for the proof of Theorem 3. Axioms P_5 to P_{13} and rule R_P allow to reduce any PL-consistent formula ϕ of $\mathscr{L}_{PL}$ to a PL-consistent formula ϕ' of $\mathscr{L}_{PL}$ without dynamic operators, i.e. a formula which is a boolean combination of $\psi\in\mathscr{L}_{DEDL}$ and $(\chi\in\mathscr{P})$. We can then equivalently rewrite this formula ϕ' in disjunctive normal form:

$$\phi' = \bigvee_{i\in\{1,\dots,k\}} \Big(\psi^i\wedge(\chi^i_1\in\mathscr{P})\wedge\ldots\wedge(\chi^i_m\in\mathscr{P})\wedge\neg(\chi^{i'}_1\in\mathscr{P})\wedge\ldots\wedge\neg(\chi^{i'}_n\in\mathscr{P})\Big)$$

where for all $i\in\{1,\dots,k\}$, $\psi^i\in\mathscr{L}_{DEDL}$ and all j, $\chi^i_j,\chi^{i'}_j\in\mathscr{E}\mathscr{N}$. Because ϕ' is consistent, there must be a $i\in\{1,\dots,k\}$ such that $\psi^i\wedge(\chi^i_1\in\mathscr{P})\wedge\ldots\wedge(\chi^i_m\in\mathscr{P})\wedge\neg(\chi^{i'}_1\in\mathscr{P})\wedge\ldots\wedge\neg(\chi^{i'}_n\in\mathscr{P})$ is PL-consistent. Then, ψ^i is PL-consistent, and therefore also L_{DEDL}-consistent. Henceforth, by Theorem 3, there is a pointed *EDL*-model (M,w) such that $M,w\models\psi^i$. Besides, it suffices to define $\mathscr{P}$ as $\mathscr{P}=\{\chi^i_j \mid j\in\{1,\dots,m\}\}$ to finally have that $\{(M,w),\mathscr{P}\}\models\phi'$. Therefore, because $\vdash\phi\leftrightarrow\phi'$ and by soundness of PL, we also have that $\{(M,w),\mathscr{P}\}\models\phi$. So ϕ is satisfiable in a privacy model. QED

If $\phi\in\mathscr{L}_{PL}$, we define $|\phi|$, the *size* of ϕ, to be the number of symbols in ϕ.[6] If $\mathscr{P}$ is a finite privacy policy then $|\mathscr{P}|=|\bigwedge\mathscr{P}|$. The following theorem states in particular that the

[6] Formally, $|\phi|$ is defined inductively as follows: if $\phi=\psi\in\mathscr{L}_{DEDL}$, then $|\phi|$ is defined as in Footnote 2; otherwise, $|\phi|=1+|\phi|, |\phi\wedge\phi'|=1+|\phi|+|\phi'|, |(\chi\in\mathscr{P})|=1, |RegComp|=1, |BehComp|=1, |Comp|=1, |P_s(Send\ \psi)|=1+|\psi|, |[Send\ \psi]\phi|=1+|\psi|+|\phi|, [+\chi]\phi|=1+|\chi|+|\phi|, |[-\chi]|\phi=1+|\chi|+|\phi|$

computational complexity of the model checking problem for our new language $\mathcal{L}_{PL}$ is basically the same as for the language $\mathcal{L}_{DEDL}$ except that the size of the privacy policy $\mathcal{P}$ has to be added to the size of the model M and the formula ϕ in $O(||M|| \times |\phi|)$ of Theorem 4.

Theorem 6 (Decidability and complexity) *The satisfiability problem for $\mathcal{L}_{PL}$ is decidable. There is an algorithm (Algorithm 4) that, given a pointed privacy model $\{(M,w),\mathcal{P}\}$ and a formula $\phi \in \mathcal{L}_{PL}$, determines, in time $O((||M|| + |\mathcal{P}|) \times (|\phi| + |\mathcal{P}|))$, whether $\{(M,w),\mathcal{P}\} \models \phi$.*

Proof Algorithms $Add_1^{\mathcal{P}}$ and $Add_2^{\mathcal{P}}$ (i.e. algorithms 5 and 6) called on lines 3 and 4 are given in the appendix. Because they terminate, algorithm 4 also terminates. Correctness of algorithm 4 is ensured by the truth conditions of language $\mathcal{L}_{PL}$ which are spelled out in Definition 12. As for complexity, the proof is similar to Theorem 4. First, at each iteration of the **while** loop, there are at most $O(||M|| + |\mathcal{P}|)$ operations, the worst case being obtained between lines 9 and 22. Now, we have to determine the size of the stack S. This size corresponds to the number of times the procedure *Push* is called in algorithms 5 and 6. As one can easily notice, by definition of algorithm 5, the number of times this happens in $Add_1^{\mathcal{P}}(S,(\phi,\mathcal{P}))$ is $|\phi|^1_{\mathcal{P}}$, where $|\phi|^1_{\mathcal{P}}$ is defined as follows:

$$\begin{array}{lcl@{\qquad}lcl}
|\psi|^1_{\mathcal{P}} & = & |\psi| & |[Send\ \psi]\phi|^1_{\mathcal{P}} & = & |\phi|^1_{\mathcal{P}} \\
|\neg\phi|^1_{\mathcal{P}} & = & 1+|\phi|^1_{\mathcal{P}} & |[Prom\ \alpha]\phi|^1_{\mathcal{P}} & = & |\phi|^1_{\mathcal{P}} \\
|\phi\wedge\phi'|^1_{\mathcal{P}} & = & 1+|\phi|^1_{\mathcal{P}}+|\phi'|^1_{\mathcal{P}} & |[+\chi]\phi|^1_{\mathcal{P}} & = & 1+|\phi|^1_{\mathcal{P}\cup\{\chi\}} \\
|BehComp|^1_{\mathcal{P}} & = & 1+|\bigwedge_{i\to O_s\alpha\in\mathcal{P}}(i\to\alpha)| & |[-\chi]|^1_{\mathcal{P}} & = & 1+|\phi|^1_{\mathcal{P}-\{\chi\}} \\
|RegComp|^1_{\mathcal{P}} & = & 1+|\bigwedge_{i\to O_s\alpha\in\mathcal{P}} O_s\alpha \wedge \bigwedge_{i\to P_s\alpha\in\mathcal{P}} P_s\alpha| & |(\chi\in\mathcal{P})|^1_{\mathcal{P}} & = & 1
\end{array}$$

$$\begin{array}{lcl}
|Comp|^1_{\mathcal{P}} & = & 1+|\bigwedge_{i\to O_s\alpha\in\mathcal{P}} O_s\alpha \wedge \bigwedge_{i\to P_s\alpha\in\mathcal{P}} P_s\alpha \wedge \bigwedge_{i\to O_s\alpha\in\mathcal{P}} (i\to\alpha)|^1_{\mathcal{P}} \\
|P_s(Send\ \psi)|^1_{\mathcal{P}} & = & 1+|[Send\ \psi](\bigwedge_{i\to O_s\alpha\in\mathcal{P}} (i\to\alpha)\wedge(\bigwedge_{i\to O_s\alpha\in\mathcal{P}} O_s\alpha \wedge \bigwedge_{i\to P_s\alpha\in\mathcal{P}} P_s\alpha))|^1_{\mathcal{P}}
\end{array}$$

