Run-Time Monitoring for Privacy-Agreement Compliance[*]

S. Benbernou, H. Meziane, and M.S. Hacid

LIRIS, University Claude Bernard Lyon1, France
{sbenbern,mshacid}@liris.univ-lyon1.fr,meziane_has@yahoo.fr

Abstract. This paper addresses the problem of monitoring the compliance of privacy agreement that spells out a consumer's privacy rights and how consumer private information must be handled by the service provider. A state machine based model is proposed to describe the Private Data Use Flow (PDUF) toward monitoring which can be used by privacy analyst to observe the flow and capture privacy vulnerabilities that may lead to non-compliance. The model is built on top of (i) properties and timed-related privacy requirements to be monitored that are specified using LTL (Linear Temporal logic) (ii) a set of identified privacy misuses.

1 Introduction

Numerous web services targeting consumers have accompanied the rapid growth of the Internet. Web services are available for banking, shopping, learning, healthcare, and government online. Most of these services require the consumer's personal information in one form or another which makes the service provider in the possession of a large amount of consumer private information along with the accompanying concerns over potential loss of consumer privacy. While *access control* aspect of security and privacy is well understood, it is unclear of how to do *usage control*. In response to the privacy concerns quoted above, in [4] we proposed a *privacy agreement* model that spells out a set of requirements related to consumer's privacy rights in terms of how service provider must handle privacy information. The properties and private requirements can be checked at a design time prior to execution, however, the monitoring of the requirements at run-time has strong motivations since those properties can be violated at run time. Thus, checking at run-time the compliance of the requirements defined in the privacy agreement is a challenging issue. That issue must be properly addressed otherwise it could lead to agreement breaches and to lower service quality. Indeed, the private data use flow must be observed which means monitoring the behaviour of the privacy agreement. From the results of the observations, analysis can be done to come up to an understanding, why the non-compliance took place and what remedy will be provided enhancing the privacy agreement.

[*] This work is partially supported by the French National Research Agency (ANR) - Program "Jeunes chercheurs:Servicemosaic" a part of the international project ServiceMosaic; http://servicemosaic.isima.fr/.

B. Krämer, K.-J. Lin, and P. Narasimhan (Eds.): ICSOC 2007, LNCS 4749, pp. 353–364, 2007.

The common approach developed to support requirements monitoring at run-time assumes that the system must identify the set of the requirements to be monitored. In fact, as part of the privacy agreement model, the set of privacy requirements to be monitored are needed from which *monitoring private units* are extracted and their occurrences at run-time would imply the violation of the requirements. Besides the functional properties (e.g operations of the service), the time-related aspects are relevant in the setting of the privacy agreement. In addition, the non-compliance or failing to uphold the privacy requirements are manifested in terms of vulnerabilities must be identified.

In this paper, we propose an approach for the management of privacy data terms defined in the privacy agreement at run-time. The approach features a model based on state machine which is supported by *abstractions* and *artifacts* allowing the run-time management. Our contribution articulates as follows:

1. From the privacy requirements defined in the privacy agreement, we extract a set of *monitoring private units* specified by the means of Linear Temporal logic (LTL) formulas,
2. The set of privacy misuses is most likely met throughout the private data use is provided. That set is not limited and can be enriched by those promptly revealed when they occur in run-time and captured by the analysis,
3. A state machine based model is provided in order to describe the activation of each privacy agreement clauses, that is, it spells out the Private Data Use Flow (PDUF). The state machine supports abstractions and by the means of previous artifacts, the behaviour observations are expressed. It will *observe* which and when a clause is activated, or which and when a clause is violated and what types of vulnerabilities happened, or which clause is compliant and etc. Such observations lead to do reasoning to enhance the privacy agreement and enrich the knowledge on misuses.

The remainder of the paper is structured as follows. We start by presenting an overview of the privacy agreement developed in our previous work in Section 2. In Section 3, we describe the architectural support for privacy data use flow monitoring. Section 4 proposes an LTL-based approach to specify the monitoring private units and presents a set of privacy misuses. In Section 5, we present the private data use flow model. We discuss related work in Section 6 and conclude with a summary and issues for future work in Section 7.

