Enabling Data Sharing in the Cloud

Marco Casassa Mont, Ilaria Matteucci, Marinella Petrocchi, Marco Luca Sbodio

HP Laboratories
HPL-2012-22

Keyword(s):
Data Sharing; Data Sharing Agreements; Cloud; Dynamic Consent; EnCoRe; Privacy; Security

Abstract:
Web interactions usually require the exchange of personal and confidential information for a variety of purposes, including enabling business transactions and the provisioning of services. A key issue affecting these interactions is the lack of trust and control on how data is going to be used and processed by the entities that receive this data. In the traditional world, this issue is addressed by using contractual agreements that are signed by the involved parties. This could be done electronically as well but there is currently a major gap between the definition of legal contracts, regulating the sharing of data and the software infrastructure required to support and enforce them. How to ensure that legal contracts can be actually enforced by the underlying IT infrastructure? How to ensure that a potentially enforceable version of the contract corresponds to the legal version of the contract? This article describes our work to address this gap through the usage of electronic Data Sharing Agreements (e-DSA). e-DSAs can be formally defined and analysed to identify inconsistencies and contradictory policies/constraints; they can then be deployed within the IT infrastructure and enforced. We specifically show how this can be achieved in a cloud scenario, where e-DSAs are enforced via policy enforcement capabilities developed in the UK EnCoRe [6] collaborative project.
Enabling Data Sharing in the Cloud

Marco Casassa Mont\textsuperscript{(1)}, Ilaria Matteucci\textsuperscript{(2)}, Marinella Petrocchi\textsuperscript{(2)}, Marco Luca Sbodio\textsuperscript{(3)}

\textsuperscript{(1)} Hewlett-Packard Labs, Cloud & Security Lab, Bristol, UK
\textsuperscript{(2)} Consiglio Nazionale delle Ricerche, Istituto di Informatica e Telematica, Pisa, Italy
\textsuperscript{(3)} work done whilst employed at HP Italy Innovation Centre, Turin, Italy

marco.casassa-mont@hp.com, ilaria.matteucci@iit.cnr.it, marinella.petrocchi@iit.cnr.it, marco.sbodio@gmail.com

Abstract

Web interactions usually require the exchange of personal and confidential information for a variety of purposes, including enabling business transactions and the provisioning of services.

A key issue affecting these interactions is the lack of trust and control on how data is going to be used and processed by the entities that receive this data. In the traditional world, this issue is addressed by using contractual agreements that are signed by the involved parties. This could be done electronically as well but there is currently a major gap between the definition of legal contracts, regulating the sharing of data and the software infrastructure required to support and enforce them. How to ensure that legal contracts can be actually enforced by the underlying IT infrastructure? How to ensure that a potentially enforceable version of the contract corresponds to the legal version of the contract?

This article describes our work to address this gap through the usage of electronic Data Sharing Agreements (e-DSA). e-DSAs can be formally defined and analysed to identify inconsistencies and contradictory policies/constraints; they can then be deployed within the IT infrastructure and enforced. We specifically show how this can be achieved in a cloud scenario, where e-DSAs are enforced via policy enforcement capabilities developed in the UK EnCoRe [6] collaborative project.

1. Introduction

Sharing data among groups of organizations and/or individuals (hereafter generally referred to as entities) is a key necessity in modern web-based society and at the very core of business transactions. However, data sharing poses several problems including privacy, data misuse and/or abuse, and uncontrolled propagation of data.

Often organizations use legal documents (contracts) to regulate the terms and conditions under which they agree to share data among themselves. However, the constraints expressed in such contracts remain inaccessible from the software infrastructure supporting the data sharing itself. Traditional legal contracts are in fact written using natural language, which, from a computational point of view, is complex, difficult to parse, and prone to ambiguity. It is therefore extremely difficult to:

- Systematically verify inconsistencies within a contract itself or between a contract and other pre-existing ones;
- Automatically build a set of policies that can consistently enforce the terms and conditions expressed in a contract;
- Relate violations of policies to the terms and conditions expressed in a contract.