Similarly, the number of times the procedure *Push* is called in $Add_2^{\mathcal{P}}(S,\phi,\mathcal{P})$ is $|\phi|^2_{\mathcal{P}}$, where $|\phi|^2_{\mathcal{P}}$ is defined as follows:

$$\begin{array}{lcl@{\qquad}lcl}
|\psi|^2_{\mathcal{P}} & = & 0 & |Comp|^2_{\mathcal{P}} & = & 0 \\
|\neg\phi|^2_{\mathcal{P}} & = & |\phi|^2_{\mathcal{P}} & |P_s(Send\ \psi)|^2_{\mathcal{P}} & = & 1+|\psi|^2_{\mathcal{P}}+|\psi|^1_{\mathcal{P}} \\
|\phi\wedge\phi'|^2_{\mathcal{P}} & = & |\phi|^2_{\mathcal{P}}+|\phi'|^2_{\mathcal{P}} & |[Send\ \psi]\phi|^2_{\mathcal{P}} & = & 1+|\psi|^1_{\mathcal{P}}+|\psi|^2_{\mathcal{P}}+|\phi|^2_{\mathcal{P}} \\
|(\chi\in\mathcal{P})|^2_{\mathcal{P}} & = & 0 & |[Prom\ \alpha]\phi|^1_{\mathcal{P}} & = & 1+|\alpha|^1_{\mathcal{P}}+|\alpha|^2_{\mathcal{P}}+|\phi|^2_{\mathcal{P}} \\
|RegComp|^2_{\mathcal{P}} & = & 0 & |[+\chi]\phi|^2_{\mathcal{P}} & = & |\phi|^2_{\mathcal{P}\cup\{\chi\}} \\
|BehComp|^2_{\mathcal{P}} & = & 0 & |[-\chi]\phi|^2_{\mathcal{P}} & = & |\phi|^2_{\mathcal{P}-\{\chi\}}
\end{array}$$

Therefore, the size of the stack S after the call of $Add_1^{\mathcal{P}}(S,(\phi,\mathcal{P}))$ and $Add_2^{\mathcal{P}}(S,\phi,\mathcal{P})$ is $|\phi|^1_{\mathcal{P}}+|\phi|^2_{\mathcal{P}}$. However, one can easily prove by induction on ϕ that for all $\phi \in \mathcal{L}_{PL}$,

Algorithm 4 Model-Check($\{(M,w),\mathscr{P}\},\phi$)

Input: A privacy model $\{(M,w),\mathscr{P}\}$, $\phi \in \mathscr{L}_{PL}$. The privacy policy $\mathscr{P}$ is implemented by a list.
Output: **True** if $\{(M,w),\mathscr{P}\} \models \phi$, **False** otherwise

```
     $StackS \leftarrow EmptyStack$
     $Add_1^{\mathscr{P}}(S,(\phi,\mathscr{P}))$
     $Add_2^{\mathscr{P}}(S,\phi,\mathscr{P})$
 5:  while not $Empty(S)$ do
       $(\psi,\mathscr{P}) \leftarrow Pop(S)$
       if $\psi \in \mathscr{L}_{DEDL}$ then
         $Model-Check^{\mathscr{P}}((M,w),(\psi,\mathscr{P}))$
       else if $\psi = (\chi \in \mathscr{P})$ then
10:      $Boolean\ b \leftarrow$ False
         $\chi' \leftarrow \mathscr{P}.head$
         while $\mathscr{P} \neq NULL$ and not $b$ do
           if $\chi' = \chi$ then
             $b \leftarrow$ True
15:        end if
           $\chi' \leftarrow \mathscr{P}.next$
         end while
         if $b$ then
           for all $w \in M$ do
20:          $Label(w,(\psi,\mathscr{P}))$
           end for
         end if
       else
         for all $w \in M$ do
25:        if $\psi = RegComp$ then
             if $(\bigwedge_{i \to O_s\alpha \in \mathscr{P}} O_s\alpha,\mathscr{P}) \in Label(w)$ and$(\bigwedge_{i \to P_s\alpha \in \mathscr{P}} P_s\alpha,\mathscr{P}) \in Label(w)$ then
               $Label(w,(\psi,\mathscr{P}))$
             end if
30:        else if $\psi = BehComp$ then
             if $(\bigwedge_{i \to O_s\alpha \in \mathscr{P}} (i \to \alpha),\mathscr{P}) \in Label(w)$ then
               $Label(w,(\psi,\mathscr{P}))$
             end if
           else if $\psi = Comp$ then
35:          if $(\bigwedge_{i \to O_s\alpha \in \mathscr{P}} O_s\alpha,\mathscr{P}) \in Label(w)$ and$(\bigwedge_{i \to P_s\alpha \in \mathscr{P}} P_s\alpha,\mathscr{P}) \in Label(w)$ and$(\bigwedge_{i \to O_s\alpha \in \mathscr{P}} (i \to \alpha),\mathscr{P}) \in Label(w)$ then
               $Label(w,(\psi,\mathscr{P}))$
             end if
           else if $\psi = \neg\psi'$ then
40:          if $(\psi',\mathscr{P}) \notin Label(w)$ then
               $Label(w,\psi,\mathscr{P})$
             end if
           else if $\psi = \psi' \wedge \psi''$ then
             if $(\psi',\mathscr{P}) \in Label(w)$ and $(\psi'',\mathscr{P}) \in Label(w)$ then
45:            $Label(w,(\psi,\mathscr{P}))$
             end if
           else if $\psi = P_s(Send\ \psi')$ then
             if $(\psi,\mathscr{P}) \in Label(w)$and$[Send\ \psi] \bigwedge_{i \to O_s\alpha \in \mathscr{P}} (i \to \alpha) \wedge \bigwedge_{i \to O_s\alpha \in \mathscr{P}} O_s\alpha \wedge \bigwedge_{i \to P_s\alpha \in \mathscr{P}} P_s\alpha,\mathscr{P}) \in Label(w)$ then
               $Label(w,(\psi,\mathscr{P}))$
50:          end if
           else if $\psi = \Box(\chi \in \mathscr{P})$ then
             if $((\chi \in \mathscr{P}),\mathscr{P}) \in Label(w)$ then
               $Label(w,(\psi,\mathscr{P})$
             end if
55:        else if $\psi = \Box\neg\psi'$ then
             if $(\Box\psi',\mathscr{P}) \notin Label(w)$ then
               $Label(w,(\psi,\mathscr{P}))$
             end if
           else if $\psi = \Box(\psi' \wedge \psi'')$ then
60:          if $(\Box\psi',\mathscr{P}) \in Label(w)$ and $(\Box\psi'',\mathscr{P}) \in Label(w)$ then
               $Label(w,(\psi,\mathscr{P}))$
             end if
           else if $\psi = [+\chi]\psi'$ then
             if $(\psi',\mathscr{P} \cup \{\chi\}) \in Label(w)$ then
65:            $Label(w,(\psi,\mathscr{P}))$
             end if
           else if $\psi = [-\chi]\psi'$ then
             if $(\psi',\mathscr{P} - \{\chi\}) \in Label(w)$ then
               $Label(w,(\psi',\mathscr{P}))$
70:          end if
           end if
         end for
       end if
     end while
75:  if $\phi \in Label(w)$ then
       Return True
     elseReturn False
     end if
```

The algorithm $Model-Check^{\mathscr{P}}$ is the same as algorithm 1 except that all the expressions of the form $Label(w,\psi)$ have to be subsituted in $Model-Check^{\mathscr{P}}$ by $Label(w,(\psi,\mathscr{P}))$ and all the expressions of the form $\psi \in Label(w)$ have to be substituted by $(\psi,\mathscr{P}) \in Label(w)$. The algorithms $Add_1^{\mathscr{P}}$ and $Add_2^{\mathscr{P}}$ are defined in the appendix as algorithms 5 and 6 respectively. The symbol $\Box$ in lines 43-51 stands for $[Send\ \psi]$ or $[Prom\ \alpha]$. We use this notation in order to avoid repeating the same instructions twice.

