RC22756 (W0303-119) March 21, 2003
Other

IBM Research Report

Individualized Privacy Policy Based Access Control

Kathryn A. Bohrer, Stephen E. Levy, Xuan Liu, Edith G. Schonberg
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Individualized Privacy Policy Based Access Control

Kathy Bohrer, Stephen Levy, Xuan Liu, Edith Schonberg
{bohrer, xuanliu, levysn, ediths}@us.ibm.com

Keywords: privacy, P3P, access control

Abstract

Privacy regulations, industry practices, OECD [1] privacy guidelines, and policy languages such as the P3P [2] encourage companies to define their practices for handling and sharing personal information, including reasonable communication of these policies to individuals. All of these efforts focus on enterprises and the policies those enterprises set and support for personal data they collect or generate about individuals. However, one popular definition of privacy, by Alan Westin [3] , is "The right of individuals to determine for themselves when, how, and to what extent information about them is communicated to others. " This paper proposes a model for individualized privacy policies as an alternative to today's common use of enterprise-wide policies. We describe how this policy information can be used to authorize actions on personal data, replacing traditional permission or role based access control.

Introduction

Governments and industries are defining laws and regulations that attempt to limit enterprises in their use of an individual's personal data. Standards such as Platform for Privacy Preferences (P3P) [2] define a common way for websites to publish a privacy policy stating what the website does with data it collects. Other work, such as IBM's Platform for Enterprise Privacy Practices (E-P3P) [4], provide a much more detailed privacy policy language for enterprises to document and enforce their internal practices for handling personal data of customers and employees. These efforts follow OECD [1] privacy guidelines requiring enterprises to get consent for data collection, limit use to stated purposes, maintain data quality, allow an individual to access their data, and to be open and accountable for personal data handling practices.

All of these efforts focus on enterprises and the policies those enterprises set and support for personal data they collect or generate about individuals. However, one popular definition of privacy, by Alan Westin [3], is "The right of individuals to determine for themselves when, how, and to what extent information about them is communicated to others. " Today's enterprise policy approach gives individuals little or no power to control how their personal data is used. Company and website privacy policies are published to customers for their consent. Consent is required to do business with the company. The policies are typically legalistic, allowing the company broad use of the individual's data, with very few choices. In fact, customer consent is generally assumed unless the customer takes some explicit action to protest a mailing or read a website policy. Allowing some of the limited choices to be "opt-in" rather than "opt-out" is considered by enterprises as giving users a high degree of control over their privacy. To us, this seems like very little control.

This paper proposes a model for individualized privacy policies as an alternative to today's common use of enterprise-wide policies. We believe this can provide value to both individuals and companies as it increases individual trust in doing business electronically. Fear of loss of privacy has resulted in many people refusing to give out personal information in exchange for service, even when the service provider is reputable and has privacy policy enforcement in place. At the same time, new businesses and services rely on being able to obtain and use an increasingly detailed amount of personal information. Examples are listed below.

- Conventional businesses, such as banks and retailers, are turning to personalization to increase customer loyalty. Meaningful personalization relies on having access to customer preferences, life situations, financial history, and buying patterns.
- Find me – reach me services store the current phone numbers and locations of individuals to allow other people to contact them immediately. Such services enable the detailed reconstruction of a person's activities over time.
- Telematics applications collect locations, speeds, and other information from automobile sensors, enabling a whole range of possible motorist services, including automatic toll payment, favorable insurance rates for good driving, driving direction and emergency assistance, and traffic control. While such services offer convenience and economies to motorists, the risks to personal privacy are similar to the above. These services track individual habits and movement over time, as well as current location.
- Identity management services, such as the Microsoft Passport proposal [5], maintain personal data that is typically exchanged during e-commerce transactions, such as userid, password, name, address, and credit card number. Such services offer customer convenience, since customers only have to enter their data once, rather than repeatedly for each website used.

The success of these personal services depends critically on user trust, and an important component of trust is user control. Users should be able to decide who can see their personal information, for what purpose, under what circumstances, and for how long.

This paper addresses issues pertaining to user-controlled privacy. First, we present background information on related work. Second, we describe the major innovations of our model that support individualized privacy policies. The model is designed to be extremely flexible for use in either enterprise or in individual agent software as the basis for authorizing access to personal data. It can easily be extended to handle all access control decisions, not just privacy related decisions. Third, we present an evaluation algorithm for the privacy model, including conflict resolution. Finally, we show how this model may be applied to an application such as those mentioned above.

Background and related work

P3P [2] is an XML privacy policy language designed to describe the privacy policy of a website, so that browsers or other user agents can easily match a user's privacy preferences with a website's policy before giving away personal data to the website. P3P is a widely-known standard, and defines many of the basic concepts of private data usage, including purpose, retention, and recipient. We have incorporated these basic concepts from P3P in our policy information model.