2 Privacy Agreement Model

To make the paper self containing, in this section we recall the privacy agreement model specified in our previous work [4,5]. We proposed a framework for privacy management in Web services. A privacy policy model has been defined as an agreement supporting a lifecycle management which is an important deal of a dynamic environment that characterizes Web services based on the state machine, taking into account the flow of the data use in the agreement. Hence, WS-Agreement has been extended including privacy aspects. In this setting, the features of the framework are:

- The privacy policy and data subject preferences are defined together as one element called *Privacy-agreement*, which represents a contract between two parties, the service customer and the service provider within a validity. We provided abstractions defining the expressiveness required for the privacy model, such as rights and obligations.
- The framework supports lifecycle management of privacy agreement. We defined a set of events that may occur in the dynamic environment, and a set of change actions used to modify the privacy agreement. An *agreement-evolution* model is provided in the privacy-agreement.
- An *agreement-negotiation protocol* is provided to build flexible interactions and conversations between parties when a conflict happens due to the events occurring in the dynamic environment of the Web service.

Informally speaking the abstraction of privacy model is defined in terms of the following requirements:

- *data-right*, is a predefined action on data the data-user is authorized to do if he wishes to.
- *data-obligation*, is the expected action to be performed by service provider or third parties (data- users) after handling personal information in data-right. This type of obligation is related to the management of personal data in terms of their selection, deletion or transformation.

Formally speaking, we defined **data-right** r_d as a tuple (u, d, op_d, p_d), with $u \subseteq \mathcal{U}$ and $d \subseteq \mathcal{D}$ and $op_d \subseteq \mathcal{PO}$ and $\mathcal{R}^d = \{\{r_d^i\}_j \ / \ i > 0 \ j > 0\}$, where $\mathcal{U}$ is the ontology of data users and $\mathcal{D}$ is the ontology of personal data and $\mathcal{PU}$ is the ontology of purposes $\mathcal{PO}$ is the set of authorized operations identifying purposes of the service and p_d is the period of data retention (the data-right validity), and $\mathcal{R}^d$ is the set of data-rights.

We defined **data-obligation** o_d as a tuple (u, d, a_d, μ_d) with $u \subseteq \mathcal{U}$ and $d \subseteq \mathcal{D}$ and $a_d \in \mathcal{A}_o$ and $\mathcal{O}^d = \{\{o_d^i\}_j \ / \ i > 0 \ j > 0\}$, where $\mathcal{U}$ is the ontology of data users and $\mathcal{D}$ is the ontology of personal data and $\mathcal{A}_o$ a set of actions that must be taken by the data user and μ_d is an activated date of the obligation, and $\mathcal{O}^d$ is the set of data-obligations.

Based on those requirements, we formalized a privacy data model $\mathcal{P}^d$ as a couple $< \mathcal{R}^d, \mathcal{O}^d >$, where $\mathcal{R}^d$ is the set of data-rights and $\mathcal{O}^d$ is the set of data-obligations. By means the proposed privacy model, we extended current WS-Agreement specifications which do not support the privacy structure and do not include the possibility to update the agreement at runtime. The proposed extension is reflected in a new component in a WS-Agreement called ***Privacy-agreement***,

A privacy-agreement structure is represented in two levels :

(1) *policy level*, it specifies the *Privacy-Data term* defined as a set of *clauses* of the contract denoted by $\mathcal{C}$ between the provider and the customer. The description of the elements defined in the privacy-data model is embedded in this level, including guarantees dealing with privacy-data model.

(2) *negotiation level*, it specifies all possible events that may happen in the service behavior, thus evolving the privacy guarantee terms defined in the policy level. Negotiation terms are all possible actions to be taken if the guarantee of privacy terms is not respected, then a conflict arises. They are used through a negotiation protocol between

the service provider and the customer. We also defined in this level the validity period of the privacy agreement and a set of penalties when the requirements are not fulfilled. In the rest of the paper, we are interested by the first level. We will present a way to observe the use of the private data throughout the run time, and how to capture the compliance of the agreement related in the privacy data terms.

3 Overview of the Monitoring Framework

We devise a privacy-compliance architecture for monitoring. It incorporates three main components discussed in this paper, they are depicted in Figure1 and are namely a *private requirements specification*, a *PDUF Observer*, a *monitor*. The figure assumes the web service executes a set of operations using private data. While executing the operations of the service, the process generates events stored in a database as logs.

In order to check the privacy compliance, the monitoring private units are extracted from the *private requirements specification* defined in the privacy agreement. Monitoring private units are specified by the means of LTL formulas taking into account the privacy time-related requirements using a set of clocks.

The *monitor* collects the raw information to be monitored regarding the monitoring private units from the event logs database. The collected data and private data misuses stored in a database are fitted together in the *PDUF Observer* component in order to check the non-compliance.