There is therefore a gap between a traditional legal contract regulating the sharing of data and the software infrastructure supporting it.
Our work aims at filling in this gap through electronic Data Sharing Agreements (e-DSA).

An e-DSA is a human-readable and machine-processable contract regulating how organizations and/or individuals share data among themselves. Compared to a traditional legal contract, an e-DSA enables automated verification/validation, and automated generation of enforceable policies through a manageable and consistent lifecycle. Such policies can then be deployed into and used by the software infrastructure controlling the flow of data among organizations and/or individuals participating in the agreement. Policies violations can also be traced back to the statements in the e-DSA.

An e-DSA is essentially a multilateral agreement consisting of two parts:

- Predefined legal background information (which is usually available from a template);
- Dynamically defined information including the definition of the validity period, the entities participating in the agreement, and, most importantly, the statements that constraint how data can be shared among entities (such statements usually include authorizations and obligations).

![Figure 1: main e-DSA lifecycle phases](image)

An e-DSA is usually subject to a lifecycle, which includes a subset of the following phases: negotiation (participating entities agree on the legal template for the e-DSA), authoring (entities edit the dynamic parts of the e-DSA, including authorizations and/or obligations statements), analysis (verification tools identify possible problems and/or inconsistencies), mapping into enforceable policies (a set of enforceable policies are derived from the e-DSA, and enforced at run-time by a software infrastructure), enforcement, change management (an entity requires changes to some parts of the e-DSA; this usually moves the e-DSA back to the authoring phase), and disposal (the e-DSA has reached its end-of-life, and it is disposed of together with associated enforceable policies). Figure 1 illustrates the main e-DSA lifecycle phases, and maps them to the corresponding human actors and supporting software tools.
2. Cloud Scenario

We consider a cloud computing scenario as a significant context where to describe e-DSAs, related policy definitions and their enforcement. We believe that e-DSAs and their automated enforcement are key to further enable business interactions and information flows within the Cloud, by providing more assurance and control on data.

Figure 2 shows an example of involved players in the Cloud and related information flows.

![Cloud Scenario Diagram]

**Figure 2: Cloud Scenario**

In this scenario, multiple Cloud Service Providers (CSP) are available in the Internet. A customer uses the services supplied by a specific CSP to access online travelling, printing, office applications, etc. In order to access these services, customers need to register and disclose personal data, inclusive of address, financial details, etc.

In order to provide the required functionalities, a Cloud Service Provider might need to interact with other Service Providers and share relevant data to enable the business transaction. For example, a travelling service might need to interact with an external billing service and flight reservation service in order to supply the required service to the customer.

In all these interactions, personal and confidential data needs to be gathered; it can potentially be analysed, processed and exchanged with other parties. A key issue highlighted by this scenario is that both the customer and service providers might quickly lose control on data when this data is exchanged between different parties in chains of interactions. Customers might desire to retain control about: how their data should be used; who can access it, etc. They might want to: dictate the purposes for which data can be disclosed to third parties; impose constraints on the retention time,
notifications, etc. Similar comments apply to a service provider disclosing information to third parties.

In other words, the entities that disclose information would like to express preferences (including privacy preferences) on how their personal and confidential data should be handled along with access control and obligation constraints. Some specific examples of authorization policies for access control and obligation policies follow:

- **Authorization Policies for Access Control**
  - Data of my credit card can be accessed by Service Provider 1 (SP1) only for Business Transaction purposes;
  - My email address can be shared with SP2 and SP3 only for business transaction and goods delivery purposes;
  - My email address details must not be shared with SP4.

- **Obligation Policies**
  - I want to be notified by email every time my data is accessed;
  - I want to be notified every time my credit card is disclosed to another Service Provider;
  - I want my data to be deleted after 1 year if not accessed/used.

Interestingly, the stated constraints might need to be enforced by all the entities involved in a chain of data disclosures, e.g. in the example, by the Travelling Service, the Printing Service, the Flight Booking Service, etc.

Furthermore, the customer might, over time, periodically change their mind and modify some of their preferences and constraints. These changes should be propagated through the chain of disclosures as well.