APPEL [6] is a language for specifying a user's privacy preferences as the set of web privacy policies that are acceptable to users, which can be subsequently matched against a P3P privacy policy to determine whether the website policy is acceptable, and how or whether to inform the user of the decision. APPEL is well-suited as the input to a privacy-enabled web browser, that automatically matches a user's privacy preferences with a website's P3P policy and displays the result. It is less well suited for management of policies for multiple users within an enterprise or automatic release of data, which benefit from finer control of data resource, data subject and data user information.

E-P3P [4] is a privacy policy language for expressing an enterprise-wide privacy policy. It's goals are somewhat different than P3P, in that it is geared towards internal policy enforcement and business practices, rather than expression of a policy to a user agent. As such, it supports enterprise-defined user roles, purposes, and arbitrary conditions and obligations that must be fulfilled. E-P3P expresses a privacy policy in abstract user role and data categories. The association of these with actual data and users or user groups in a system is outside the scope of E-P3P. E-P3P assumes an enterprise-wide policy, where users can opt-in or opt-out.

Instance based access control extends role based access control as described in [7, 8]. With role based access control, permissions are associated with user roles instead of individual users, which facilitates administration as different users change roles. A problem with role based access control is that the number of policies needed to manage an enterprise may grow very large. This problem can be alleviated by using template policies, where different organizations can share policies. Instance based access control uses implicit relationships between roles and resources which are dynamically evaluated. Using appropriate relationships, it is possible to control access to data resources at a fine-grained level with only a few policies.

A component architecture and applications for managing individualized privacy policies are described in [9, 10]. This architecture works with the model proposed in this paper.

Policy Information Model

Major abstractions

We initiated our privacy policy model using an object oriented analysis approach. We first defined the objects that seemed central to privacy issues. The major objects involved in privacy policy considerations are personal data, data subjects, data users, data actions, and data usage. We now define each of these objects.

Privacy concerns only relate to data that is personally identifiable. Personal data (PII) is defined by the European Union [11] as data that is associated with an identifiable person. This includes much more than just information that describes a person. It would include any associated information, like membership in groups, relationships to other people, addresses, phone numbers, financial data, buying history, web transaction logs, etc. It does not matter whether PII is collected directly from an individual, generated in the course of doing business, or gathered from 3rd parties. If the data is associated with an identifiable individual, it is a target for privacy control.

A data subject is the identifiable entity, generally an individual, with whom PII data is associated. It is the data subject's privacy that is of concern.

A data user is an individual, organization, or system running on behalf of an individual or organization, that accesses PII data with intent to use it in various ways.

Data usage defines how PII data that is acted upon will be used. Data usage defines "why" a data action is being performed, and may also restrict or require subsequent actions on the data. For example, email address data might be read for the purpose of contacting a customer. That might be totally acceptable to a data subject, where getting their email address in order to disclose it to a telemarketer might be considered an invasion of their privacy. The expression of purpose of data use, and with whom the data can or will be shared, is central to privacy policies. The P3P standard also includes retention, which is a statement of whether and how long PII data will be kept, once acquired. IBM's Enterprise Privacy Architecture (EPA) [12] defines "obligations" to cover other conditional information that may require a data user to perform certain actions on PII at a future time, or in certain situations. For example, there could be an obligation to delete a PII after a certain amount of time. All of these can be categorized as data usage.

Data actions are the specific operations being performed on PII data. These correspond to the actions or permissions that often exist in access control systems. We have considered two different levels of action definition. One matches the operations common on storage systems: create, delete, modify, and query actions. The other matches higher level semantics of operations that have more specific meaning in privacy regulations and discussions. These higher level actions are defined inEPA as: release, utilize, disclose, update, delete, access, notify, add consent, withdraw consent, depersonalize, repersonalize, anonymize. We have settled on using low-level actions, since higher level actions are mappable to low-level actions with additional constraints that can be captured in the other data user, data subject, and data usage information. For example, the release action of EPA is equivalent of a create action where the data subject is also the data user initiating the action.

At an abstract level, it can be argued that data action is just a subset of data usage, since there is a wide spectrum of granularity to possible usage statements. If someone is updating a personal data record to change a purchase order status to “shipped”, is the purpose/usage “processing” the order, “tracking” the order, or “updating” the order ? These might correspond to the business process, the business task within the process, and the actual interface call to the data repository within a system. Task level access control is often implemented in business applications. Generally this is in addition to access control enforced by a storage subsystem such as a file system or database. Our system supports both action and purpose. Our assumption is that action expresses a concrete operation being performed on the data and purpose expresses a larger business intent or function for which that action is performed.