The PDUF observer observes the behaviour of the private data use flow. The privacy agreement clauses are observed, which means, when a clause is activated, or which

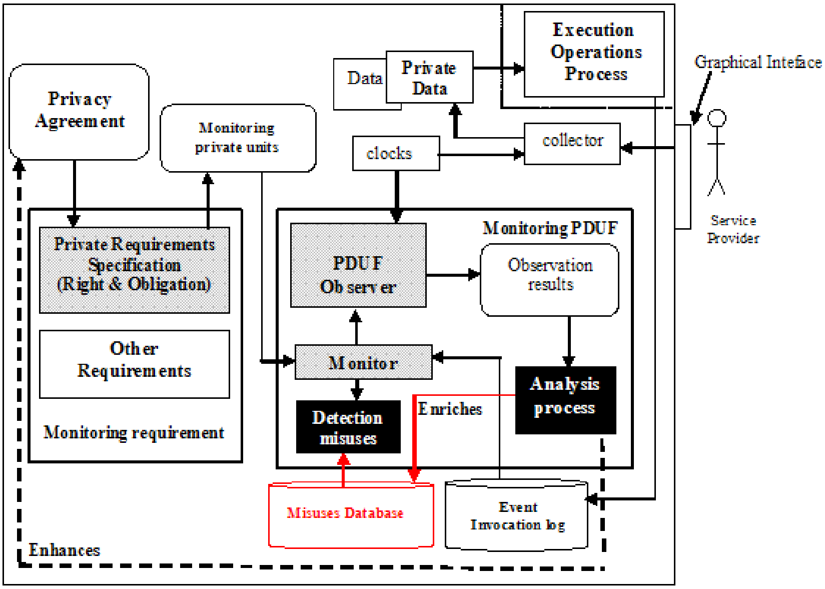

Fig. 1. Monitoring framework

and when a clause is violated and what types of vulnerabilities happened, or which clause is compliant etc. A model to represent such behaviour is provided. At the end of the observations the *observation results* report is generated to the Analysis process depicted in the figure.

From the previous observed results and reasoning facilities, the analysis process will provide diagnosis of violations, for instance understanding why the non-compliance took place and what remedy will be provided enhancing the privacy agreement. It can also enrich the database of misuses by those promptly revealed when they occur at run-time. Finally, the detection misuses component consumes the misuses recorded in the database and identify the violation types from compliant usage behaviour. We will not give more details about the analysis and detection components, they are out of the scope of the paper.

4 Requirements for Monitoring Privacy

One of the key aspects for the reliability of the service is the trustworthiness of the compliance of its collected private data use to the agreement. To ensure the privacy agreement compliance, the observation of the service behaviour and its private data use becomes a necessity. For making the compliance happen, keep track of all uses is a fact, that is, from the result of the observations, if needed when violations are detected, the revision of the agreement can be held and relaxed. Indeed, to make the observation effective, two essential ingredients are required, we need to define what kind of knowledge must be monitored and the knowledge which makes the agreement not compliant. In this section we discuss the two aspects.

4.1 Monitoring Units for Privacy

We distinguish four types of unit to be monitored: *private data unit*, *operation unit*, *temporal unit* and *role unit*.

• *Private data unit.* The private data unit d is the core of our monitoring framework. In fact, from the log, we need to observe only the private data and its behaviour.

• *Operation unit.* We distinguish two types of actions (i) actions used to complete the service activity for the current purpose for which it was provided and are denoted by $Op_{current}$ (ii) actions used by a service to achieve other activities than those for which they are provided, called $Op_{extra-activity}$. Those two kinds of operations are proposed in order to know when a compliance is compromised, while the service is running for which it was provided or for some operations else. The set of the operations is denoted Op.

• *Role unit.*We need to observe who will use the private data.

• *Temporal unit.* The analysis of time-related aspects of the privacy monitoring requires the specification of operation durations and timed requirements. The instance monitor i.e. temporal unit is defined as a temporal formula using Linear Temporal Logic (P,S,H, operators) [10]. We identify four types of temporal units, and we denote the set of temporal units by $\mathcal{T}$:

Definition 1. ***(Right triggering time).*** *For each collected private data d, the right triggering time denoted* ϵ *is the activation time of the operation associated to the right:* $\forall R^i_d \in \mathcal{C} \rightarrow \exists \epsilon^i_d \in T \mid (op^i_d.R^i_d)^{\epsilon^i_d}$ *is activated, where i is the i th right associated to the private data d,* $\mathcal{C}$ *is a set of clauses in the agreement, and T is a domain of time values, and the LTL formula using P the past temporal operator is* $\models_{\epsilon^i_d} P\ op^i_d.R^i_d$*, which means in the past at* ϵ^i_d *time the operation is true.*