$|\phi|^1_{\mathscr{P}} + |\phi|^2_{\mathscr{P}} = |\phi|_{\mathscr{P}}$, where $|\phi|_{\mathscr{P}}$ is defined inductively as follows:

$$\begin{array}{lcl@{\qquad}lcl}
|\psi|_{\mathscr{P}} & = & |\psi| & |[+\chi]\phi|_{\mathscr{P}} & = & 1+|\phi|_{\mathscr{P}\cup\{\chi\}} \\
|\neg\phi|_{\mathscr{P}} & = & 1+|\phi|_{\mathscr{P}} & |[-\chi]\phi|_{\mathscr{P}} & = & 1+|\phi|_{\mathscr{P}-\{\chi\}} \\
|\phi\wedge\phi'|_{\mathscr{P}} & = & 1+|\phi|_{\mathscr{P}}+|\phi'|_{\mathscr{P}} & |P_s(Send\ \psi)|_{\mathscr{P}} & = & |[Send\ \psi]Comp|_{\mathscr{P}} \\
|(\chi\in\mathscr{P})|_{\mathscr{P}} & = & 1 & |[Send\ \psi]\phi|_{\mathscr{P}} & = & 1+|\psi|_{\mathscr{P}}+|\phi|_{\mathscr{P}} \\
|BehComp|_{\mathscr{P}} & = & 1+|\bigwedge\limits_{i\to O_s\alpha\in\mathscr{P}}(i\to\alpha)| & |[Prom\ \alpha]\phi|_{\mathscr{P}} & = & 1+|\alpha|_{\mathscr{P}}+|\phi|_{\mathscr{P}}
\end{array}$$

$$\begin{array}{lcl}
|Comp|_{\mathscr{P}} & = & 1+|\bigwedge\limits_{i\to O_s\alpha\in\mathscr{P}} O_s\alpha \wedge \bigwedge\limits_{i\to P_s\alpha\in\mathscr{P}} P_s\alpha \wedge \bigwedge\limits_{i\to O_s\alpha\in\mathscr{P}} (i\to\alpha)| \\
|RegComp|_{\mathscr{P}} & = & 1+|\bigwedge\limits_{i\to O_s\alpha\in\mathscr{P}} O_s\alpha \wedge \bigwedge\limits_{i\to P_s\alpha\in\mathscr{P}} P_s\alpha|
\end{array}$$

Now, one proves by induction on $\phi\in\mathscr{L}_{PL}$ that $|\phi|_{\mathscr{P}} = O(|\phi|+|\mathscr{P}|)$. We only prove the cases $|BehComp|_{\mathscr{P}}$ and $|[-\chi]\phi|_{\mathscr{P}}$: $|BehComp|_{\mathscr{P}} = 1+|\bigwedge\limits_{i\to O_s\alpha\in\mathscr{P}}(i\to\alpha)| = 1+O(|\bigwedge\limits_{\chi\in\mathscr{P}}\chi|) = 1+O(|\mathscr{P}|) = |BehComp|+O(|\mathscr{P}|)$; $|[-\chi]\phi|_{\mathscr{P}} = 1+|\phi|_{\mathscr{P}-\{\chi\}} = 1+|\phi|+O(|\mathscr{P}-\{\chi\}|) = 1+|\phi|+O(|\mathscr{P}|) = O(1+|\phi|+|\mathscr{P}|) = O(1+|\phi|+|\chi|+|\mathscr{P}|) = O(|[-\chi]\phi|+|\mathscr{P}|)$.

Hence, the total running time of the **while** iteration between lines 5 and 74 is in $O((||M||+|\mathscr{P}|)\times(|\phi|+|\mathscr{P}|))$. One proves similarly as above and by induction that the running time of $Add_1^{\mathscr{P}}$ and $Add_2^{\mathscr{P}}$ (i.e. algorithms 5 and 6) is in $O(|\phi|+|\mathscr{P}|)$, taking into account the fact that the instructions of lines 49 and 51 are executed in time $O(1)$ and $O(\mathscr{P})$ respectively. So, finally, the total runing time of algorithm 4 is in time $O((||M||+|\mathscr{P}|)\times(|\phi|+|\mathscr{P}|))$. QED

Example 23 The mechanisms involved in the website example can be better analysed and understood with this new language. In Example 13, the initial situation is compliant with respect to the current privacy policy:

$$\{(M,w),\mathscr{P}_1\} \models Comp.$$

Note that it was not possible to express this with the previous language $\mathscr{L}_{DEDL}$. The Sender then learns by the security administrator that the new privacy policy is $\mathscr{P}_5$. This change boils down to first removing the epistemic norm P_sK_re ($[-P_sK_re]$) and then adding the epistemic norm $\neg P_sK_re$ ($[+\neg P_sK_re]$). The situation is no longer compliant with this new privacy policy because it is not regulatory compliant anymore with the privacy policy $\mathscr{P}_5$:

$$\{(M,w),\mathscr{P}_1\} \models [-P_sK_re][+\neg P_sK_re]\neg RegComp.$$

Therefore, the Sender now has to enforce this new privacy policy $\mathscr{P}_5$ by means of a promulgation. He does so by promulgating the norm $\neg K_re$. That was the process described in Example 22:

$$\{(M,w),\mathscr{P}_5\} \models \neg RegComp \wedge [Prom\ \neg K_re]Comp.$$

We see in the above example that the language $\mathscr{L}_{PL}$ really allows the security monitor to reason about which actions he can perform so that a new privacy policy be enforced or so that the situation be compliant with respect to the privacy policy.

5 Related work

Comparison with the Cuppens-Demolombe logic. Cuppens and Demolombe (1996) extend their original framework (Cuppens, 1993) by using an epistemic deontic logic to model security in databases. Their modal language is actually a restriction of our language $\mathscr{L}_{EDL}$ and is defined as follows:

$$\mathscr{L} : \phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid P_s K_u \psi \mid O_s K_u \psi$$

where ψ ranges over propositional formulae and p over Φ^{ϕ}. Their semantics in terms of Kripke models is the same as ours. However, their accessibility relation for the modality $K_u\phi$ is not reflexive because it stands for "the user *knows* that the database *believes* ϕ" and is therefore the combination of two modalities, the modality for knowledge being reflexive and the modality for belief being serial. Nevertheless, if we assume that our modality K_r is serial and give it the same reading as their reading, then our formalism clearly embeds their formalism. Overall, we improve the Cuppens-Demolombe logic in the following respects:

Actions. Our logic also deals with actions. They just define privacy policies in terms of the permitted and forbidden knowledge of the recipient of information, but they do not derive the permitted messages by combining and reasoning on this knowledge like us.

Obligations. Our logic also deals with obligations. This is a desirable feature since, as Barth et al (2006) notice, privacy laws actually specify which counter measures *should* apply in case a situation is not compliant with a privacy policy. We can express in our logic not only that it is obligatory to know *whether* something holds but also that it is obligatory to send a message.

Meta-policies. Our logic can also express meta-policies. These are policies about how to access the policy itself, like for example $P_s K_r P_s K_r p$. In our logic, we can even send a message stating that it is permitted that the Recipient knows p. The problem is that in some cases some sensitive information may be deduced by the Recipient by disclosing the policy itself.

Policy change. Our logic also deals with change of privacy policy. They cannot express that a new epistemic norm is added or removed from the current privacy policy, as we illustrated it in Example 23.

Planning/Enforcement. Given that their logic does not deal with actions, they cannot *a fortiori* reason about the effects of actions and plan which action should be executed in order to enforce a given privacy policy. This is possible in our logic, and we illustrate it in Example 23.

Given that our approach is based on their model, their solutions to several problems can naturally be transferred in our setting. For example, they show that multi-level security policies which assign a degree of clearance l to formulae ϕ and which might be incomplete can be expressed in their framework by indexing the modality $P_s K_r \phi$ with the degree of clearance l: the formula $P_s K_{rl} \phi$ reads as 'an agent r cleared at level l is explicitly permitted to know that the database believes ϕ'. They also avoid possible conflicts between roles and policies by defining the role of an agent as an index i of the modality $P_s K_{ri} \phi$ and by introducing an external structure on these roles.

