Conceptual Modeling of Privacy-Aware Web Service Protocols

Rachid Hamadi, Hye-Young Paik, and Boualem Benatallah

School of Computer Science and Engineering
The University of New South Wales, Sydney NSW 2052, Australia
{rhamadi,hpaik,boualem}@cse.unsw.edu.au

Abstract. Internet users are becoming increasingly concerned about their personal information being collected and used by Web service providers. They want to ensure that it is stored and used according to the providers' privacy policies. Since these policies are mainly developed and maintained separately from the business process that collects and manipulates data, it is hard to perform analysis and management of the processes in terms of privacy policies. To address this problem, we propose a formal technique with which Web service providers describe the use and storage of personal data. The description is integrated with a Web service protocol using an extended state machine model. Having such a conceptual model will enable model-driven development and management of Web service protocols with respect to their privacy aspects such as collection, disclosure, and obligation. A tool support has been implemented, as part of `ServiceMosaic`, to let designers model privacy aspects within the Web service protocol.

Key words: Web service protocols, privacy policies, conceptual modeling.

1 Introduction

Internet users are becoming increasingly concerned about their personal information being collected by online businesses and where and how this information might be used. The fact that modern business applications are extremely complex and often involve interactions with many other autonomous and heterogeneous partner systems complicates the task of preserving privacy even more.

In this paper, we focus on privacy aspects in Web services. Web services are emerging as a promising technology for the effective *automation of inter-organizational interactions*. Several Web services related standards such as WSDL [1], UDDI [1], and BPEL4WS [2] are already developed and used in the real-world business environments. For instance, Amazon.com's e-catalog system integrates hundreds of other independent second-hand booksellers. The booksellers' systems are loosely coupled with Amazon.com's system as *Web services* through Amazon.com's Web services APIs.

Despite its growing popularity, the development of technologies addressing privacy issues in Web services has not kept the same pace. For example, the customers of Amazon.com search, browse and buy items using a unified user interface provided by the company, however, the actual processing of an order, delivery, analysis of sales data, personalised services, etc. may involve passing the customer's personal data to third parties (e.g., the individual second-hand booksellers). Furthermore, the third parties may have their own set of privacy policies that are different from Amazon.com's or different from each other. In the midst of these business operations, it is not clear where and how the statements in the privacy policies apply to the activities and whether they will be enforced or not.

One of the problems is that there is no proper modelling technique for capturing the privacy aspects for a Web service. That is, no current Web service modelling technologies offer a simple way to state a privacy requirement (e.g., "The intended recipient of this message is a delivery service and the data should be removed after the delivery is completed") in a Web service model.

So far, online companies have dealt with privacy issues largely by publishing privacy policies. Privacy policies describe the organisation's general business practices based on the criteria set by government rules and regulations. However, privacy policies do not discuss the behaviour of individual business applications within the organisation that actually collect/analyse and distribute personal data. This makes the enforcement of the policies difficult. We argue that a model-driven approach, where privacy policies are modelled explicitly as part of the Web service behaviour, can contribute to making the privacy policies explicit and enforceable. Having such a conceptual model will enable model-driven development and management of Web service protocols with respect to their privacy policies.

In this paper, we propose a Web service modeling technique purposely designed to capture privacy abstractions while describing the behaviour of a Web service. The basis of the proposed idea is from our previous work published in [3, 4] in that we used the same model to represent the way Web services interact with others (i.e., Web service protocols). Our contributions are as follows:

- We identify common privacy abstractions in Web service protocols. The identification is based on the study of many privacy policies publicly available on Web portals.
- We propose an extended state machine as a conceptual model that incorporates the privacy abstractions into a Web service protocol model. In the model, we introduce the concept of *states with multiple privacy properties*. We reflect the consequence of a transition (e.g., execution of a Web service operation) as privacy properties such as access, disclosure and retention in a state. We also consider the removal of the retained data as a *timed transition*.

This paper is organised as follows. Section 2 introduces privacy and the related terminology. Section 3 discusses the privacy policies in Web services and gives some observations. Section 4 introduces the proposed conceptual model. Section 5 describes the tool supporting the model proposed as well as the application of privacy-aware Web service protocols. Section 6 reviews some related work. Finally, Section 7 concludes the paper.

2 Overview of Privacy Policies

Before we discuss the modelling of privacy in Web services, we first introduce privacy policies in general to familiarize the readers with the main issues and terminology in the topic. We also present the current standard languages for encoding privacy policy specifications.

2.1 Privacy Policies

What kind of privacy aspects are addressed or declared in a privacy policy may be different depending on the rules and regulations. However, studying many privacy policies of the online companies that are publicly available on the Web tells us that there are some standard elements that commonly appear in all privacy policies. We summarise the gist of privacy policies as follows:

Common Elements. The key elements of almost all privacy policies can be categorised as follows:

Personal Data: The first thing we notice in policy statements is the identification of personal data collected by a Web site. The statements may differentiate information collected via an explicit means (e.g., user account registration, payment account information, member profile or preference setting, etc.) and an automatic means such as IP addresses, Web server logs and cookies. It may also declare information collected from other sources. For example, Amazon.com's policy states that the company may obtain credit history information of a customer from credit bureaus.

Purposes: Once the data are identified, the statements will also declare the purpose of using the data. The statement about purposes may appear in a separate section and does not refer to the specific data. Hence, we would only see generic purposes such as "to fulfill your request" or "to customise advertising"[1], etc.

[1] Quoted from Yahoo's Web site

Other times, the statement may be more specific and refers to particular types of data used for the purpose (e.g., "If you are registered for our Internet Banking, we may use your email address to advise you of any upgrades or changes to these services."[2])

Recipient: In many examples we have seen, the policies do not explicitly state the intended recipient of each type of data collected. However, it is reasonable to assume that unless stated otherwise, the intended recipient of the data is the organisation itself.