Definition 2. ***(Right end time).*** *For each collected private data d, the right end time denoted* β *is the end time of the data use (operation) associated to the right:* $\forall R^i_d \in \mathcal{C} \rightarrow \exists \beta^i_d \in T \mid (op^i_d.R^i_d)^{\beta^i_d}$ *is finished, and the LTL formula is* $\models_{\beta^i_d} P \neg op^i_d.R^i_d$ *at* β *time the operation is not valid.*

Definition 3. ***(Obligation triggering time).*** *For each collected private data d, the obligation triggering time denoted* μ *is the activation time of the action associated to the obligation:* $\forall O_d \in \mathcal{C} \rightarrow \exists \mu_d \in T \mid (a_d.O_d)^{\mu_d}$ *is activated, the LTL formula using S since operator,* $\models_{\mu_d} (a_d.O_d) S (\neg op_d.R_d)$*, which means* $a_d.O_d$ *is true since* $\neg op_d.R_d$ *(The formula is valid for the last occurrence of each right in which the obligation is associated to).*

Definition 4. ***(Obligation end time).*** *For each collected private data d, the obligation end time denoted* α *is the end time of the action associated to the obligation:* $\forall O_d \in \mathcal{C} \rightarrow \exists \alpha_d \in T \mid (a_d.O_d)^{\alpha_d}$ *is ended, the LTL formula is* $\models_{\alpha_d} P \neg a_d.O_d$ *at* α *time the action is not valid.*

4.2 Privacy Misuses

In this section, we identify the non-compliance or failing to uphold the agreement manifested in terms of vulnerabilities or misuses. We provide a privacy misuses which is most likely met throughout the private data use. We have classified them into two classes *explicit* and *implicit misuses*. The former one can be visualized in our private data use flow model whereas the latter can not be identified. For instance, *security on data*, *accountability* can not be identified in our model, so it is not in the scope of the paper. We classified three types of explicit misuses, *Temporal misuses*, *operation misuses* and *role misuses*. Table 1 summarizes such misuses. However, the listed misuses are not unique, while run-time, some new misuses can be detected and come to enrich the misuse database. How to detect such misuses is not discussed in this paper.

5 Monitoring Private Data Use Flow

In order to describe the lifecycle management privacy data terms defined in the agreement, we need to *observe* the data use flow. Such observations will allow us to make analysis, diagnosis and to provide reasoning on violations, for instance why the violations happen, what we can improve in the agreement for making the compliance of the agreement happens etc. The analysis aspect is not handled in this paper.

We propose to express the Private Data Use Flow (PDUF) as a state machine because of its formal semantic, well suited to describe the activation of different clauses of the

Table 1. Misuses identification through privacy data use flow

Requirement	Compliance Category	Misuses	Type of misuses
Data-right	Use	no-authorized operation op_d *[wrong-use]*; the misuse happens when the following formula is not valid: $\not\models Hop_d.R_d$, in all the past op_d is not admitted.	Explicit
	Retention time	violation of data retention period: the misuse happens when the formula $\models P\ ((\beta_d - \epsilon_d) > p_d)$ is valid.	Explicit
	Disclose-To	a *[wrong collector]* as third party; the following formula is not valid: $\not\models Hu_d.R_d$, in all the past u_d is a wrong user.	Explicit
Data-obligation	Obligation Activation date	violation of the obligation activation, the misuse happens when the formula $\models P(\beta_d > \mu_d)$ is valid.	Explicit
	Security on data (delete, update, hide, unhide,...)	Lack or failure of mechanism or procedure.	Implicit
Security	/	1)loss of confidentiality and integrity of data for flows from the Internet, 2) external attacks on the processes and platform operating systems since they are linked to the Internet, 3) external attacks on the database,...	Implicit

privacy agreement. It is an effective way to identify privacy vulnerabilities, where a service 's compliance to privacy regulations may be compromised. It will show *which* and *when* a clause is activated toward the monitoring or which and when a clause is violated. The time-related requirement properties set in the agreement are depicted explicitly in the state machine. It will specify the states of each activated clause in the policy level. The semantic of the state machine is to define all the triggered operations involving private data from the activation of the agreement (initial state) to the end of the agreement (final state) . We need to keep track of all private data use with or without violations. Figure 2 shows an example of the privacy data term activation for the purchase service provider.

We have identified several abstractions in relation to private data flow, *private data use* abstractions and *authorization* abstractions. The first abstractions describe the different states in which the agreement is -which private data is collected and when it is used and for what and who use it- . The authorization abstractions provide the conditions that must be met for transitions to be fired.