Other related work. Languages for access control in security have been used for modeling privacy regulations too (Bishop, 2003). However, they are not easily adapted to the new task, because they do not provide ways of reasoning about the information and about effects of messages. Moreover, they rarely consider the context of communication.

Specific languages for privacy policies have been proposed, but have some limitations. Extensible Access Control Markup Language XACML's policies (Moses, 2005) can lead to obligations, but "obligation" is just an uninterpreted symbol which receives meaning at the point of policy enforcement. Enterprise Privacy Authorization Language EPAL's policies (Karjoth and Schunter, 2002) are concerned with a single sender (the enterprise itself) and a single recipient role. EPAL structures obligations with a subsumption relation rather than allowing to reason about knowledge. The Platform for Privacy Preferences (P3P) language (Cranor, 2002) contains only positive norms and a temporal dimension restricted to opt-in and opt-out conditions.

Bonatti et al (1995) use a similar logical framework for reasoning about security in database access. They explicitly model the beliefs of the user of the database and the actions which change these beliefs. However, again they only use combined epistemic and deontic operators, in the sense that they do not make an explicit distinction between epistemic and deontic modalities, with resulting limitations such as the impossibility to model permissions and obligations about actions. Moreover, the belief change mechanism is superimposed to Kripke semantics, while we use a general dynamic epistemic logic approach and we are also able to change permissions and obligations and not only beliefs. As they do, by distinguishing the point of view of the database (Sender) from the beliefs of the user (Recipient), we could model situations where the sender of information is lying, even if this possibility seems less useful in the context of privacy policies. Finally, we can model meta-policy in our framework, to specify that it is prohibited to know the privacy policy. Differently from their work, we also provide a semantics to meta-policy since we allow nestings of epistemic and deontic modalities.

Barth et al (2006) propose a formalization of the theory of privacy called contextual integrity. They introduce positive and negative norms, depending on whether they refer to actions that are allowed or disallowed. Temporal conditions are modelled by means of linear temporal logic with past and future operators to express, for example, that certain information may be disclosed only if the subject mentioned has previously given permission or that if certain information is made public, notification must be sent to the concerned party. These norms are interpreted in a model of agents who respect the norms if the trace history of their communication satisfies a temporal formula constructed from the norms by taking the disjunction over positive norms and the conjunction over negative norms. Their language constitutes an advancement with respect to other policy languages, both for the temporal aspect and for including a relation enabling agents to combine messages to compute additional information about the subject, (e.g., computing postal code from postal address), elucidating the notion of a "data hierarchy" found in P3P and EPAL. However, their privacy policies cannot be changed. On the other hand, we do not consider the temporal aspect yet: to incorporate this aspect in our model it might be necessary to resort to an epistemic temporal logic, as in Pacuit et al (2006). Pacuit et al (2006) also introduce a logic combining deontic and epistemic notions but they can express only particular epistemic norms called knowledge-based obligations of the form $K_r\phi \to O_s\psi$, where ψ does not contain any knowledge operator.

DeYoung et al (2010) extend Barth et al (2006) to apply the model to privacy laws such as HIPAA and GLBA, including support for self-reference, purposes of uses and disclosures, some dynamics concerning roles and beliefs of principals. This extension goes in the direction of our work, showing the need to introduce dynamics and epistemic operators, while we leave for future work the treatment of self-references such as “except as otherwise provided in this subchapter”.

A problem of Barth et al (2006) is the obscurity of the formalism used to model legal norms, which in turn present ambiguities and difficulties. To cope with this problem, Lam et al (2009) propose a more readable formalism based on logic programming. Our modal logic aims at improving readability too, but at the same time it allows to study precisely the properties of the deontic operators.

Logic or logic programming used in the papers discussed above (see also Barker (2002)) are not the only methodologies to formalize privacy policies. May et al (2006) use an extension of access control matrix operations to include operations for notification and logging and constructs that ease the mapping between legal and formal language. They apply their methodology to HIPAA policies of health insurance. Nielson and Nielson (2007) propose to use π-calculus for privacy in the context of service oriented architectures.

A further issue in privacy is the interaction between policies and the organizations which have to enforce them. This is addressed, e.g., by Barth et al (2007) and Kanovich et al (2007). Our plan to address this problem is to extend the modal language to a multi-agent language in order to express obligations, beliefs, knowledge and *goals* of the different parties involved.

In dynamic epistemic logic, Balbiani et al (2009) focus in a multi-agent setting on the notion of permission to announce. They provide a sound, complete and decidable logic by enriching public announcement logic with the operator $P(\psi,\phi)$ which reads ‘after ψ has been publicly announced, it is permitted to say ϕ’. There is no privacy policy nor compliance, although the specification of such a policy could be somehow derived via the specification of their operator $P(\psi,\phi)$ (whose first argument handles the dynamic character of the situations they consider). But as in all the other approaches mentioned, the (implicit) privacy policy is specified directly on the announcements/actions and the epistemic character of the situations they consider does not really play a role. Finally, in their logic, privacy policies cannot change and they do not have a notion of obligatory announcement or enforcement.

Other work in dynamic deontic logic (Meyer, 1988; Van der Meyden, 1996) extends dynamic logic with obligations using a violation constant, and define that an action is obliged if the absence of occurence of the action leads to a violation state. They are interested in particular in logical relations among obligations, and therefore in obligations of complex actions such as sequences of actions.

This paper is an extended and revised version of Aucher et al (2010b). In a companion paper (Aucher et al, 2010a) we show how to express the distinction between descriptive and prescriptive obligations in dynamic epistemic deontic logic.

6 Conclusion

Summary. Classical problems of security such as the Chinese wall problem (Brewer and Nash, 1989) need a more fined-grained analysis taking into account the dynamic context. It might be permitted to know something but not to send a message containing this piece of information, depending on the particular situation. In our website example, which is an instance of this Chinese wall problem, even if it is permitted to know the mapping of users with their respective numbers ($P_s K_r c$) it is not permitted to send it ($P_s(Send\ c)$) if the mapping of numbers with the visited websites (e) is already known or has been sent before. On the other hand, privacy policies are often defined in terms of permitted messages (actions), for example in traditional access control languages (Bishop, 2003; Cranor, 2002; Moses, 2005; Karjoth and Schunter, 2002), but this direct representation of privacy policies in terms of permitted messages is difficult to manage in a dynamic context. Instead, in this paper we derived the permitted (and obligatory) messages from 'static' privacy policies defined in terms of permitted and obligatory knowledge. To specify and reason about such privacy policies, we extended a multi-modal logic introduced by Cuppens and Demolombe with update operators modeling the dynamics of both knowledge and privacy policies. We then defined a richer language that allows us to check whether a situation is compliant with respect to a privacy policy and to determine which actions should be executed in order to enforce a privacy policy. We axiomatized and proved the decidability of our logic and we studied its complexity properties.

Potential application. In order to use this logic in real situations, the security monitor (Sender) would need to implement a privacy model representing the current epistemic and deontic state of affairs. He could then check compliance with respect to a given policy and determine which actions can and should be done by model checking this privacy model with algorithm 4. The low complexity of this algorithm is a sign that this kind of application of our logic is feasible and realistic.

Future work. A topic for further research is to deal with multi-agent scenarios involving more agents than just a Sender and a Recipient, each agent having its own privacy policy to comply with. Many privacy issues deal with more persons than only a sender and receiver. For example, you may be permitted to know your medical file, while it may not be permitted that someone not being a doctor sends you your medical file. Or you may be permitted to know the budget, while at the same time being forbidden to know who else knows the budget. You may tell your poker agent never to inform your opponents about your cards to hide your bluff strategy, or tell your medical database that it may give only generic information, not information about specific individuals… In this multi-agent setting, the distinction between permitted action and knowledge is still relevant: you may tell your web agent never to reveal to your boss which websites you have visited either by specifying that your boss may never know which websites you visited, or by specifying directly that it may not send your boss a list of the websites you have visited.