In this context, we believe that e-DSA, coupled with enforcement mechanisms, provide a key contribution in capturing constraints and ensuring that they are fulfilled at the IT and operational levels. Our work provides tools and capabilities to organisations to:

1. Enable users (customers) to jointly define e-DSAs and related constraints, by means of wizards/editors;
2. Deploy mechanisms to manage and enforce e-DSA;
3. Deploy mechanisms to agree and share e-DSAs with third parties when disclosing personal and confidential data;
4. Enable users to update e-DSAs and propagate changes to the various points where data have been disclosed.

We envisage a situation where:

1. Online services are supplied by Cloud Service Providers to end-users (customers) or trusted third parties (acting on their behalf) to support the authoring and verification of e-DSAs;
2. e-DSA enforcement frameworks are used by Service Providers to enforce eDSAs.

Specifically, the required e-DSA enforcement capabilities can be provided by services and mechanisms developed by the UK EnCoRe collaborative project [5], in particular by means of the EnCoRe Architecture and related EnCoRe Toolbox [7]. The EnCoRe Toolbox not only provides mechanisms for the management and enforcement of e-DSA authorization and obligation policies, but it also supports the tracking of data whereabouts (once disclosed between multiple entities) and the propagation of relevant policy constraints (and preferences) to third parties, along with the disclosed data. In this context, EnCoRe supports the propagation of e-DSAs and their updates.
between the involved parties.

Figure 3 provides a high level overview of how e-DSAs and EnCoRe capabilities can be deployed in a Cloud Computing scenario:

![Cloud Scenario involving e-DSA management and enforcement](attachment:image)

**Figure 3: Cloud Scenario involving e-DSA management and enforcement**

Specifically, our work introduces e-DSA management and enforcement services, including: the Authoring and Verification of e-DSAs; the EnCoRe Enforcement Framework. The latter includes the capabilities to map e-DSA authorization and obligation policies into enforceable policies.

These e-DSA management and enforcement capabilities can be provided in a variety of ways:
- Cloud Providers can provide them as shared services (or platform-level services) to be used by the various involved Service Providers;
- Service Providers might decide to deploy their own e-DSA management and enforcement mechanisms;
- Hybrid approaches are also possible, depending on the involved transactions and data to be shared.

The remaining part for the article provides additional details about e-DSA management, in terms of authoring and validation capabilities, and their enforcement via the EnCoRe solutions.

### 3. e-DSA Authoring

The e-DSA Editor\(^1\) is a lightweight Web-based application allowing users to edit an e-DSA by using

---

\(^1\) The e-DSA Editor and related technologies are the subject of the international patent application
a controlled natural language. The e-DSA Editor is a key component to implement the e-DSA lifecycle: by using a controlled language, our Web-based Editor simplifies the creation and modification of an e-DSA by non-technical users, and, at the same time, it ensures the formal representation of the e-DSA content, thus enabling its automated processing. Indeed, the controlled natural language is used for the authoring of authorizations and obligation statements. An external customizable vocabulary defines the set of terms, actions and predicates that users can combine to build these statements. Figure 4 illustrates the high-level architecture of the e-DSA editor. The front-end layer is a lightweight Web 2.0 application enabling interactive editing of an e-DSA. The front-end layer relies on the application service layer for accessing e-DSA data and related vocabulary. The storage abstraction layer decouples the application from the actual storage systems (filesystem or database or content management systems, etc.) that is used for storing e-DSAs and vocabularies.

The e-DSA Editor displays the predefined legal background information from one of the available templates, and allows the user to interactively fill in the dynamic information of the e-DSA, and particularly the authorization and obligation statements. The e-DSA Editor guides the user in creating such statements by using a customizable vocabulary of relevant domain terms, and by ensuring that the resulting statements are human-readable simplified-English sentences.

Figure 4: e-DSA Editor Architecture

Figure 5 is a screenshot of the e-DSA Editor showing some sections of an example e-DSA: note the authorizations and obligations section where statements are presented in CNL4DSA [1], the controlled natural language specifically developed for e-DSA.