Disclosure: If the recipient of the data is not the organisation, or the data can be shared with business partners (e.g., shipment service), we would see a statement that declares it. For example, Google's policy states that only aggregated, non-personal information is shared with third parties. You can also find a statement that says "If Google becomes involved in a merger, acquisition, we will provide notice before personal information is transferred.".

Extra Elements. Besides the common features, many policies include more information about privacy. We list some of them here:

Data retention: Some policies might state explicitly how long the collected data are retained by the organisation. More than often, data may be retained indefinitely, unless stated otherwise.

Access to data: Some of the retained personal data can be accessed by the owner for correction or update purposes. Many privacy policies include information as to what kind of data the users will have access to. Many organisations that require users to register before receiving any services will often provide a facility for the user to view/update some part of the personal data. It is noted that, as explained, the term "Access to data" refers to the right of the data owner to access the his/her personal data after it is collected. It does not refer to access by third parties.

Opt in/out: A company may provide services that users can choose to receive or not to receive. A typical example of this is subscription to a mailing list to which product updates are posted. This feature of privacy policy is often referred to by the research community as *User consent management*.

2.2 Privacy Policy Specification Standards

Although majority of the publicly accessible privacy policies are written in plain English, there are standard languages designed for encoding the various aspects of privacy policies:

Platform for Privacy Preferences (P3P). P3P [5] is mainly designed for Web site operators to declare their intended use of the data they collect about the user. Having this standard allows the policies to be effectively retrieved and analysed using existing query techniques. Together with P3P user agent, for example, the user can block some of the Web content that does not match his/her privacy practice preferences. P3P was developed by the World Wide Web Consortium (W3C) and is officially recommended in 2002. See [5] for a complete specification of P3P.

Enterprise Privacy Authorization Language (EPAL). EPAL [6] is designed for specifying enterprise privacy policies, focusing on access authorisation to personal data. EPAL policies are expressed as "rules" which can be enforced through an implementation of the rule engine. Also, the rule execution logs from such engine can be used for auditing to demonstrate the organisation's compliancy with privacy legislation or recommended practices. At the time of writing this paper, the language is still in the submission status in W3C. See [6] for a complete specification of EPAL.

Our work does not make any assumption about the choice of the language that the policies are written. They could be in plain English or in one of the standard languages. What we would like to focus on is to encode such policies into Web service modelling process. If the policies are written in a standard language (e.g., P3P), of course, some of the encoding process can be done automatically by "reading" in the policies into the tool we provide.

[2] Quoted from Westpac Banking Co. Web site

3 Web Services and Privacy Policies

Privacy aspects in real world scenarios. In search for real world examples of Web services and their privacy policies, we have studied customer-application interactions in many commercial organisations that offer their services online (e.g., Web portals, retails, mortgage brokers, banking services, etc.). An ideal situation would have been to study existing Web services and understand their characteristics and requirements in relation to privacy. However, the Web services are still in its infancy. Only few Web services are available on the Internet, and they typically provide very simple functionalities (such as stock quote, weather service, etc.), without any multiple interactions with the client. There are indeed a few contexts in which Web services are available and are used for sophisticated business collaboration, but this mostly happens within a closed community of business partners, which means these services are not publicly available[3].

We looked for other possible sources. Well-known B2C Web sites such as Amazon.com or Travelocity.com seemed reasonable for many reasons. There are many such Web sites successfully operating for many years and they include privacy policy documents that describe what kind of data are collected for what purposes, etc. Of course, these Web portals are designed to interact with humans, while Web services are oriented to applications. Nonetheless, we believe that by analyzing the portal Web sites it is possible to extrapolate what would be the behavior of an "equivalent" Web service. For example, in our case, by analyzing the privacy policy statements of a Web site (such as `Amazon.com`) and by understanding the operations it makes available to the browsing user, we can make an educated guess about the behaviour of the Web service if the site were implemented as one.

A timed Web service protocol model. The model we use to describe the behaviour of a Web service is based on [3, 4]. The model uses the concept "a Web service protocol". A Web service protocol in a Web service is understood as the set of acceptable message exchanges and the order in which they should occur when interacting with the service.

The model is based on the traditional state machine formalism, since it is simple, well known, and suited to describe reactive behaviors. According to the model, a Web service protocol which a Web service supports is described by a set of states and transitions. States are labeled with a logical name, such as `Logged` or `Ordered`. Transitions are labeled by either input or output messages (or service operations), i.e., messages sent by the requester (input) or by the provider (output). We use the notation `m(+)` (respectively, `m(-)`) to denote an *input* (respectively, an *output*) message. When a service operation `op` is invoked within a business protocol that is in state `src`, then:

- if `src` has an output transition `tr` labeled with operation `op`, then the protocol moves into the destination state of `tr`,
- if `src` has no output transition labeled with operation `op`, then the protocol remains in state `src`.

In our previous work on protocol modeling [3, 4], we found that, although most state transitions occur due to explicit operation invocations, there are cases in which transitions occur without an explicit invocation by requesters. We refer to these transitions as *implicit transitions.* The majority of implicit transitions are due to timing issues (i.e., deadline expirations). For instance, many services allow requesters to perform certain operations (such as cancel purchase) only within a time window, after which these operations can not be invoked.

To characterise observation, we use timed transitions. Consider an execution of a service `S` that supports a Web service protocol `P`. If `m` is an input message in `P`, we use the expression `(m(+),t)` to denote the reception of the message `m` at an instant `t`. `t` represents the elapsed time since the *beginning* of the execution `S`. We will use the term *timed business protocol* (or *protocol* for short) to denote a Web service protocol whose definition contains timed transitions. We also use the term *protocol schema* to denote the specification of a protocol, while the term *protocol instance* will refer to an individual execution of a protocol between a service provider and a service requester.

[3] This was the same dilemma faced by [3] and we chose a similar solution.