Another topic for further research is to enrich the dynamics to allow not only operations which promulgate norms but also operations which contract (derogate) or revise norms. Indeed, even if our language $\mathscr{L}_{PL}$ allows us to remove or add epistemic norms from/to the privacy policy, sometimes promulgation might not be enough to enforce these new epistemic norms and we might need to resort to more fine-grained dynamics like contraction or revision, as in AGM theory of belief change (Alchourrón et al, 1985). This enrichment of the dynamics would allow for example to model declassification of documents.

Finally, we plan to study the consequences of our logic for deontic logic. The distinction between permitted and forbidden knowledge and permitted and forbidden actions is known in the deontic logic literature as the distinction between ought-to-be and ought-to-do, expressed respectively by $O(p)$ and $O(\alpha)$, where O is a modal operator, p a propositions, and α an action. It is well known that the translation of ought-to-be to ought-to-do and vice versa is a non-trivial challenge (Horty, 2001).

Acknowledgements We thank the anonymous reviewers of this paper for their extensive and helpful comments.

References

Alchourrón C, Gärdenfors P, Makinson D (1985) On the logic of theory change: Partial meet contraction and revision functions. Journal of Symbolic Logic 50(2):510–530

Anderson A (1958) A reduction of deontic logic to alethic modal logic. Mind 67:100–103

Åqvist L (1967) Good samaritans, contrary-to-duty imperatives, and epistemic obligations. Nôus 1:361–379

Aucher G, Boella G, van der Torre L (2010a) Prescriptive and descriptive obligations in dynamic epistemic deontic logic. In: AI approaches to the complexity of legal systems (AICOL 2009), Springer, Berlin, LNAI, vol 6237, pp 150–161

Aucher G, Boella G, van der Torre L (2010b) Privacy policies with modal logic: the dynamic turn. In: Governatori G, Sartor G (eds) Deontic Logic in Computer Science (DEON 2010), Springer, Berlin, LNCS, vol 6181, pp 196–213

Balbiani P, van Ditmarsch H, Seban P (2009) Reasoning about permitted announcements. In: ESSLLI 2009 workshop Logical Methods for Social Concepts, Bordeaux

Baltag A, Moss L (2004) Logic for epistemic programs. Synthese 139(2):165–224

Baltag A, Moss L, Solecki S (1998) The logic of common knowledge, public announcement, and private suspicions. In: Gilboa I (ed) Proceedings of the 7th conference on theoretical aspects of rationality and knowledge (TARK98), pp 43–56

Barker S (2002) Protecting deductive databases from unauthorized retrieval and update requests. Data and Knowledge Engineering 43(3):295–315

Barth A, Datta A, Mitchell JC, Nissenbaum H (2006) Privacy and contextual integrity: framework and applications. In: IEEE Symposium on Security and Privacy, IEEE Computer Society, Los Alamitos (CA), pp 184–198

Barth A, Mitchell JC, Datta A, Sundaram S (2007) Privacy and contextual integrity: framework and applications. In: IEEE Computer Security Foundations Symposium CSF'07, IEEE Computer Society, Los Alamitos (CA), pp 279 – 294

Bishop M (2003) Computer Security: Art and Science. Addison Wesley Professional

Blackburn P, de Rijke M, Venema Y (2001) Modal Logic, Cambridge Tracts in Computer Science, vol 53. Cambridge University Press

Boella G, Governatori G, Rotolo A, van der Torre L (2010) A logical understanding of legal interpretation. In: Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010, AAAI Press

Bonatti P, Kraus S, Subrahmanian V (1995) Foundations of secure deductive databases. IEEE Transactions on Knowledge and Data Engineering 7(3):406–422

Brewer DFC, Nash MJ (1989) The chinese wall security policy. In: IEEE Symposium on Security and Privacy, IEEE Computer Society, Los Alamitos (CA), pp 206–214

Castañeda HN (1981) The paradoxes of deontic logic: the simplest solution to all of them in one fell swoop. In: Hilpinen R (ed) New studies in deontic logic: norms, actions, and the foundations of ethics, Synthese library, Reidel publishing co., pp 37–86

Castañeda HN (1988) Knowledge and epistemic obligation. Philosophical perspectives 2:211–233

Cranor L (2002) Web Privacy with P3P. O'Reilly and Associates Inc.

Cuppens F (1993) A logical formalization of secrecy. In: IEEE Computer Security Foundations Workshop CSFW'93, IEEE Computer Society, Los Alamitos (CA)

Cuppens F, Demolombe R (1996) A deontic logic for reasoning about confidentiality. In: Deontic Logic, Agency and Normative Systems, Third International Workshop on Deontic Logic in Computer Science (DEON 1996), Springer, Berlin

Cuppens F, Demolombe R (1997) A modal logical framework for security policies. In: Ras Z, Skowron A (eds) Foundations of Intelligent Systems, 10th International Symposium, ISMIS '97, Springer, Berlin, LNCS, vol 1325, pp 579–589

DeYoung H, Garg D, Jia L, Kaynar D, Datta A (2010) Experiences in the logical specification of the HIPAA and GLBA privacy laws. In: Proceedings of the 9th annual ACM Workshop on Privacy in the Electronic Society, ACM, New York, NY, USA, WPES '10, pp 73–82

van Ditmarsch H, van der Hoek W, Kooi B (2007) Dynamic Epistemic Logic, Synthese library, vol 337. Springer, Berlin

Fagin R, Halpern J, Moses Y, Vardi M (1995) Reasoning about knowledge. MIT Press

Federal Trade Commission (1998) Children's Online Privacy Protection Act of 1998 (COPPA). http://www.ftc.gov/ogc/coppa1.htm

Federal Trade Commission (1999) Gramm-Leach-Bliley Act (GLBA). http://www.ftc.gov/privacy/glbact/glbsub1.htm

Halpern J, Moses Y (1992) A guide to completeness and complexity for modal logics of knowledge and belief. Artificial Intelligence 54(3):311–379

Hinke TH (1988) Database inference engine design approach. In: Database Security DB-Sec, pp 247–262

Horty J (2001) Agency and deontic logic. Oxford University Press, USA

Kanovich M, Rowe P, Scedrov A (2007) Collaborative planning with privacy. In: IEEE Computer Security Foundations Symposium CSF'07, IEEE Computer Society, Los Alamitos (CA), pp 265–278

Karjoth G, Schunter M (2002) A privacy policy model for enterprises. In: IEEE Computer Security Foundations Workshop CSFW'02, IEEE Computer Society, Los Alamitos (CA)

Lam P, Mitchell J, Sundaram S (2009) A formalization of HIPAA for a medical messaging system. In: Trust, Privacy and Security in Digital Business, TrustBus 2009, Springer, Berlin, pp 73 – 85

May M, Gunter C, Lee I (2006) Privacy APIs: Access control techniques to analyze and verify legal privacy policies. In: IEEE Computer Security Foundations Symposium CSF'06, IEEE Computer Society, Los Alamitos (CA), pp 85–97

Van der Meyden R (1996) The dynamic logic of permission. Journal of Logic and Computation 6:465–479

Meyer JJC (1988) A different approach to deontic logic: deontic logic viewed as a variant of dynamic logic. Notre Dame Journal of Formal Logic 29(1)

Moses T (2005) Extensible Access Control Markup Language (XACML) version 2.0. http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

Nielson H, Nielson F (2007) A flow-sensitive analysis of privacy properties. In: IEEE Computer Security Foundations Symposium CSF'07, IEEE Computer Society, Los Alamitos (CA), pp 249–264

Office for Civil Rights (2003) Summary of the HIPAA privacy rule. `http://www.hhs.gov/ocr/privacy/hipaa/understanding/summary/privacysummary.pdf`

Pacuit E, Parikh R, Cogan E (2006) The logic of knowledge based obligation. Synthese 149(2):311–341

Sahlqvist H (1975) Completeness and correspondence in the first and second order semantics for modal logics. In: Kanger S (ed) Proceedings of the 3rd Scandinavian Logic Symposium 1973, North Holland, no. 82 in Studies in Logic

Sweeney L (2002) k-anonymity: a model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10(5):557–570

United Nations General Assembly (1948) Universal Declaration of Human Rights (UDHR). `http://www.un.org/en/documents/udhr/index`

Warren S, Brandeis L (1890) The right to privacy. Harvard Law Review 193(4):193–220

Westin A (1968) Privacy and freedom, 5th edn. Atheneum, New York, U.S.A.