The e-DSA Editor drives the authoring of statements by displaying terms taken from a customizable vocabulary, and by following CNL4DSA predefined syntax patterns (for example “IF [set of conditions] THEN [subject] CAN [action] [object]”). CNL4DSA essentially defines the structure of
the allowed statements, but it remains open with respect to the actual terms used to build the statements. Such terms are taken from an external vocabulary, which is defined by a formal ontology. By exploiting the ontological relationships, the e-DSA Editor can also ensure semantic correctness of the statements.

The following is an example of the kind of policies that the user can create with the e-DSA Editor:

**IF** a data has category a credit card details AND a provider has identifier SP1 AND that data has purpose a business transaction **THEN** that provider CAN access that data

The above authorization is the CNL4DSA version of the first authorization policy for access control listed in the Cloud Scenario section. Note how natural language is constrained by the use of a predefined syntax and a controlled vocabulary with well formed semantics.

In the screenshot shown in Figure 5, the user is editing a new authorization; the list of available vocabulary terms is displayed in a pop-up window visible on the right-hand side (note that some terms are arranged hierarchically to visually represent the superclass and subclass relationships in the ontology defining the vocabulary).

When authoring a statement, the user can also make references to previously used terms (in the same or other statements). For example, in the previous authorization, the expression “that data” is a reference to the term “data” appearing at the beginning (“IF a data...”). The e-DSA Editor enables the creation of references with a simple point-and-click mechanism (the user points to the term that she wants to refer to, and the reference is automatically created).
Additionally, the e-DSA Editor can highlight references in the various statements, thus showing the implicit interconnections.

While providing user-friendly authoring capabilities, the DSA Editor guarantees that the dynamically created authorizations and obligations statements are formally encoded using CNL4DSA, thus enabling further automated processing. The final result produced by our application is an e-DSA, which embeds in a predefined legal template the dynamically created authorizations and obligations statements that constraint how entities participating in the agreement share data. The resulting e-DSA is saved as XML, and it contains both the human-readable and the machine-processable versions of authorizations and obligations. The following XML fragment shows how the example authorization introduced above is automatically translated into the formal language CNL4DSA used for e-DSA automated analysis.

```xml
<authorization id="AUTHORIZATION_1" index="0">
  <expression language="CNL4DSA-E">if ?X_2:Data hasDataCategory ?X_3:CreditCardDetails and ?X_4:Provider hasIProviderdentifier ?X_5:SP1 and ?X_2 hasPurpose ?X_6:BusinessTransaction then ?X_4 can Access ?X_2</expression>
  <expression language="UserText">IF a data has category a credit card details AND a provider has identifier SP1 AND that data has purpose a business transaction THEN that provider CAN access that data</expression>
  <expression language="CNL4DSA">if hasDataCategory(?X_2,?X_3) and hasIProviderdentifier(?X_4,?X_5) and hasPurpose(?X_2,?X_6) then can [?X_4, Access, ?X_2]</expression>
</authorization>
```

Note that variables (such as ?X_2, ?X_3) are automatically generated by the e-DSA Editor, and represent instances of terms defined in the vocabulary used to author the e-DSA. More precisely variables (such as ?X_2) are individuals belonging to the ontological class defining the term (such as “Data”). Finally, we observe that the e-DSA Editor can exploit concept hierarchies in the ontology defining the vocabularies to automatically expand statements that contain implicit meaning (generic concepts are automatically replaced by a disjunction of their most specialized subclasses).

For example, if the ontology defines:

- ECommerceWebSite and HotelBookingSystem as subclasses of Provider;

then the authorization constraint:

\[
\text{IF a data has category a credit card details AND a provider has identifier SP1 THEN that provider CAN access that data}
\]

is translated by the e-DSA Editor into a CNL4DSA statement equivalent to the following (indentation is used only for readability):

\[
\text{IF a data has category a credit card details AND ( a e-commerce web site has identifier SP1 OR a hotel booking system has identifier SP1 ) THEN}
\]
( that e-commerce web site CAN access that data AND that hotel booking system CAN access that data )

The proposed solution (e-DSA Editor) addresses the shortcomings of a traditional legal contract by combining the use of Controlled Natural Language and a simple, lightweight Web-based user interface. The e-DSA Editor allows non-technical users to easily create/edit an e-DSA, and, at the same time, it handles in the background the generation of its formal representation for further machine processing.