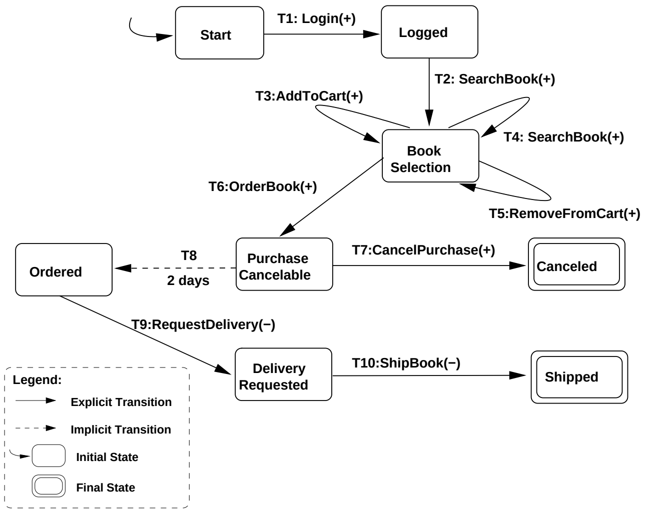

Fig. 1. The behaviour of `Snowy.com` as a Web service protocol

3.1 A Running Example: Snowy.com

Based on the study of the portal sites and their privacy policies, we have constructed a running example which is largely based on the behaviour of the Amazon.com Web site and its privacy policy [7]. We have simplified and modified the actual behaviour and policy to make it easier to illustrate our approach throughout the paper. We named the fictitious book-selling company `Snowy.com`.

The Scenario. Figure 1 presents an example of a protocol schema. According to the model, once the user is logged in, s/he can search books, add/remove books to/from a shopping cart and order the books. Once the user has ordered the bookes, s/he will have two days to cancel the order. Otherwise, the ordered books are shipped, which completes the scenario. It is noted that `T8` is a timed (i.e., implicit) transition.

Snowy.com Privacy Policies. Let us assume that Snowy.com has the following key elements in its privacy policies (For ease of understanding, we use plain English to specify the policies instead of P3P or other language):

> ***We collect*** the data that are given by you. As a result of actions such as ordering a book, you supply us with name, address, and phone numbers and credit card number. Some data may be ***collected automatically*** when you are connected to our site. We may collect login id, password, your order history, products you added or removed from your shopping cart.
> ***We share*** your name, address and phone numbers with our delivery service partners. This will ensure us that we can update your delivery status in a prompt manner.
> ***We use*** the information that you provide for purposes such as responding to your requests, customizing future shopping for you, improving our stores, and communicating with you. Especially, information about the order and shopping cart history can help us provide you with a personalised shopping experience. ***We also use*** the order history data in our market analysis.
> ***We retain the collected data for the following periods:*** the shopping cart history data for 6 months; login data for 2 months; and order detail for 3 months. However, it is assured that in case you cancel your order, we delete the order detail from your order history immediately. As long as the data are retained with us, you will be able to ***access the following data through our user management system:*** login id, password, payment information and order history.

Applying the Policies to the Scenario. Although studies suggest that people feel more comfortable with the Web sites that have privacy policies, only a small number of people actually read them [8]. Even if one fully understands the policies, it will not be easy to be on alert for every step of the way while s/he interacts with a Web site.

According to the policies above, when a customer visits the Web site, quite a lot of data about him/her is collected through automatic means. For example, let us consider when and how the policies should apply to Joe (the customer) when he interacts with Snowy.com. We will examine some of the transitions and see whether any part of the privacy policies are relevant to them.

The first transition is via `T1:Login` which carries a message that is likely to contain login id and password of Joe. Naturally, this is the first point that a reference to the policies is made. A statement in the policies says login id and password are collected. It is reasonable to think that the purpose of collecting the data is to process Joe's login request. Once Joe is logged in, he may try to search some books (`T2:SearchBook`). The message should contain the search terms. We do not see any claim in the policy about collecting search terms, so transition `T2` seems to bear no privacy concern.

However, data associated with activities such as adding/removing an item to/from the shopping cart is collected. This means the data involved in `T3` and `T4` will be retained by the company and, according to the policy, it will be used for "possibly" purposes such as *responding to your requests, customizing future shopping for you, improving our stores, and communicating with you.*

The statement also claims that in the case of cancellation, the order details will be deleted from Joe's order history. That is, an obligation for the company is to make sure that the data are not retained after cancellation. If Joe places any order and receives the goods, he should be able to access the details of the order through the user management system, as the policy states. Also, the collected data is retained for three months.

3.2 Discussions and Observations

In this section, we discuss the main lessons learned during the analysis of privacy policies and Web portals.

Privacy statements need to have clear semantics. Although policies include the standard elements that are required by the rules and regulations, it is often difficult to extract information about a particular action (i.e., a transition in a protocol schema) due to informality of the language used.

For example, in the above scenario, data used in the operations `T3/T4` will be retained, but which purpose (out of all mentioned) should apply to the data is not clear in the statement. We point out that this is quite common in many policies we have seen. It is not always possible to map a purpose to specific data.

What is also not clear is the identification of data being collected. In the protocol, such data are buried inside the messages in the operations. A message may consists of many parts and it is not easy to identify which part of a message is related to the personal data referenced in the policies.

For the policies to be enforceable, we need to be explicit about the identification of data and purpose of using it. This information should be explicitly expressed in the model without ambiguity.

Explicit transitions and their privacy implication. As characterised in [3, 4], most transitions between states occur due to explicit operation invocations (i.e., message exchanges). We refer to these transitions as "explicit" transitions. We showed in the running example that some transitions in a protocol schema may have associated privacy aspects which are identified from the privacy policies. We argue that for such transitions, one should consider any *privacy implications* generated by them.

For example, a privacy implication of the transitions `T2:SearchBook` or `T3:AddToCart` is that after they are fired, some personal data is collected; or a consequence of firing `T9:RequestDelivery` is that some personal data will be disclosed to a third-party service (i.e., the delivery service).

The proposed model should be able to express these implications.