A An extension of Castañeda's deontic logic

In this appendix, we give an extension of our epistemic deontic logic embedding Castañeda's deontic logic (Castañeda, 1981). Starting from a linguistic analysis, the insight of Castañeda is to acknowledge the grammatical duality of expressions depending on whether they are within or without the scope of an obligation operator. This leads him formally to introduce two sets of formulas: circumstances which cannot *alone* be the foci of deontic operators, unlike what he calls practitions. The former are usually expressed grammatically in the indicative form and the latter are usually expressed grammatically in the infinitive/subjunctive form. For example, "Freud cures Anna O" in the indicative form is a circumstance, but the same sentence in "*it is obligatory* that Freud cures Anna O" in subjunctive/infinitive form is a practition. Just as practitions are the foci of deontic operators, circumstances are dually the foci of knowledge operators, as pointed out by Castañeda (1988). In that respect, note that an expression ϕ in the scope of a knowledge operator $K_r\phi$ is always in the indicative form and never in the subjunctive/infinitive form, even if $K_r\phi$ is in the scope of a deontic operator O.

We extend Castañeda (1988)'s intuition to the context of epistemic permissions and obligations. In a deontic setting the reading of the term knowledge or belief can be twofold: either as a circumstance or as a practition. On the one hand, in the sentence "it is obligatory that John *knows* / for John *to know* that there is an infinity of prime numbers", the verb 'to know' is the focus of a deontic operator and is in the subjunctive/infinitive form. On the other hand, the sentence "John *knows* that there is an infinity of prime numbers" alone describes a circumstance and the interpretation of the verb 'to know' in the indicative form matches the one usually studied in epistemic logic. The former use of the term knowledge within the scope of a deontic operator is not studied in epistemic logic. For these reasons we enrich the language of Castañeda with two knowledge modalities, one for circumstances K_r and the other one for epistemic practitions K'_r. This allows us to express new kinds of statements which cannot be expressed directly with our language $\mathscr{L}_{DEDL}$, such as

$$O_s(K_r\phi \to K'_r\psi) \tag{10}$$

Formula 10 reads as 'it is obligatory for the Sender that, if the Recipient knows ϕ then he also knows ψ'. In our language $\mathscr{L}_{DEDL}$, this statement would be expressed by the intuitively equivalent but formally different expression:

$$K_r\phi \to O_sK'_r\psi \tag{11}$$

Formula 11 reads as 'if the Recipient knows ϕ then it is obligatory for the Sender that the Recipient also knows ψ'. Both of these formulations 10 and 11 are intuitively equivalent, but they are both intuitively different from the following expression:

$$O_s(K'_r\phi \to K'_r\psi) \tag{12}$$

Formula 12 reads as ‘if it is obligatory for the Sender that the Recipient knows ϕ then it is also obligatory for the Sender that the Recipient knows ψ’. Note that this last formulation 12 is itself also quite similar to the reading of the formula $O_s K'_r \phi \to O_s K'_r \psi$. Formulas 10 is not a well-formed formula of our language $\mathscr{L}_{DEDL}$, but it is a well-formed formula of the following language $\mathscr{L}_{DL}$.

Definition 13 Let Φ^α be a set of propositionnal letters. The language $\mathscr{L}_{DL} = \mathscr{L}^{\phi'}_{EDL} \cup \mathscr{L}^{\alpha'}_{EDL}$, whose formulas are denoted ϕ^* in general, is defined inductively as follows.

$$\mathscr{L}^{\phi'}_{EDL} : \phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid K_r\phi \mid O_s\alpha$$
$$\mathscr{L}^{\alpha'}_{EDL} : \alpha ::= \beta \mid K_r\phi \mid \neg\alpha \mid \alpha \wedge \alpha \mid \alpha \wedge \phi \mid \phi \wedge \alpha$$

where p ranges over Φ^ϕ and β over Φ^α.

The only difference with the language $\mathscr{L}_{DL}$ is that we now have pure practitions Φ^α and that practitions can now be of the form $\phi \wedge \alpha$ or $\phi \to \alpha$ where ϕ is a circumstance. Pure practitions Φ^α are expressions in the scope of a deontic operator that cannot be expressed with a knowledge operator, such as ‘to cure Anna O’ in ‘it is obligatory to cure Anna O’. Therefore, just as epistemic practitions, they are in the subjunctive/infinitive form. Moreover, with this definition of practitions we can also express formulas of the form $O_s(\phi \to \alpha)$ and in particular Formula 10 above. Obviously, we would like to have the following validity:

$$\models O_s(\phi \to \alpha) \leftrightarrow (\phi \to O_s\alpha)$$

which is a generalization to the epistemic case of Castañeda’s key validity. For example, “it is obligatory that if Freud knows that Anna O is sick, then he cures her” ($O_s(K_r\phi \to \alpha)$) has the same meaning as “if Freud knows that Anna O is sick, then it is obligatory that he cures her” ($K_r\phi \to O_s\alpha$). This would also make Formulas 10 and 11 formally equivalent. To obtain this validity, we need to add an extra condition $(*)$ in our definition of EDL-model and so define EDL-model’.

Definition 14 An *EDL-model’* M is a tuple $M = (W, D, R, R', V)$, where W is a non-empty set of possible worlds, $R : W \to 2^W$, $R' : W \to 2^W$ and $D : W \to 2^W$ are accessibility relations on W, D being serial. $V : \Phi^\phi \cup \Phi^\alpha \to 2^W$ is a valuation such that:

for all $w \in W$, all $v, v' \in D(w) \cup \{w\}$, (M, v) is RD-bisimilar to (M, v').[7] $(*)$

The truth conditions are defined as in Definition 14.

The semantic condition $(*)$ intuitively means that the (epistemic) context where a normative system applies is fixed. One can easily show that any Castañeda model (Castañeda, 1981) can be embedded into an EDL-model’, in the sense that the Castañeda model and the corresponding EDL-model’ satisfy the same formulas of $\mathscr{L}'_{EDL}$ without epistemic operators K_r or K'_r. One can also show that the semantics of $\mathscr{L}'_{EDL}$ is *sound and complete* with respect to the logic L_{EDL} to which we add the axiom scheme $\vdash O_s(\phi \to \alpha) \leftrightarrow (\phi \to O_s\alpha)$.

Theorem 7 *The semantics of $\mathscr{L}'_{EDL}$ is* sound and complete *with respect to the logic L'_{EDL} axiomatized as follows. The symbol K below stands either for K_r or K'_r.*

A_1	*All propositional tautologies based on Φ^ϕ*
A_2	$\vdash O_s(\phi \to \alpha) \leftrightarrow (\phi \to O_s\alpha)$
A_3	$\vdash O_s\alpha \to P_s\alpha$
A_4	$\vdash O_s(\alpha \to \alpha') \to (O_s\alpha \to O_s\alpha')$
A_5	$\vdash K(\phi^* \to \psi^*) \to (K\phi^* \to K\psi^*)$
R_1	*If* $\vdash \alpha$ *then* $\vdash O_s\alpha$
R_2	*If* $\vdash \phi^*$ *then* $\vdash K\phi^*$
R_3	*If* $\vdash \phi^* \to \psi^*$ *and* $\vdash \phi^*$ *then* $\vdash \psi^*$

[7] Two pointed models (M, v) and (M', v') are RD-bisimilar if there is a relation on $W \times W'$ satisfying the base condition for Φ^ϕ and the back and forth conditions for R and D (see footnote 3 or Blackburn et al (2001) for details).