In this formalism, the fact that the private data has a time retention for a right (respectively the activation time of an obligation) called *fixed guard time*, the private data use time is represented by time increment in the state, followed by the end of the right (respectively obligations) with success or a violation of that time. Intuitively, PDUF is a

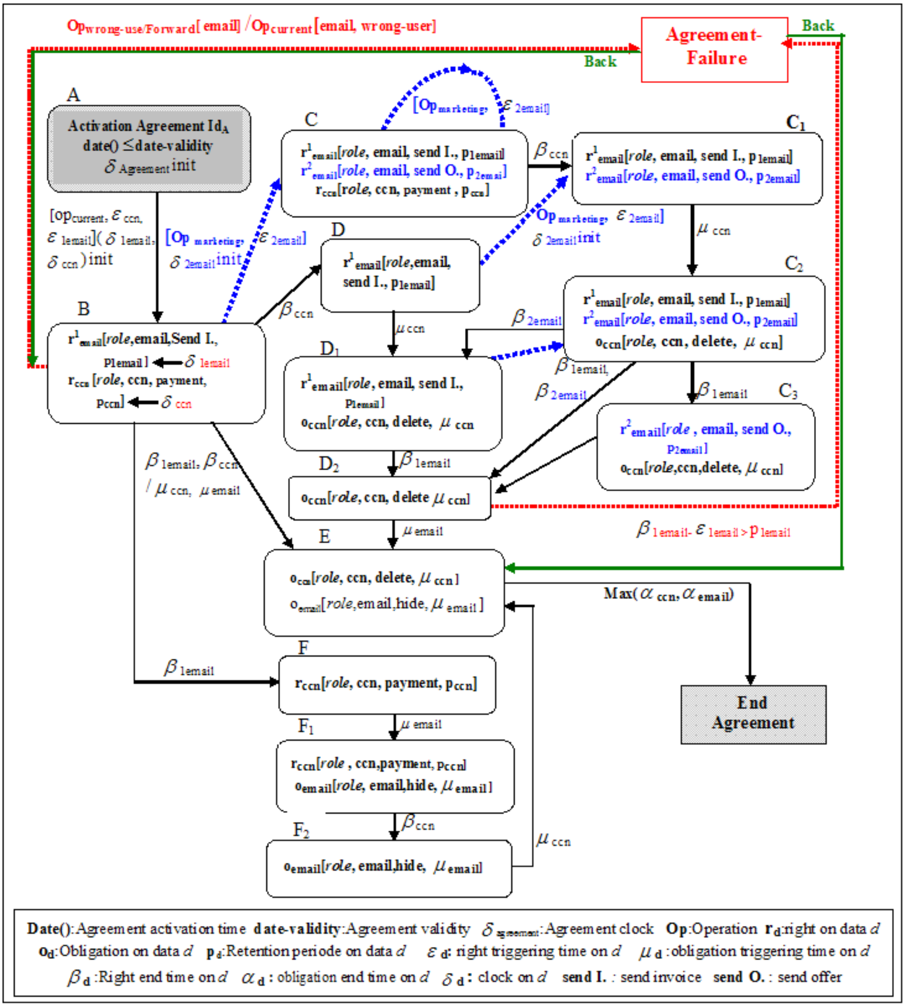
Opwrong-use/Forward[email] /Opcurrent[email, wrong-user]
Agreement-Failure
Back
Back
A
Activation Agreement IdA
date() ≤ date-validity
δ Agreement init
[opcurrent, ε ccn, ε 1email](δ 1email, δ ccn) init
[Op marketing, ε 2email]
δ 2email init
B
r1email[role, email, Send I., P1email] ← δ 1email
rccn [role, ccn, payment, Pccn] ← δ ccn
C
[Op marketing, ε 2email]
r1email[role, email, send I., p1email]
r2email[role, email, send O., p2email]
rccn[role, ccn, payment , pccn]
β ccn
C1
r1email[role, email, send I., p1email]
r2email[role, email, send O., p2email]
Op marketing, ε 2email]
δ 2email init
μ ccn
D
r1email[role, email, send I., p1email]
β ccn
μ ccn
C2
r1email[role, email, send I., p1email]
r2email[role, email, send O., p2email]
occn[role, ccn, delete, μ ccn]
β 2email
β 1email,
β 2email
β 1email
C3
r2email[role , email, send O., P2email]
occn[role,ccn,delete, μ ccn]
D1
r1email[role, email, send I., P1email]
occn[role, ccn, delete, μ ccn
D2
β 1email
occn[role, ccn, delete μ ccn]
β 1email, β ccn / μ ccn, μ email
μ email
β 1email- ε 1email > p 1email
E
occn[role, ccn, delete, μ ccn]
oemail[role, email, hide, μ email]
Max(α ccn, α email)
End Agreement
F
β 1email
rccn[role, ccn, payment, pccn]
F1
μ email
rccn[role , ccn, payment, pccn]
oemail[role, email, hide, μ email]
F2
β ccn
μ ccn
oemail[role, email, hide, μ email]
Date():Agreement activation time date-validity:Agreement validity δ agreement:Agreement clock Op:Operation rd:right on data d
od:Obligation on data d pd:Retention periode on data d ε d: right triggering time on d μ d :obligation triggering time on d
β d :Right end time on d α d : obligation end time on d δ d : clock on d send I. : send invoice send O. : send offer