Obligations. Some of the privacy implications could mean more than collection or disclosure of personal data. They may lead to an action that the organisation is obligated to implement.

Let us consider two of the privacy elements discussed in section 2.1: data retention and access to data. First, the data retention element states how long the data is going to remain in the organisation's system. This means that the organisation must implement an action that will remove the data from their system. For example, when the transition `T7:CancelPurchase` is fired, the privacy implication is (according to the policy) that the data collected during `T6` (i.e., order history) should be delete immediately.

Second, the access to data element states whether the collected data will be accessible by the owner later. Such access is recommended by the privacy standard bodies as a way of ensuring the high quality of the data (e.g., the owner can check and verify that the data is up-to-date). This means that the organisation is obligated to provide a user interface and operations for the users to access/update their personal data. For example, `T6:OrderBook` will collect payment information and order history. The privacy implication is that the data should be viewable by the owner.

The proposed model should be able to express these obligations. It is noted that the concept of obligations in privacy is generic, in that it represents any obligatory actions that an organisation takes as a way of preserving privacy. Our model considers the above two elements without losing generality.

4 Conceptual Modelling of Privacy-Aware Web Service Protocols

In this section, we introduce the proposed conceptual model for privacy-aware Web service protocols.

When requesters disclose personal information to a Web service provider, they want to ensure that it is stored and used according to the provider's privacy policies. Since these policies are mainly developed and maintained separately from the business process that collects and manipulates data, it is hard to check the enforcement of these policies. The proposed model will allow service providers describe the use and storage of personal data. The description is integrated with a Web service protocol using an extended state machine model. Having such a conceptual model will also enable analysis of privacy aspects such as data collection, disclosure, access, and retention in Web service protocols.

To cater for privacy policies, states (and consequently transitions) are extended beyond the traditional timed state machine model defined in [4]. Figure 2 represents `Snowy.com` of Figure 1 as an extended state machine augmented with privacy properties. Some extra implicit transitions, as they are not caused by explicit operation invocations, are added. They represent the implication of data retention, that is, deleting the collected data when their retention period is expired. The symbol `Ci` (respectively, `Di`, `Ai`, and `Ri`), $i \in \mathbb{N}^+$, within states means *Collection* (respectively, *Disclosure*, *Access*, and *Retention*) privacy property.

We use state machines for defining the privacy-aware protocol supported by a Web service (although other analogous models are possible). State and transitions have the same meaning as those described earlier. However, we generalize the approach by enabling the association of several *privacy policy properties* with states to characterize when a privacy property enforcement should occur and what are its implications (e.g., destroy information when retention period expires). We have identified the following characterizations as being useful for both conceptually describing a privacy-aware protocol and for automatically supporting its enforcement:

Privacy-aware protocol objects. Privacy-aware protocol objects refer to key elements of privacy policies such as personal data collected, the purpose of using the collected data, and the intended recipient of each type of data collected. As illustrated below, among other usages, privacy-aware protocol objects can be referenced in the attributes of privacy properties.

Requester profiles. Requester profiles characterise users invoking operations. A requester profile consists of a set of privacy attributes such as identity of user, age, purchase history, membership to a certain group (e.g., *Premier* or *Gold* member). Similarly to privacy-aware protocol objects, requester profiles may be used in the attributes of privacy properties. For example, service providers are allowed to collect personal data only when the requester is over 18 years.

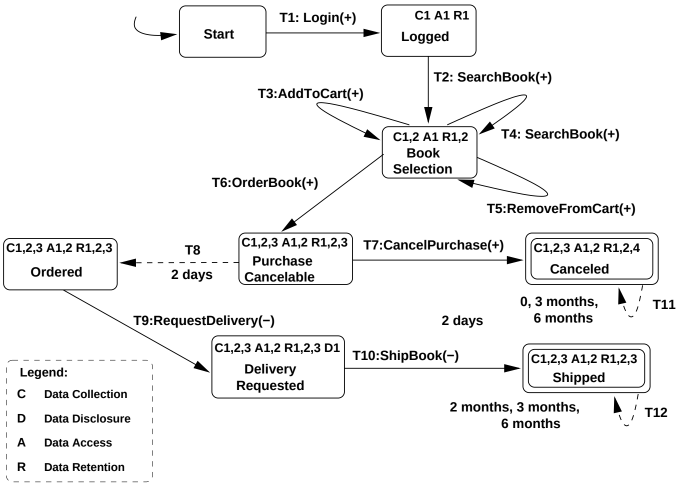

Fig. 2. The augmented protocol of Snowy.com with privacy properties

4.1 States with Multiple Privacy Properties

In this section, we will describe states using our privacy-aware protocol model. We discuss a list of privacy properties that can be used to capture privacy policies described in Section 2.1. These privacy properties consist of an initial set of privacy abstractions that we have found to be useful and commonly needed in many practical situations, namely *collection*, *disclosure*, and *obligation* privacy properties. The model is extensible in the sense that other privacy properties may be defined and used. For instance, we can add a *description* privacy property which will provide human-understandable description (e.g., textual or HTML document) about the Web service privacy policies. It will also contains any information that is not easily formalized and can not be interpreted by automated tools.