4. e-DSAs Validation

The e-DSA management phase includes also validation of the e-DSA statements, via the e-DSA analyser, before their actual enforcement. Indeed, the main goal of the e-DSA analyser is to check if the e-DSA policies have been authored in a way that is consistent with the authors’ requirements.

In particular, the analysis process allows the answering to questions like the following.

- Action list: what are the authorised actions in the investigated e-DSA, under a set of contextual conditions?
- Answer to specific authorization/obligations-related queries: is it true that, for a given e-DSA, or for a given set of e-DSA, subject x is authorized/obliged to perform action z on object y, under a set of contextual conditions?
- Check conflicts: is it true that subject x can perform action z on object y and that, under the same contextual conditions, subject x cannot perform action z on object y?

Answers to these questions are obtained exploiting built-in capabilities of the formal engine Maude, the analysis tool that we have chosen for the e-DSA validation process.

The e-DSA analyser consists of two parts:

- The formal engine Maude that actually performs the analysis of the policies;
- A graphical user interface that allows the user to dynamically load contextual conditions and queries for the analysis of the set of policies.

The e-DSA analyser is available online at http://dev4.iit.cnr.it:8080/DsaAnalyzerWebGUI-0.1/?dsaID=cloud3.xml. The interested reader should press the Submit for the Analysis button in order to load the running example of this paper and starts the e-DSA verification.

The engine. CNL4DSA has been designed with a precise formal semantics, based on a modal labelled transition system [5]. Thus, the language is governed by rules regulating states and transitions between these states. This allows for a precise translation of CNL4DSA in Maude. Maude is an executable programming language that models distributed systems and the actions within those systems [4]. Systems are specified by defining algebraic data types axiomatizing systems states, and rewrite rules declaring the relationships between the states and the transitions between them. Maude is executable and comes with a toolkit that allows formal reasoning about the specifications produced. In particular, the Maude facilities can be exploited to search for allowed traces, i.e., sequence of actions, of a policy specified in CNL4DSA. These traces represent the sequences of actions that are authorised, or required, or denied by the policy. CNL4DSA has been made executable by translating its syntax and formal semantics in Maude [2,3]. The Maude template used for the analysis of the policies is available online at: www.iit.cnr.it/staff/marinella.petrocchi/template.maude. This template contains static parts defining the translation from CNL4DSA to Maude and logic operators. Also, some modules are dynamically loaded depending on the kind of policies, contextual conditions, and queries that the user is going to deal with.
The graphical user interface. The GUI is deployed as a Web Application and it allows the user to query the Maude engine and visualize its results. The GUI is in charge of retrieving the set of policies that a user wants to analyse and the related vocabulary from a repository. Both the set of policies and the vocabulary are defined by means of the e-DSA Editor. The inner logic of the GUI exploits it in order to create and show a set of menus whose information is consistent with the vocabulary. The interface helps the user to create dynamic contexts, which represent the environment under which the analysis will be performed. The inner logic of the GUI updates the information according to the selected context. Furthermore, it is possible to compose different types of queries, related to authorizations, obligations, and prohibitions. Once the user selected both context and queries, the GUI sends all the inputs, i.e., the vocabulary, the CNL4DSA description of the policies, the context defining the conditions on which the policies have to be evaluated, and the set of queries to Maude, that performs the analysis. When the analysis has been performed, the results are shown through the GUI.

Figure 6 shows the selection of some contextual conditions, while Figure 7 shows an example of query composition. These examples are related to the policies of Section Cloud Scenario.

![Figure 6: e-DSA Analyser: Select a context](image)

Note that all the selected contexts are automatically set to true. We assume that everything that it is not explicitly specified does not hold. Hence, the user shall select each context that is supposed to be true.
Figure 7: e-DSA Analyser: Compose a query

Finding the traces allowed by a set of policies is particularly useful to detect conflicts before the actual enforcement of those policies. As a simple example, we consider the two following authorization policies for access control, slightly modified with respect to the policies listed in section Cloud Scenario.