Fig. 2. Private Data Use Flow (PDUF)

finite state machine for which a set of clock variables is assigned denoted by Δ. A variable is assigned for each activation of the clauses (rights and obligations). The values of these variables increase with passing the time. The transition will take place when an operation is activated or monitoring time units are triggered. If the temporal units are compliant to the guard times, it will happen the transition will take place with success and no violation is recorded in that state. However, if non-compliance is detected, the transition will take place with violation, then the state is marked as violated.

Definition 5. *(PDUF.) A PDUF is a tuple* $(\mathcal{S}, s_0, s_f, \mathcal{M}, \mathcal{R}, \mathcal{Q})$

- $\mathcal{S}$ *is a set of states;*
- $s_0 \in \mathcal{S}$ *is the initial state, and* $s_f \in \mathcal{S}$ *is the final state ;*

- *$\mathcal{M}$ is a set of monitoring private units: set of triggered operations and/or set of temporal units, $\mathcal{M} = \{OP, \mathcal{T}\}$;*
- *$\mathcal{R} \subseteq \mathcal{S}^2 \times \mathcal{M} \times 2^{\Delta}$ is a set of transitions with a set of operations or a set of triggering time and a set of clocks to be initialized $\delta_{d-init} \in \Delta$;*
- *$\mathcal{Q} : \mathcal{S} \rightarrow \{\delta_i \mid \delta_i \in \Delta, i \geq 1\}$ assigns a set of clocks to the states.*

The effect of each transition $\mathcal{R}(s, s^{'}, m, c)$ from the source state s to the target state $s^{'}$is to set a status of the clauses in the agreement which means to perform an operation $op \in \mathcal{OP}$ using a private data or a monitoring time unit $t \in \mathcal{T}$ is activated.

Let's define the semantic of PDUF through the following example for the agreement with a set of clauses (rights and obligations).

Example 1. Let us consider the example of a purchase service without giving details about transactions between the customer and the service. An agreement has been signed between them setting up a set of clauses with a validity period denoted by *validity-date*. Those clauses are specified as follows: at the date *date()* the agreement is activated and the service collects email address (email) and Credit card number (ccn). Those private data are used for two types of operations **(1)** to complete the service activity for the current purpose i.e. the email is used to send invoices and Credit card number for the payment of invoices. The operation are expressed by the following rights $r^1_{email}(role, email, send\ \ invoice, p_{1email})$ and $r_{ccn}(role, ccn, payment\ \ invoice, p_{ccn})$ **(2)** to achieve other activities than those for which they are provided, for instance marketing purpose i.e. the email is used to send the available products and their prices, that clause is expressed by the right $r^2_{email}(role, email, send\, offer, p_{2email})$.

When the retention times of the private data email and ccn $(\beta_{1email}, \beta_{2email}, \beta_{ccn})$ are elapsed, the corresponding obligations are triggered, $O_{email}(role, email, hide, \mu_{email})$ and $O_{ccn}(role, ccn, delete, \mu_{ccn})$. Those obligations specifying the role must hide (respectively delete) as soon as the activation date μ_{email} (respectively μ_{ccn}) is reached.

In what follow, due to the space limitation we will not comment on all the state machine, and for the sake of clarity, we omit some details about it, such as the clocks on the states and all the misuses etc.