The conceptual model shown in Figure 3 represents a UML static model for the different components that constitute the privacy properties of a state. Each privacy property is described using a set of attributes. The model is also open to the extension of the definitions of privacy properties by adding new domain-specific attributes[4]. The remainder of this section gives details about the identified state privacy properties, namely *collection*, *disclosure*, and *obligation.*

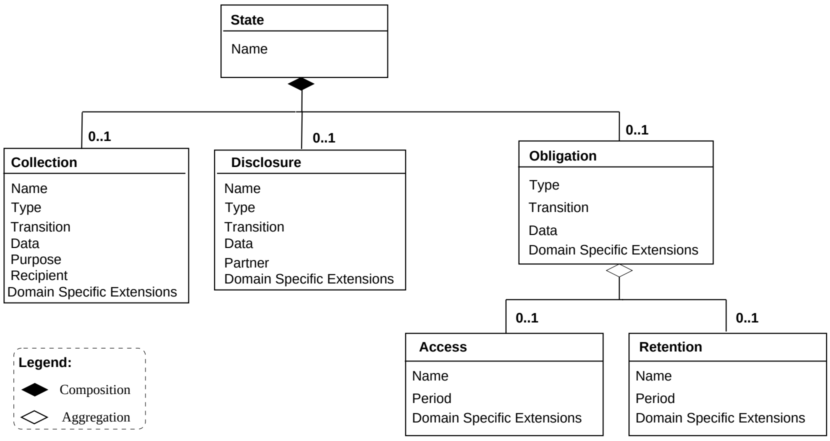

Fig. 3. UML conceptual model for privacy properties

[4] Attributes named in Figure 3 are cross-domain attributes.

Collection Privacy Property. In the extended state machine model, besides the fact that a state has a name, a *collection* privacy property expresses that data (or group of data) have been collected by the service provider when invoking an operation such as the operation `T6:OrderBook` of `Snowy.com` example (see Figure 2). More precisely, it specifies that the data are either automatically or predictably collected. Hence the collection type is either *automatic* (`Type="automatic"`) or *anticipated* (`Type="anticipated"`). The attribute `Transition` is the name of the triggered input transition of the state. The attributes `Data` specifies the data or group of data collected, `Purpose` specifies the purpose of the data, and `Recipient` expresses the recipient of the data. It is important to note that the collection privacy property will be carried over by the subsequent states until the collected data is deleted (see *obligation* privacy property below). In this case, if there are `n` collection privacy properties `C1,C2,..,Cn` within a state, they will be represented as `C1,2,..n`.

We adopt XPath[9] as a language to express queries and conditions as privacy-aware protocol objects and requester profiles are represented using XML.

Let us consider the description of the collection privacy property `C1` of the state `Logged` (see Figure 2). The following XML code represents the description of the collection privacy property `C1`:

```
<state name="Logged">
  <collection name="C1", type="automatic", transition="T1",
         data="/user[@login_data]", Purpose="", Recipient=""/>
</state>
```

The following XML code represents the description of the collection privacy property `C3` of the state `PurchaseCancelable` (see Figure 2):

```
<state name="PurchaseCancelable">
  <collection name="C3, "type="anticipated", transition="T6",
         data="/user[@order_data]", Purpose="", Recipient=""/>
</state>
```

Disclosure Privacy Property. The disclosure privacy property of a state `S` declares that data (or group of data) are shared with service partners when invoking an operation that leads to `S`. Similarly to collection privacy property, this privacy property specifies that the data are either automatically or predictably disclosed. Therefore, the disclosure type is either *automatic* (`Type="automatic"`) or *anticipated* (`Type="anticipated"`). The other attributes of this property are the disclosure privacy property name (`Name`), the name of the triggered input transition of the state (`Transition`), the data disclosed (`Data`), and the service partner to which the data have been disclosed (`Partner`). Note that the disclosure privacy property will not be carried over by the subsequent states since the purpose is to annotate only the states in which data have been disclosed.

The following XML code represents the description of the disclosure privacy property `D1` of the state `DeliveryRequested` (see Figure 2):

```
<state name="DeliveryRequested">
  <disclosure name="D1", type="anticipated", transition="T9",
              data="/user[@delivery_data]", partner="Delivery Service Partner"/>
</state>
```

Obligation Privacy Property. This privacy property models data retention and data access. We distinguish the following types of the obligation privacy property.

- *Access* to denote that some collected personal data are accessible by its owner for a specific period or indefinitely. We use `Ai`, $i \in \mathbb{N}^+$, to annotate this type of obligation privacy property.
- *Retention* to denote that certain requester's collected data are retained for a specific period or indefinitely. We use `Ri`, $i \in \mathbb{N}^+$, to annotate this type of obligation privacy property.

The obligation attribute Type indicates whether some collected data are retained (Type="R"), can be accessed (Type="A"), or both (type="mixed"), i.e., some collected data are both retained and can be accessed. The other attributes of this property are data accessed and/or retained (Data), the name of the triggered input transition of the state (Transition), the obligation privacy property name (Name), and the period of time the data are accessed or retained (Period). The implications of this property are the implicit transitions Access and/or Retention.

When an obligation privacy property contains an Access transition as implication, that is when Type="A" or Type="mixed", the privacy property will be carried over by the subsequent states until the period expiration of the accessed data. A state S carrying an Ai, $i \in \mathbb{N}^+$, annotation expresses that an implicit transition, for which S is both source and target state, is created. This means certain personal data are accessible by the requester from S. The time associated with this implicit transition is equal to Period the first time Ai appears and will decrease in subsequent states as the execution of the business process progresses. For clarity of presentation, we ommitted the representation of these implicit transitions in Figure 2.

When an obligation privacy property contains a Retention transition as implication, that is when Type="R" or Type="mixed", the privacy property will be carried over by the subsequent states until the period expiration of the retained data. A state S carrying an Ri, $i \in \mathbb{N}^+$, annotation expresses that an implicit transition is created for each final state F of the extended state machine for which F is both source and target state. The time associated with these implicit transitions is equal to Period the first time Ri appears and will decrease as the execution of the business process progresses. When the retention period expires, the corresponding retained data will be deleted (and its annotation removed from the final state).

The definition of temporal constraints use XPath[9] time functions (e.g., current-time()).

Let us consider the description of the obligation privacy properties R2 of the state BookSelection (see Figure 2). The following XML code represents the description of the obligation privacy property R2:

```
<state name="BookSelection">
  <obligation type="R", data="/user[@cart_data]", transition="T3">
    <retention name="R2", period="6 months"/>
  </obligation>
</state>
```

The obligation privacy property R2 specifies that the data will be removed after a period of six months according to the associated Retention transition. But there is no Access transition associated with it.