- My credit card data can be shared with SP3 and SP 4 for business transaction purposes
- My credit card data cannot be shared with SP4

These two policies give and deny to SP4 the possibility of accessing the credit card data. This happens when the following contextual conditions are set:

- datum has data category credit card;
- provider is SP4;
- datum has purpose business transaction.

Given this context, both the authorization and the prohibition are enacted. An alert message is then shown to the user, that can decide to go back to the e-DSA editor to modify the clauses that give the conflict, before their enforcement. The conflict detection is shown in Figure 8.
As previously mentioned, the capabilities of e-DSA Authoring/Editing, Analysis and Verification can be provided as services in the cloud, by a variety of approached. The end-user (customer) can directly use them or a trusted third party can act on their behalf.

Once an e-DSA has been analysed and its correctness checked, its constraints (authorization and obligation policies) need to be deployed within an organisation in order to be enforced.

Our work leverages the EnCoRe framework [7] which provides a configurable environment for the deployment and enforcement of authorization policies for access control and obligation policies, within and across organisations. EnCoRe supports flexible management of privacy and data subject (people, user)’s consent. We first explain the technical capabilities of the EnCoRe framework and then describe how it can be used to deploy and enforce eDSA agreements.

5. EnCoRe Policy Enforcement Framework

EnCoRe [6,7] is a UK collaborative project that involves contributors from the social, legal and technological areas. EnCoRe aims at providing user-friendly and reliable consent and privacy management capabilities to individuals and organisations. Specifically, the objective of EnCoRe is to: provide data subjects with better control on their personal data once disclosed to organisations, by enabling explicit consent on how data should be handled, via the definition of privacy preferences and supporting later changes; enabling organisation to fully enforce these privacy preferences, along with security and privacy policies, inclusive of authorization and obligation policies.

The main area of technical innovation in EnCoRe consists of the overall end-to-end mechanisms and capabilities for consent and privacy management, spanning across the various stakeholders: users, organizations and third parties. These capabilities are provided by the EnCoRe D2.3 Technical Architecture [7] and related EnCoRe Framework including:

- explicitly capturing end-users (data subjects)’ consent by means of privacy preferences;
- storing and processing preferences along with associated personal data;
- explicit representation and enforcement of privacy-aware authorization and obligation policies when handling data, accordingly to stated consent;
- tracking data whereabouts, within and across organizations;
- sharing personal data, across organizational boundaries in a secure and accountable way by leveraging the HP sticky policies approach and technology.

Figure 9 shows these key, privacy & data management capabilities provided by EnCoRe within and across organisations:

<table>
<thead>
<tr>
<th>Query</th>
<th>Expected</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a provider access a data?</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>Cannot a provider access a data?</td>
<td>false</td>
<td>true</td>
</tr>
</tbody>
</table>
Figure 9: key capabilities provided by EnCoRe

The “EnCoRe Toolbox” refers to the set of technical EnCoRe capabilities. Figure 10 shows the high-level EnCoRe Technical Architecture underpinning this toolbox:

Figure 10: EnCoRe Technical Architecture
Specifically, Figure 10 provides an overview of the key EnCoRe components, including:

- a user side plug-in to capture users’ consent for specific personal data items, by means of privacy preferences;
- a back-end provisioning component, to store personal data, references to privacy preferences (in a data registry) and configure authorization policies for access control and obligations;
- a data registry to store the actual privacy preferences and the data whereabouts;
- a privacy-aware access control module to provide access control on data, driven by security & privacy constraints and preferences;
- an obligation policy module to deal with duties dictated by privacy preferences (e.g. data deletion, notification, data minimization, etc.);
- an external workflow manager to handle interactions with third parties and exchange personal data, along with consent using the sticky policy mechanism;
- auditing/logging capabilities.

These components have been designed and implemented as independent, configurable services: they support secure communication and audit/logging capabilities. They can be flexibly deployed within an organization and at the end-user site, depending on needs and leveraging existing middleware, such as Identity and Access Management (IAM) solutions.