States: we define four types of states:

- The initial state s_i represents the activation of the agreement where the first private data of the customer is collected. In Figure 2, s_i is defined by A.
- The intermediary states represent the flow of the collected private data use. By entering a new state, a private data is used.
  - to complete the activity of the service for which it was provided, identified in Figure 2 by $Op_{current}$. In the state B, the current operations are *SendInvoice* and *payment*. In this state, the clocks δ_{1email} and δ_{ccn} are activated respectively to r^1_{email} and r_{ccn} and incremented passing the time.
  - and/or to achieve an extra activity as depicted in Figure 2 by $Op_{marketing}$. The right r^2_{email} is activated in the state C as soon as the marketing operation is triggered. The same operation can be activated as many times as the data time retention p_{email} is valid. It is represented by a *loop* in the state C. The privacy agreement remains in the same state.

- and the data use is finished (the right). For instance, the agreement will be in the state C_1 since the data retention guard time is reached, which means the finishing time of the right is over and is denoted by β_{ccn}.
- and/or to activate an operation dealing with the security (e.g. obligations) when the retention time of the private data defined as a fixed time in the right is elapsed and the time for triggering the obligations starts. For instance, such case is depicted in Figure 2 in the state C_2, where o_{ccn} is activated when the usage time of the date β_{ccn} is reached and the obligation time starts defined in the transition by μ_{ccn}.

- The *virtual* state labeled *Failure agreement* will be reached when a private data is used to achieve the operation misuse, and/or role misuse and/or time misuse happens regarding the clock variable values and fixed times. For instance, the first type of misuse is identified by $Op_{wrong-use/Forward}[email]$ between state B and Failure agreement state. We call this state as a virtual state because it is considered only like a flag of misuses.
- The final state s_f represents the end of the agreement which means the validity of the agreement is over, and either the data use in all its shapes is compliant to the agreement or the agreement is not respected due to the misuses. The best case is to reach the end of the agreement without any misuses as depicted in the figure from the state E to the end-agreement state.

Transitions: Transitions are labeled with conditions which must be met for the transition to be triggered. We have identified three kinds of authorization abstractions:

- Activation conditions. We define two types of activation (i) an operation has the authorization to collect private data to achieve the current aim of the service, for instance, $op_{current}$ condition on the transition from the state A to the state B, an operation dealing with an extra activity of the service has the authorization to be triggered. For instance, the operation $op_{marketing}$ from the state B to the state C.
- Temporal conditions. The transition is called *timed transition*. Regarding the temporal monitoring unit, we define four types of timed transitions (1) *right triggering time* ϵ, for instance from the state B to the state C the timed transition is labeled by ϵ_{2email} along with the activation of the clock δ_{2email} assigned to the right r^2_{email} (2) *Right end time* β, from the state C to state C_1 the transition is labeled β_{ccn}, which means the ccn use is over (3) *Obligation triggering time* μ, the authorization to keep the private data is finished and the obligation is triggered, for instance from the state C_1 to C_2, the transition is labeled μ_{ccn}, the operation of security must be fired (4) *Obligation end time* α, the obligation is over, for instance from the state E to the end-agreement state, we calculate the maximum of the two end times α_{email} and α_{ccn}, in our case it is the best way to finish the compliance of the agreement.
- Misuse Conditions. The transition can be labeled by all the misuses identified in section 4.2. For the misuse dealing with the operations , the target state of the transition is *failure-agreement* and *Back* to the previous state, for instance, the operation $op_{wrong-use/forward}$ on the transition between the state B and the failure-agreement state, and back to the state B. For the temporal misuse the target state of the transition is *failure-agreement* and no back to the previous state rather to the

next state, for instance, a time violation happens in D_2 and the system passes to the next state E.

6 Related Work

The literature is very scarce on works dealing with monitoring the privacy compliance in web service. However, the problem of web services and distributed business processes monitoring is investigated in the works [2,3,9,12,1,7]. The research in [9,12] is focusing on monitoring of service-based software (SBS) systems specified in BPEL. They use event calculus for specifying the requirements that must be monitored. The run-time checking is done by an algorithm based on integrity constraint checking in temporal deductive databases. Barezi et al in [2,3], developed a tool that instruments the composition process of an SBS system in order to make it call external monitoring services that check assertions at runtime. The work in [1] is close to the previous works, the authors present a novel approach to web services described as BPEL processes. The approach offers a clear separation of the service business logic from the monitoring functionality. Moreover, it provides the ability to monitor both the behaviours of single instances of BPEL processes, as well as behaviours of a class of instances. Lazovik et al. [8] propose an approach based on operational assertions and actor assertions. They are used to express properties that must be true in one state before passing to the next, to express an invariant property that must be held throughout all the execution states, and to express properties on the evolution of process variables. While providing facilities for the verification of processes these approaches do not take privacy requirements into account.