Let us consider now the description of the obligation privacy properties A2 and R3 of the state PurchaseCancelable (see Figure 2). The following XML code represents the description of the obligation privacy properties A2 and R3:

```
<state name="PurchaseCancelable">
  <obligation type="mixed", data="/user[@order_data]", transition="T6">
    <access name="A2", period="3 months"/>
    <retention name="R3", period="3 months"/>
  </obligation>
</state>
```

This obligation privacy property contains an Access transition A2 as implication which states that requesters are allowed to access their personal data. This access data privacy property will be carried over by the subsequent states.

Finally, let us consider the description of the obligation privacy properties R4) of the state Canceled (see Figure 2). The following XML code represents the description of the obligation privacy properties R4:

```
<state name="Canceled">
  <obligation type="R", data="/user[@order_data]", transition="T7">
    <retention name="R4", period="0"/>
  </obligation>
</state>
```

The obligation privacy property `R4` expresses that the order data collected by the service provider must be deleted immediately if the requester cancels her/his purchase of the books. This will override the obligation privacy property `R3` which states that the same order date will be deleted after three months. Hence, `R3` will not be carried over from the state `PurchaseCancelable` to the state `Canceled` but instead will be replaced by `R4`.

4.2 Privacy-Aware Protocol Formal Model

Formally, a timed Web service protocol can be modelled as a finite state machine as follows:

Definition 1 (Timed Web Service Protocol).
A Web Service Protocol is a tuple $\mathcal{P} = (S, \mathcal{O}, T, s^0, \ell)$ *where:*

- S *is a finite set of states,*
- $\mathcal{O}$ *is a set of operation names,*
- $T \subseteq S \times (\mathcal{O} \cup \{(\varepsilon, t) \mid t \in \mathbb{R}^+\}) \times S$ *is a finite set of transitions. Implicit transitions will be given the empty operation name* ε *and a time* $t \in \mathbb{R}^+$*,*
- $s^0 \in S$ *is the initial state of* $\mathcal{TP}$*,*
- $\ell : S \rightarrow \mathcal{SN}$ *is a naming function where* $\mathcal{SN}$ *is a set of state names.* □

The state machine of `Snowy.com` timed Web service protocol (see Figure 1) is defined as follows: $\mathcal{P} = (S, \mathcal{O}, T, s^0, \ell)$ where:

- $S = \{s_1, ..., s_8\}$,
- $T = \{t_1, ..., t_10\}$,
- $t_1 = (s_1, \mathit{Login}, s_2)$, $t_2 = (s_2, \mathit{SearchBook}, s_3)$,
  $t_3 = (s_3, \mathit{AddToCart}, s_3)$, $t_4 = (s_3, \mathit{SearchBook}, s_3)$,
  $t_5 = (s_3, \mathit{RemoveFromCart}, s_3)$, $t_6 = (s_3, \mathit{OrderBook}, s_4)$,
  $t_7 = (s_4, \mathit{CancelPurchase}, s_5)$, $t_8 = (s_4, (\varepsilon, 2\ \mathit{days}), s_6)$,
  $t_9 = (s_6, \mathit{RequestDelivery}, s_7)$, and $t_8 = (s_7, \mathit{ShipBook}, s_8)$,
- $s^0 = s_1$,
- $\ell = \{(s_1, \mathit{Start}), (s_2, \mathit{Logged}), (s_3, \mathit{BookSelection}), (s_4, \mathit{PurchaseCancelable}),$
  $(s_5, \mathit{Canceled}), (s_6, \mathit{Ordered}), (s_7, \mathit{DeliveryRequested}), (s_8, \mathit{Shipped})\}$.

The multiple privacy properties of a state are formally defined as follows:

Definition 2 (State Privacy Properties).

Given a Web Service Protocol $\mathcal{P} = (S, \mathcal{O}, T, s^0, \ell)$*. The privacy properties of a state* $s \in S$ *are defined as a triple* $(\mathit{Collection}, \mathit{Disclosure}, \mathit{Obligation})$ *where:*

- $\mathit{Collection} \in \mathit{Col}$ *denotes the collection privacy property with:*
  - Name *is the name of the collection privacy property,*
  - $\mathit{Type} \in \{\mathit{anticipated}, \mathit{automatic}\}$ *denotes the collection type,*
  - $\mathit{Transition}$ *is the input transition,*
  - Data *are the data or group of data collected,*
  - $\mathit{Purpose}$ *specifies the purpose of the data, and*
  - $\mathit{Recipient}$ *is the recipient of the data.*
- $\mathit{Disclosure} \in \mathit{Dis}$ *denotes the disclosure privacy property with:*
  - Name *is the name of the disclosure privacy property,*

- *$Type \in \{anticipated, automatic\}$ denotes the disclosure type,*
- *$Transition$ is the input transition,*
- *Data are the data or group of data disclosed, and*
- *$Partner$ is the service partner to which the data have been disclosed.*

– *$Obligation \in Obl$ denotes the obligation privacy property with:*

- *$Type \in \{A, R, mixed\}$ denotes the disclosure type which can be A (Access), R (Retention), or mixed meaning both Access and Retention,*
- *$Transition$ is the input transition,*
- *Data are the data accessed and/or retained,*
- *$Access = (Name, Period)$ specifies the implication of the access data obligation privacy property where:*
  - *$Name$ is the name of the access obligation privacy property,*
  - *Period is the period of time the collected data are accessed,*
- *$Retained = (Name, Period)$ specifies the implication of the retention obligation privacy property where:*
  - *$Name$ is the name of the retention obligation privacy property, and*
  - *Period is the period of time the collected data are retained.* □

The extended state machine that models the privacy-aware Web service protocol is defined as follows:

Definition 3 (Privacy-Aware Service Protocol).
A Privacy-Aware Service Protocol is a tuple $\mathcal{PP} = (S, \mathcal{O}, T, P, s^0, \ell)$ where:

- *S is a finite set of states,*
- *$\mathcal{O}$ is a set of operation names,*
- *$T \subseteq S \times (\mathcal{O} \cup \{(\varepsilon, t) \mid t \in \mathbb{R}^+\}) \times S$ is a finite set of transitions. Implicit transitions will be given the empty operation name ε and a time $t \in \mathbb{R}^+$,*
- *$P : S \rightarrow Col \times Dis \times Obl$ is the state privacy property function where Col (respectively, Dis and Obl) represents a set of Collection (respectively, Disclosure and Obligation) privacy properties,*
- *$s^0 \in S$ is the initial state of $\mathcal{PP}$,*
- *$\ell : S \rightarrow \mathcal{S}$ is a naming function where $\mathcal{S}$ is a set of state names.* □

5 Tool Support and Application of Privacy-Aware Web Service Protocols

In this section, we present the implementation of the tool supporting the model proposed as well as possible applications of privacy-aware Web service protocols.

To simplify the entire service development and management lifecycle, the following is needed:
(1) *Models and languages.* Users should have at their disposal protocol models that are easy to understand and use. The key is to include frequently needed privacy aspects, but avoid overloading the model with too many features. Another important aspect is that the privacy-aware protocol model should be formal enough to allow automated analysis and code generation.
(2) *Tools.* In the end what people really need and work with are tools. Hence, models and languages need also to be developed by considering how tools can leverage the concepts to provide concrete benefits to developers.

We have developed a privacy-aware Web service protocol tool to facilitate the creation, management, and analysis of privacy-aware Web service protocols. It is implemented as part of the `ServiceMosaic` model-driven framework for Web services life-cycle management. A description of `ServiceMosaic` framework can be found in [10]. The privacy-aware Web service protocol tool (see Fig. 4) assists designers creating privacy-aware protocol definitions. A privacy-aware protocol definition is edited through a visual interface (see Fig. 4), and translated into an XML document. The visual interface offers an editor for describing an extended state machine diagram of a privacy-aware protocol. It also provides means to describe the privacy properties of states.

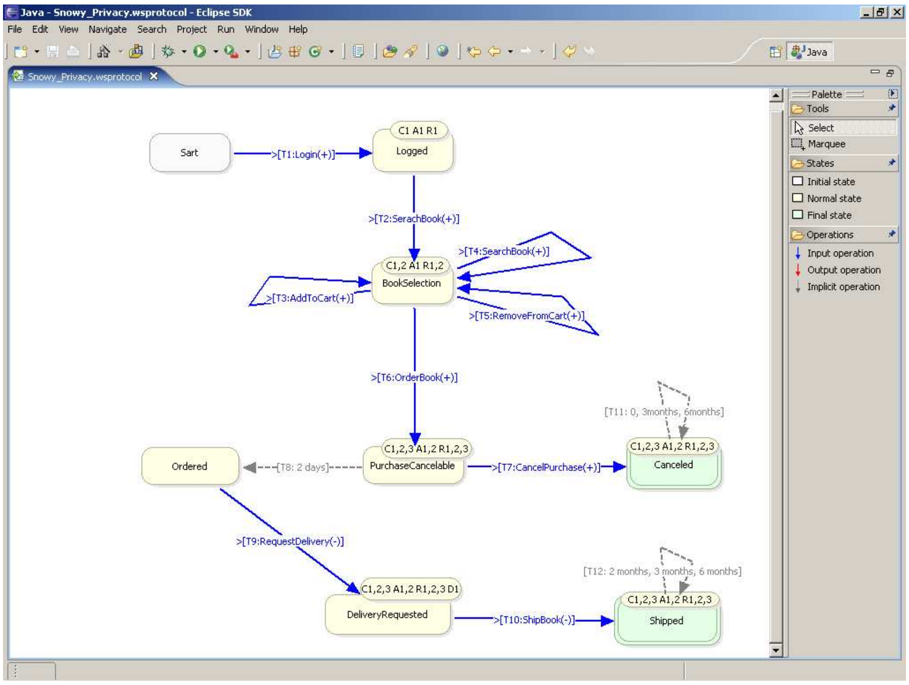

Fig. 4. Privacy-aware service protocol editor as part of ServiceMosaic

The ultimate objectives of this research is to provide developers with a privacy-aware Web service development environment. The major advantages of having a formal model are as follows [11]:

- **Automated code generation.** Privacy-aware protocols can support Web service implementation by enabling the automated generation of code skeletons. They can also be leveraged to verify whether an existing service implementation can support the privacy-aware protocol as declared.
- **Automated exception handling.** Privacy-aware protocol specifications enable the development of generic tools that read protocol specifications and verify at runtime that the interaction is occurring in compliance with the specification, raising an exception otherwise (e.g., returning a fault message when an "illegal" use of personal data is detected).
- **Development-time analysis.** During service development, protocols of clients and providers can be analyzed to identify which part of the protocols are compatible with respect to privacy policies, therefore suggesting possible areas of modifications to increase the level of compatibility with a desired service.
- **Auditing and compliance.** A privacy audit has the objective of discovering whether records of personal information are being maintained in accordance with the service provider's privacy policies. Having a formal model will help in developing an auditing framework. Furthermore, privacy-aware protocols analysis and management provide opportunities to understand whether the service is compliant with certain organisations' privacy policies or guidelines.
- **Change management.** Web services operate autonomously within potentially dynamic environments. As a result, their privacy policies may evolve, e.g., because of changes to laws and regulations and changing business strategies. Consequently, services may fail to invoke required operations when needed.

6 Related Work

There are many active projects in the research community looking into various aspects of privacy and computing. Especially, we identify two areas of the research: language and modeling, and privacy policy enforcement.

The two areas are not totally separated in that an architecture of a privacy enforcement framework may depend on what kind of modelling capabilities are assumed. Our work is directly related to the language and modelling aspects of privacy in that we propose a privacy modeling technology for Web services. However, we view privacy enforcement being one of the possible applications of our model.