Proof Soundness is routine. We prove completeness by building the canonical model of our logic. Let W be the set of all maximal $\mathsf{L}'_{\mathsf{EDL}}$-consistent subsets of $\mathscr{L}_{DL}$. For all $\Gamma,\Gamma' \in W$, we set $\Gamma' \in R(\Gamma)$ iff for all $K_r\phi \in \Gamma, \phi \in \Gamma'$. We define O_s and R' similarly. Besides, for all $\Gamma \in W$, $\Gamma \in V(p)$ iff $p \in \Gamma$, and $\Gamma \in V(\beta)$ iff $\beta \in \Gamma$. We have therefore defined the canonical model $M = (W,D,R,R',V)$. We now show by induction on ϕ the 'truth lemma': for all $\Gamma \in W$ and $\phi \in \mathscr{L}_{DL}$, $M,\Gamma \models \phi$ iff $\phi \in \Gamma$ (1). If $\Gamma = p$ then (1) holds. The other boolean cases work by induction hypothesis. Assume $\phi = K_r\phi'$. If $K_r\phi' \in \Gamma$ then for all $\Gamma' \in R(\Gamma)$, $\phi' \in \Gamma'$ by definition of R. So $M,\Gamma' \models \phi'$ for all $\Gamma' \in R(\Gamma)$ by induction hypothesis, i.e., $M,\Gamma \models K_r\phi'$. If $M,\Gamma \models K_r\phi'$ then assume that $S \subseteq \{\phi \in \mathscr{L}_{DL} \mid K_r\phi \in \Gamma\} \cup \{\neg\phi'\}$ is consistent. It follows that there is $\Gamma^0 \in W$ such that $S \subseteq \Gamma^0$. So there is $\Gamma^0 \in R(\Gamma)$ such that $\neg\phi' \in \Gamma^0$. Therefore $M,\Gamma \models \neg K_r\phi'$ which is absurd. So S is inconsistent and so there must be $\phi^1,\dots,\phi^n \in S$ such that $\vdash (\phi^1 \wedge \dots \wedge \phi^n) \to \phi'$. By R_2 and A_5 we get $\vdash (K_r\phi^1 \wedge \dots \wedge K_r\phi^n) \to K_r\phi'$ and because $K_r\phi^i \in \Gamma$, we finally have $K_r\phi' \in \Gamma$. The proof is similar for the operators O_s and K'_r. One can also show that D is serial.

Now we have to show that condition $(**)$ holds in our canonical model M. We first show that for all $\Gamma \in W$, all $\Gamma',\Gamma'' \in D(\Gamma)\cup\{\Gamma\}$, $\Gamma' \leftrightsquigarrow \Gamma''$,i.e., for all $\phi \in \mathscr{L}^{\phi'}_{EDL}$, $\phi \in \Gamma'$ iff $\phi \in \Gamma''$. Let $\phi \in \mathscr{L}^{\phi'}_{EDL}$ and assume $\phi \in \Gamma'$. If $\phi \notin \Gamma$ then $\neg\phi \in \Gamma$, and $O_s\alpha \in \Gamma$ for some $\alpha \in \mathscr{L}^{\alpha}_{EDL}$. So $M,\Gamma \models \neg\phi \wedge O_s\alpha$, therefore $M,\Gamma \models O_s(\neg\phi \wedge \alpha)$. Then $M,\Gamma' \models \neg\phi \wedge \alpha$, and so $\neg\phi \in \Gamma'$. This is impossible, so $\phi \in \Gamma$. By the same reasoning we get that $\phi \in \Gamma''$. Likewise vice versa. We now show that $\leftrightsquigarrow$ is a RD-bisimulation relation. Assume $\Gamma \leftrightsquigarrow \Gamma'$. The base case for Φ^{ϕ} clearly works. We prove the forth condition for R. Let $\Gamma_1 \in R(\Gamma)$ and let $\Gamma_1^* = \{\phi \in \mathscr{L}^{\phi'}_{EDL} \mid \phi \in \Gamma_1\}$ and assume that for all $\Gamma'_1 \in R(\Gamma')$ it is not the case that $\Gamma_1 \leftrightsquigarrow \Gamma'_1$, i.e., $\Gamma_1^* \nsubseteq \Gamma'_1$. Let $S_1 = \Gamma_1^* - \bigcup\limits_{\Gamma'_1 \in R(\Gamma')} \Gamma'_1$ and let us define $S = S_1 \cup S_2$ where $S_2 = \{\phi \in \mathscr{L}^{\phi'}_{EDL} \mid K_r\phi \in \Gamma\}$. S is consistent, because $S \subseteq \Gamma_1$. So there is $\Gamma_2 \in W$ such that $S \subseteq \Gamma_2$. But $\{\phi \in \mathscr{L}^{\phi'}_{EDL} \mid K_r\phi \in \Gamma'\} = \{\phi \in \mathscr{L}^{\phi'}_{EDL} \mid K_r\phi \in \Gamma'\} = \{\phi \in \mathscr{L}^{\phi'}_{EDL} \mid K_r\phi \in \Gamma\}$ because $\Gamma \leftrightsquigarrow \Gamma'$. $\Gamma_2 \in R(\Gamma')$ and $S_1 \subseteq \Gamma_2$ which is impossible by assumption. So there is $\Gamma'_1 \in R(\Gamma)$ such that $\Gamma^* \subseteq \Gamma'_1$, i.e., such that $\Gamma_1 \leftrightsquigarrow \Gamma'_1$. The same reasoning applies for the back condition. It also applies for the back and forth conditions for D by replacing S_2 by $S'_2 = \{\alpha \in \mathscr{L}^{\alpha'}_{EDL} \mid O_s\alpha \in \Gamma\}$. QED

Axioms A_1 to A_4 and rules R_1 and R_3 provide an alternative axiomatization of Castañeda's language. We can then derive in this logic the following theorems. In particular, note that our notion of knowledge is truthful, even if it was not explicitly mentionned in the axiomatization.

Proposition 5 *For all* $\phi \in \mathscr{L}_{EDL}$,

$$\vdash K'_r\phi \to \phi \tag{13}$$

$$\vdash K_r\phi \to \phi \tag{14}$$

$$\vdash O_sK'_r\phi \to \phi \tag{15}$$

$$\vdash \neg P_sK'_r\phi \to \phi \tag{16}$$

Equation 16 allows us to derive that as a result of informing the recipient that he should not know that ϕ holds, this very Recipient actually learns that ϕ holds. Indeed, as a result of sending this message, the Recipient knows that he should not know ϕ ($K_r\neg P_sK'_r\phi$), and therefore by application of Equation 16, he also knows that ϕ ($K_r\phi$). This derivation was not possible in our logic L_{DEDL}, as we noted it in Example 9.

B Algorithms $Add_1^{\mathscr{P}}$ and $Add_2^{\mathscr{P}}$

Algorithms $Add_1^{\mathscr{P}}$ and$Add_2^{\mathscr{P}}$ below (i.e. algorithms 5 and 6) are called in algorithm 4. They are adapted from algorithms Add_1 and Add_2 (i.e. algorithms 2 and 3) to take into account the presence of the privacy policy $\mathscr{P}$ in the language $\mathscr{L}_{PL}$.