It is important to notice that the EnCoRe Framework implements explicit capabilities to handle (privacy-aware) authorization policies for access control and obligation policies. These policies [7] are flexible and support a representation of the e-DPA authorization and obligation policies in a way that can be enforced and monitored.

The e-DPA authorization and obligation constraints are translated into an internal programmatic representation, based on XML [8], that captures the various conditions along with references to data items. This includes:

- Constraints dictating agreed purposes for accessing and disclosing data;
- Constraints on which entities can/cannot access the data and or data can/cannot be disclosed to;
- Constraints on deletion, notification, data minimisation, etc.

An example of EnCoRe authorization policy for access control follows:

```xml
<policy>
  <name>policy-ID1</name>
  <target>Credit Card</target>
  <trigger>
    <expression>
      <and>
        <condition>Request.Obj.Location=="PII.DB"</condition>
        <condition>Context.Request.Purpose contained in Context.PrivacyPreferences.Purpose</condition>
      </and>
    </expression>
  </trigger>
  <rules>
    <rule>
      <expression>
        <and>
          <condition>Request.ThirdPartyDisclosure== Context.PrivacyPreferences.3rdParty</condition>
        </and>
      </expression>
    </rule>
  </rules>
</policy>
```
AllowedThirdParties</condition>

Request.Purpose=="Business_Transaction"</condition>

</and>
</expression>
<action>
  <decision>yes</decision>
</action>
</rule>

</rules>
</policy>

This authorization policy is a general purpose policies, covering the specific example of e-DSA authorization policy:

My credit card data can be shared with SP3 and SP 4 for business transaction purposes

The EnCoRe system, at the enforcement time, knows (for each data subjects) what the specified (privacy) preferences are, including preferences in terms of purposes, third parties where data can/cannot be disclosed to, etc. This information is available as contextual information [7].

Similarly, an example of representation of obligation policies in EnCoRe follows:

<obligation>
  <name>obligation-ID1</name>
  <type>one-off</type>
  <target>attributes</target>
  <eventList>
    <event>Event_Access_Granted</event>
  </eventList>
  <trigger>
    <expression>
      <and>
        <condition>Request.Obj.Location=="PII.DB"</condition>
        <condition> Context.PrivacyPreferences.Notify == YES</condition>
      </and>
    </expression>
  </trigger>
  <actions>
    <do>send notification to data subjects</do>
    <onViolation>Log error</onViolation>
  </actions>
</obligation>

This obligation policy can be used to represent the following e-DSA obligation:

I want to be notified by email every time my data is accessed

At the enforcement time, EnCoRe detects when the end-users (data subjects)’ data is accessed and, in such a case, it sends notifications to data subjects, if they required to.
A detailed description of the representation of EnCoRe policies and their enforcement capabilities is discussed in [7].

The EnCoRe Framework supports four common use cases:

- **An end-user discloses personal data along with consent/privacy preferences**: the system captures these via user-side plug-ins; the information is sent to the back-end provisioning component for internal configuration (via policies and the data registry). This includes setting privacy obligations in the privacy obligation manager, driven by user preferences, e.g. about notifications (on usage/disclosure of data), data deletion, etc.;

- **Employees and/or applications trying to access data for specific purposes** (e.g. marketing, transaction processing, research, etc.): the privacy-aware access control module intercepts these requests (via SQL interception and/or specific interception points within applications) and grants/denies access based on the evaluation of authorization policies for access control. These policies not only describe security constraints (who can access what) but also privacy constraints based on users’ preferences (e.g. purposes for using data, black/white lists of entities that can handle data, etc.);

- **End-user changes his consent/privacy preferences**: the end-user can change, at any time, their privacy preferences. This triggers a chain of updates of stored privacy preferences within the organisation (via the back-end Service Framework), including updates of the data registry, authorization policies for access control and obligations. If the updated preferences relate to data shared with third parties, these parties will also receive update notifications;

- **Personal data is disclosed to a third party**: the system intercepts the attempt of applications to disclose data to third parties (via locally deployed agents). If the transfer of data is authorized by the access control component, then the personal data is disclosed to the third party via the external workflow manager, by using the sticky policy mechanisms that bundle data to policies and privacy preferences [9]. The degree of stickiness (simple association or strong cryptographic binding) can be configured. The data registry is updated accordingly about the data whereabouts.