Language and modeling: We briefly introduced the privacy language standards in Section 2.2. Some improvements on one of the language P3P were suggested by [12, 13]. In [13], the authors proposed simplified syntax, especially the concept of "consent-block" was introduced to model user consent (i.e., opt-in/out services) which can be associated with multiple statements. [12] analysed formal semantics of P3P using a relational model, providing data-centric or purpose-centric perspective. [14] points out that the acceptance rate of P3P by Web sites is still low. The EPAL language considers data access control aspects within an organisation. However, how obligations (e.g., data retention) is defined and enforced within the framework, which is very much focused on access control rules, is not clear.

In [15], the authors suggest *Integrated Privacy View* system in which visualised privacy policy "anchors" are attached to each HTML Form. Such anchors are used to link the Form fields to specific part of P3P statements. This helps users to easily understand how and where privacy policies apply to the Web site. Although the approach limited to Web pages (i.e., extension of HTML code), the similar idea can be applied to business processes.

The the development of the language and modelling in privacy has been focused on caterering the organisations' needs to publish privacy notices or enforce privacy data access policies within an enterprise. To the best of our knowledge, no specific work has been attempted for developing a privacy model over a business protocol by annotating privacy aspects.

Privacy enforcement: In [16], the authors introduce the concept of *Hippocratic database*. In a Hippocratic database, portions of personal data are hidden from inappropriate users when the policy does not allow it. The technique modifies an SQL query to consider a relevant privacy policy statement, purpose and recipient. [17] presents another approach to authorised privacy data acess in Web services. The actual prevention of inapproriate disclosure happens at the database component of the architecture via query rewriting. Enforcing policies are still an open problem and authorised data access through extended access control mechanism is not enough to solve privacy aspects such as data retention obligation.

Our focus, in this paper, is on formally incorporating privacy policies within business protocols.

7 Conclusions

In this paper, we proposed a conceptual model for privacy-aware Web service protocols. The description of the use and storage of personal data is integrated with a Web service protocol using an extended state machine model. This conceptual model enables model-driven development and management of Web service protocols with respect to their privacy aspects such as data collection, disclosure, access, and retention. A tool support has been implemented, as part of `ServiceMosaic`, to let designers model privacy aspects within the Web service protocol.

References

1. Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S.: Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI. IEEE Internet Computing **6**(2) (2002) 86–93

2. Curbera, F., Goland, Y., Klein, J., Leymann, F., Roller, D., Thatte, S., Weerawarana, S.: Business Process Execution Language for Web Services (BPEL4WS). http://dev2dev.bea.com/techtrack/BPEL4WS.jsp (2002)
3. Benatallah, B., Casati, F., Toumani, F., Hamadi, R.: Conceptual Modeling of Web Service Conversations. In: Proceedings of the 15th International Conference on Advanced Information Systems Engineering (CAiSE'03). LNCS 2681, Klagenfurt, Austria, Springer (2003) 449–467
4. Benatallah, B., Casati, F., Ponge, J., Toumani, F.: On Temporal Abstractions of Web Service Protocols. In: CAiSE'05 Short Paper Proceedings, Porto, Portugal (2005)
5. Cranor, L., Langheinrich, M., Marchiori, M., Presler-Marshall, M., , Reagle, J.: The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. W3C Recommendation (2002)
6. Ashley, P., Hada, S., Karjoth, G., Powers, C., Schunter, M.: Enterprise Privacy Authorization Language (EPAL 1.1) Specification. IBM Research Report. http://www.zurich.ibm.com/security/enterprise-privacy/epal (2003)
7. Amazon.com: Amazon.com Privacy Notice. http://www.amazon.com/gp/help/customer/display.html?nodeId=468496 (2006)
8. Cranor, L.F.: Web Privacy with P3P. O'Reilly (2002)
9. Clark, J., DeRose, S.: XML Path Language (XPath) Version 1.0. http://www.w3.org/TR/xpath (1999)
10. Benatallah, B., Casati, F., Toumani, F., Ponge, J., Motahari Nezhad, H.: Service Mosaic: A Model-Driven Framework for Web Services Life-Cycle Management. IEEE Internet Computing **10**(4) (2006) 55–63
11. Benatallah, B., Casati, F., Toumani, F.: Representing, Analysing and Managing Web Service Protocols. Data and Knowledge Engineering **58**(3) (2006) 327–357
12. Yu, T., Li, N., Antón, A.: A Formal Semantics for P3P. In: Proceedings of the 2004 Workshop on Secure Web Wervice (SWS'04), Fairfax, USA, ACM (2004) 1–8
13. Karjoth, G., Schunter, M., Herreweghen, E.: Translating Privacy Practices into Privacy Promises - How to Promise What You Can Keep. In: Proceedings of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'03), Lake Como, Italy, IEEE Computer Society (2003) 135–146
14. Egelman, S., Cranor, L., Chowdhury, A.: An analysis of P3P-Enabled Web Sites Among Top-20 Search Results. In: Proceedings of the 8th International Conference on Electronic Commerce (ICEC'06), Fredericton, Canada, ACM (2006) 197–207
15. Levy, S., Gutwin, C.: Improving Understanding of Website Privacy Policies with Fine-Grained Policy Anchors. In: Proceedings of the 14th International Conference on World Wide Web (WWW'05), Chiba, Japan, ACM (2005) 480–488
16. Agrawal, R., Kiernan, J., Srikant, R., Xu, Y.: Hippocratic Databases. In: Proceedings of the 28th International Conference on Very Large Data Bases (VLDB'02), Hong Kong, China, Morgan Kaufmann (2002) 143–154
17. Rezgui, A., Ouzzani, M., Bouguettaya, A., Medjahed, B.: Preserving Privacy in Web Services. In: Proceedings of the 4th ACM CIKM International Workshop on Web Information and Data Management (WIDM'02), Virginia, USA, ACM (2002) 56–62