Algorithm 5 $Add_1^{\mathscr{P}}(S,(\phi,\mathscr{P}))$

Input: A Stack S, a formula $\phi \in \mathscr{L}_{PL}$ and a privacy policy $\mathscr{P}$

Output: The stack S added with the subformulas of ϕ, ignoring the subformulas ψ and α of ϕ appearing in dynamic operators $[Send\ \psi]$, $[Prom\ \alpha]$ and $P_s(Send\ \psi)$, and replacing the subformulas $RegComp, BehComp, Comp$ and $P_s(Send\ \psi)$ by their definition.

```
    $Push(S,(\phi,\mathscr{P}))$
    if $\phi = \psi \wedge \psi'$ then
      $Add_1^{\mathscr{P}}(S,(\psi,\mathscr{P}))$
 5:   $Add_1^{\mathscr{P}}(S,(\psi',\mathscr{P}))$
    else if $\phi = K_r\psi, \neg\psi$ then
      $Add_1^{\mathscr{P}}(S,(\psi',\mathscr{P}))$
    else if $\phi = O_s\alpha$ then
      $Add_1^{\mathscr{P}}(S,(\alpha,\mathscr{P}))$
10: else if $\phi = (\chi \in \mathscr{P})$ then
      $Add_1^{\mathscr{P}}(S,((\chi \in \mathscr{P}),\mathscr{P}))$
    else if $\phi = [Send\ \psi]p$ then
      $Add_1^{\mathscr{P}}(S,(p,\mathscr{P}))$
    else if $\phi = [Send\ \psi]\psi' \wedge \psi''$ then
15:   $Add_1^{\mathscr{P}}(S,([Send\ \psi]\psi',\mathscr{P}))$
      $Add_1^{\mathscr{P}}(S,([Send\ \psi]\psi'',\mathscr{P}))$
    else if $\phi = [Send\ \psi]\neg\psi'$ then
      $Add_1^{\mathscr{P}}(S,([Send\ \psi]\psi',\mathscr{P}))$
    else if $\phi = [Send\ \psi]O_s\alpha$ then
20:   $Add_1^{\mathscr{P}}(S,(O_s\alpha,\mathscr{P}))$
    else if $\phi = [Send\ \psi]K_r\psi'$ then
      $Add_1^{\mathscr{P}}(S,([Send\ \psi]\psi',\mathscr{P}))$
      $Add_1^{\mathscr{P}}(S,(K_r\psi',\mathscr{P}))$
    else if $\phi = [Prom\ \alpha]p$ then
25:   $Add_1^{\mathscr{P}}(S,(p,\mathscr{P}))$
    else if $\phi = [Prom\ \alpha](\chi \in \mathscr{P})$ then
      $Add_1^{\mathscr{P}}(S,((\chi \in \mathscr{P}),\mathscr{P}))$
    else if $\phi = [Prom\ \alpha]\psi \wedge \psi'$ then
      $Add_1^{\mathscr{P}}(S,([Prom\ \alpha]\psi,\mathscr{P}))$
30:   $Add_1^{\mathscr{P}}(S,([Prom\ \alpha]\psi',\mathscr{P}))$
    else if $\phi = [Prom\ \alpha]\neg\psi$ then
      $Add_1^{\mathscr{P}}(S,([Prom\ \alpha]\psi,\mathscr{P}))$
    else if $\phi = [Prom\ \alpha]O_s\alpha'$ then
      $Add_1^{\mathscr{P}}(S,([Prom\ \alpha]\alpha',\mathscr{P}))$
35:   $Add_1^{\mathscr{P}}(S,(O_s\alpha',\mathscr{P}))$
    else if $\phi = [Prom\ \alpha]K_r\psi$ then
      $Add_1^{\mathscr{P}}(S,(K_r\psi,\mathscr{P}))$
    else if $\phi = RegComp$ then
      $Add_1^{\mathscr{P}}(S,(\bigwedge_{i\to P_s\alpha\in\mathscr{P}} P_s\alpha \wedge \bigwedge_{i\to O_s\alpha\in\mathscr{P}} O_s\alpha,\mathscr{P}))$
40: else if $\phi = BehComp$ then
      $Add_1^{\mathscr{P}}(S,(\bigwedge_{i\to O_s\alpha\in\mathscr{P}} i \to \alpha,\mathscr{P}))$
    else if $\phi = Comp$ then
      $Add_1^{\mathscr{P}}(S,RegComp,\mathscr{P}))$
      $Add_1^{\mathscr{P}}(S,(\bigwedge_{i\to O_s\alpha\in\mathscr{P}} O_s\alpha \wedge \bigwedge_{i\to P_s\alpha\in\mathscr{P}} P_s\alpha \wedge \bigwedge_{i\to O_s\alpha\in\mathscr{P}} (i \to \alpha),\mathscr{P}))$
45: else if $\phi = P_s(Send\ \psi)$ then
      $Add_1^{\mathscr{P}}(S,([Send\ \psi](\bigwedge_{i\to O_s\alpha\in\mathscr{P}} (i \to \alpha) \wedge \bigwedge_{i\to O_s\alpha\in\mathscr{P}} O_s\alpha \wedge \bigwedge_{i\to P_s\alpha\in\mathscr{P}} P_s\alpha),\mathscr{P}))$
    else if $\phi = [+\chi]\phi$ then
      $\mathscr{P} \leftarrow \mathscr{P} \cup \{\chi\}$
      $Add_1^{\mathscr{P}}(S,(\phi,\mathscr{P}))$
50: else if $\phi = [-\chi]\phi$ then
      $\mathscr{P} \leftarrow \mathscr{P} - \{\chi\}$
      $Add_1^{\mathscr{P}}(S,(\phi,\mathscr{P}))$
    end if
```

Algorithm 6 $Add_2^{\mathscr{P}}(S,\phi,\mathscr{P})$

Input: A stack S, $\phi \in \mathscr{P}_{PL}$, and a privacy policy $\mathscr{P}$

Output: The stack S added with the subformulas ψ and α of ϕ appearing in dynamic operators of the form $[Send\ \psi]$, $[Prom\ \alpha]$ or $P_s(Send\ \psi)$

```
    if $\phi = \psi \wedge \psi'$ then
      $Add_2^{\mathscr{P}}(S,(\psi,\mathscr{P}))$
      $Add_2^{\mathscr{P}}(S,(\psi',\mathscr{P}))$
 5: else if $\phi = O_s\alpha$ then
      $Add_2^{\mathscr{P}}(S,(\alpha,\mathscr{P}))$
    else if $\phi = K_r\psi, \neg\psi$ then
      $Add_2^{\mathscr{P}}(S,(\psi,\mathscr{P}))$
    else if $\phi = [+\chi]\psi$ then
10:   $Add_2^{\mathscr{P}}(S,(\psi,\mathscr{P} \cup \{\chi\}))$
    else if $\phi = [-\chi]\psi$ then
      $Add_2^{\mathscr{P}}(S,(\psi,\mathscr{P} - \{\chi\}))$
    else if $\phi = [Send\ \psi]\psi'$ then
      $Push(S,\psi)$
15:   $Add_1^{\mathscr{P}}(S,(\psi,\mathscr{P}))$
      $Add_2^{\mathscr{P}}(S,(\psi,\mathscr{P}))$
      $Add_2^{\mathscr{P}}(S,(\psi',\mathscr{P}))$
    else if $\phi = [Prom\ \alpha]\psi$ then
      $Push(S,\alpha)$
20:   $Add_1^{\mathscr{P}}(S,(\alpha,\mathscr{P}))$
      $Add_2^{\mathscr{P}}(S,(\alpha,\mathscr{P}))$
      $Add_2^{\mathscr{P}}(S,(\psi,\mathscr{P}))$
    else if $\phi = P_s(Send\ \psi)$ then
      $Push(S,\psi)$
25:   $Add_1^{\mathscr{P}}(S,(\psi,\mathscr{P}))$
      $Add_2^{\mathscr{P}}(S,(\psi,\mathscr{P}))$
    end if
```