Various use cases carried out with customers and government organisations demonstrated that the EnCoRe framework and solution can be easily integrated with existing enterprise data management and Identity & Access Management (IAM) solutions, for example by leveraging IAM Provisioning solutions.

The EnCoRe framework has been used in this context to deal with the explicit deployment and enforcement of e-DSA policies and constraints.

### 6. e-DSA Deployment and Enforcement via EnCoRe

The deployment and enforcement of e-DSA within organisations is pretty much straightforward with the EnCoRe framework.

Data subjects (users), at the time of disclosing their personal data and preferences, engage in the definition of e-DSA via the specific e-DSA management services provided either by the Cloud Providers or the Service Providers.

The resulting e-DSA, once analysed and validated by the e-DSA management services, contains references to the relevant types of data items to be disclosed and the agreed authorization and obligation policies. These policies reflect, among other things, the (privacy) preferences expressed by the user.

Users disclose their personal data along with the generated e-DSA, by using the EnCoRe plug-in.

This information is intercepted by the EnCoRe Consent & Revocation Provisioning Component
(integrated with the Service Provider back-end components).

This component analyses the content of the e-DSA. It stores the data in the Service Provider’s Data Repository whilst the extracted preferences and the e-DSA is stored in the Data Registry (along with a reference to the actual data). Furthermore, it extracts the authorization policies for access control and obligation policies from the e-DSA, maps them into the EnCoRe internal representation of authorization and obligation policies (1-1 mapping) and deployed them respectively in the EnCoRe Privacy-aware Access Control and Obligation Management components.

Access to data is regulated by the EnCoRe Access Control component driven by the policies and the specific user preferences (contained in the e-DSA). The Obligation Management component deals with involved duties, such as sending notifications, deleting or minimising data, etc.

As discussed in the four general use cases, the EnCoRe framework supports changes made by users to the e-DSA and related policies and preferences. These changes are reflected in the entire chain of data disclosures via a sequence of update requests and local updates of the configuration of EnCoRe internal systems.

EnCoRe also deals with the steps involved with the sharing of data with third parties. In this context data and e-DSAs are shared to the third parties. Accountability of the third parties (i.e. in ensuring that the e-DSA agreements will be enforced) is ensured by using the sticky policies mechanism [9].

7. Conclusions

We believe that the joint usage of e-DSAs and the EnCoRe Enforcement Framework in the Cloud provides the required capabilities to enable users to have better control on their data as well as increase the level of accountability of the various involved parties. Furthermore, e-DSA management and EnCoRe provide the required enforcement capabilities by filling the gap between stating policies and ensuring that can be fulfilled by organisations.

Users can actively engage with Service Providers, at the time of disclosing their personal and confidential data, to define the suitable e-DSAs. This capability can be provided by trusted third party services (offered by Cloud Providers or run by independent Service Providers), to further simplify the overall process and support degrees of trust, e.g. by using signatures and non-repudiable mechanisms. Users can, over time, ask to renegotiate existing e-DSAs, at least in terms of changing some of the involved preferences.

Service Providers translate the e-DSAs into actionable enforcement steps and controls, via the EnCoRe framework and the involved authorization and obligation management capabilities. The locally deployed EnCoRe framework supports the propagation of data and e-DSA across multiple Service Providers by keeping track of the data whereabouts. This capability also propagates updates of modified e-DSAs and/or related preferences to the involved third parties.

Service Providers can further specialise existing e-DSA (i.e. make more restrictive) when sharing data with other Service Providers, based on transactional and business requirements.

Finally, the sticky policy mechanism, developed by EnCoRe, ensures that Service Providers are more accountable when disclosing or receiving confidential data and e-DSAs. Service Providers will have to make specific, non-repudiable assertions about their willingness to fulfil e-DSA agreements by using an approach outlined in [9].
References