In terms of privacy compliance, there exist few works including [6,13,11]. In [6], the authors examine privacy legislation to derive requirements for privacy policy compliance systems. They propose an architecture for a privacy policy compliance system that satisfies the requirements and discuss the strengths and weaknesses of their proposed architecture. In [13] the author proposes a graphical visualization notation to facilitate the identification of private information vulnerabilities that can lead to privacy legislation non-compliance. In [11], the authors automate the management and enforcement of privacy policies (including privacy obligations) and the process of checking that such policies and legislation are indeed complied with. This work is related to enterprise. While providing tools for privacy compliance in the previous works, however, these approaches do not take private data use flow into account and no formal method along with reasoning and also no time-related properties are discussed.

7 Conclusion

This work has proposed an effective and formal approach to observe and verify the privacy compliance of web services at run-time. We have emphasized private data use flow monitoring of privacy-agreement requirements, which is an important issue to date has not been addressed. It is a state machine based approach, that allows to take into account the timed-related properties of privacy requirements and to facilitate the identification of private information misuses. The privacy properties to be monitored are specified in LTL. The monitored units are extracted from the privacy agreement requirements. The approach supports the monitoring of a set of identified misuses that lead to non-compliance, and which can be enriched from the observation diagnosis. The approach is still under development. Our ongoing work and a promising area for the future include: (1) The development of reasoning facilities to provide a diagnosis of misuses, (2) The development of tools for detecting the misuses (3) The development of tools along with metrics for enhancing the privacy-agreement from the observations (4) Expanding the approach to handle the composition of the services.

References

1. Barbon, F., Traverso, P., Pistore, M., Trainotti, M.: Run-time monitoring of instances and classes of web service compositions. In: ICWS'06. Proceedings of the IEEE International Conference on Web Services, pp. 63–71. IEEE Computer Society Press, Chicago (2006)
2. Baresi, L., Ghezzi, C., Guinea, S.: Smart monitors for composed services. In: ICSOC '04. Proceedings of the 2nd international conference on Service oriented computing (2004)
3. Baresi, L., Guinea, S.: Towards dynamic monitoring of ws-bpel processes. In: Benatallah, B., Casati, F., Traverso, P. (eds.) ICSOC 2005. LNCS, vol. 3826, pp. 269–282. Springer, Heidelberg (2005)
4. Benbernou, S., Meziane, H., Li, Y.H., Hacid, M.: A privacy agreement model for web services. In: SCC'07. IEEE International Conference on Service Computing, IEEE Computer Society Press, Salt Lake City, USA (2007)
5. Guermouche, N., Benbernou, S., Coquery, C.E, Hacid, M.: Privacy-aware web service protocol replaceability. In: ICWS'07. IEEE International Conference on Web Services, IEEE Computer Society Press, Salt Lake City, USA (2007)
6. Yee, G., Korba, L.: Privacy policy compliance for web services. In: ICWS'04. Proc. of the IEEE International Conference on Web Services, IEEE Computer Society Press, San Diego, USA (2004)
7. Kazhamiakin, R., Pandya, P., Pistore, M.: Representation, verification, and computation of timed properties in web. In: ICWS'06. Proceedings of the IEEE International Conference on Web Services, IEEE Computer Society Press, Los Alamitos (2006)
8. Lazovik, A., Aiello, M., Papazoglou, M.: Associating assertions with business processes and monitoring their execution. In: ICSOC '04. Proceedings of the 2nd international conference on Service oriented computing (2004)
9. Mahbub, K., Spanoudakis, G.: Run-time monitoring of requirements for systems composed of web-services: Initial implementation and evaluation experience. In: ICWS. 2005 IEEE International Conference on Web Services, IEEE Computer Society Press, Orlando, Florida, USA (2005)
10. Manna, Z., Pnueli, A.: The Temporal Logic of Reactive and Concurrent Systems:Specification. Springer, Heidelberg (1992)
11. Mont, M.C., Pearson, S., Thyne, R.: A systematic approach to privacy enforcement and policy compliance checking in enterprises. In: Fischer-Hübner, S., Furnell, S., Lambrinoudakis, C. (eds.) TrustBus 2006. LNCS, vol. 4083, pp. 91–102. Springer, Heidelberg (2006)
12. Spanoudakis, G., Mahbub, K.: Non intrusive monitoring of service based systems. International Journal of Cooperative Information Systems (2006)
13. Yee, G.: Visualization for privacy compliance. In: VizSEC '06. Proceedings of the 3rd international workshop on Visualization for computer security, Fairfax, USA (2006)