Policy model overview

Our privacy policy model is represented in a UML object model. The model supports a privacy policy made up of multiple rules. Privacy rules can be grouped into named privacy rule sets for easier reference and management. These privacy policy rules are intended to be used to authorize actions on personal data. This privacy policy information can be used in conjunction with a traditional access control system, but is rich enough to totally replace traditional access control systems. In this regard, the privacy policy model described here can be considered an advance in access control that addresses privacy as well as security concerns.

Our policy rules extend role based access control and instance based access control to add the dimensions of data subject and data usage. Where instance based access control represents policy rules as 4-tuples of:

[user group, actions, resource group, relationship]

we use:

[user group, actions, resource group, data subjects, data usages]

This is interpreted as who can perform what actions on which data items belonging to which data subjects for what purposes or under what other data usage constraints. For example, “retail companies can query shipping address information of John Doe for the purpose of filling an order”, or “ABC Credit Union can create, delete, modify and query credit union account and contact information of John Doe for any purpose, but may not disclose this information to any 3rd party”, or “anyone in Mary Smith’s department can query her calendar for the purpose of scheduling meetings”. Specifying purpose and other usage constraints is core to any privacy policy, whether enterprise wide or individualized. Restricting a privacy policy rule to a specific data subject, or data subjects, is the key to supporting individualized policies.

The data subject(s) qualifier on the policy rule says that the rule applies only to PII data associated with those individuals. So, medical records of one data subject, or group of data subjects, may be covered by a different privacy policy rule from some other data subject or data subject group. This is in contrast to current systems in which enterprises define privacy policies that apply to all their customers, and consent to that policy is tracked. In some cases, an enterprise may allow a data subject to “opt-in” or “opt-out” of certain policy rules, generally some purposes or some disclosures associated with marketing. This does not allow an individual to change the purposes or disclosures defined in a rule, but only to completely accept or reject that rule. Adding a data subject group to a privacy policy allows completely individualized policies, or policies shared across many users. Our policy model supports both data subject groups with static lists of data subjects and dynamic data subject groups where the data subjects in the group are determined at runtime by evaluating a condition that may include any data item in the system plus runtime context information. For example, “all data subjects under the age of 18”.

When data subjects are allowed to define their own privacy rules, there would generally be only one data subject associated each rule. This would be the case in user agent software, similar to that envisioned by APPEAL, and user centered applications like calendar system or mobile device applications. However, even these applications might provide rule templates that resulted in many data subjects sharing many of the same rules. In that case, even though the user may see the application as defining user specific rules, the implementation might use data subject groups to make the rule information more compact.

Enterprises can use data subject groups to define policy rules specific to regulations for customers residing in different countries, or specific to different classifications of customers, or to allow individuals to define policies on all or a part of their data. This model can also be used to manage opt-in, opt-out, and consent by adding and removing data subjects from the lists or by specifying dynamic data subject groups that depend on consent or opt-in/opt-out information.

Many of role based access control, role templates, and instance based access control functions are also useful in a privacy policy model. In particular, resource groups which aggregate PII instances as well as types (instance based access control), data groupings based on data attributes (implicit resource groups from instance based access control), data user roles defined by conditions that may include current context (role based access control, role templates, implicit user groups from instance based access control) are all concepts included in our privacy policy model. However, the way in which these concepts are incorporated into the privacy policy model is specific to our design goals and features. We will discuss the major features of our design next.

Major design features

Overall goals for the policy model were to :

- Provide a model expressive enough to handle both individualized and enterprise privacy policies
- Make the policy information easy to manage,
- Have the policy information scale to many individuals and their rules

This led to the following major design features that support the goals listed above:

Shared rule components

It is likely that policy rules will reuse the same data user groupings, PII data view classifications, data subject groupings, and privacy usage controls in different combinations in different rules. For example, a user or enterprise might classify PII data into a hierarchy of views that group various types and instances of data. Then different rules simply cover different parts of this view hierarchy. For example, a user might group their data by sensitivity. Rules then give various data users the right to take certain actions for certain purposes on one or more of these data views. Similarly, users or enterprise are likely to define groups corresponding to roles of various people in their lives or organizations. User might have groups for business colleagues, friends, family, websites they use, etc. A bank might have user groups for relationship managers, tellers, loan officers, administrators, etc. Different privacy rules would allow a different set of these user groups to take action for various purposes on different data.

It is impossible to know whether an enterprise or individual will want to reuse user groups, data subject groups, PII data classifications, and data usage specifications or whether they will need to define unique groups, data classifications and usage specifications for a rule. In general, it can be assumed that both these situations will arise.

Our model maximizes reuse by allowing each part of a rule to be shared by multiple rules. Each privacy rule references its potentially shared parts: a data view, a data subject, a data user, a data action control, and a privacy usage control. Sharing the rule parts can reduce the administration of rules, increase the understanding of rules, and result in more compact representation of policies both in storage and in presentations to users. The model maximized reuse in one other way, through the use of composite object hierarchies. This is described the following section.

Flexible composition hierarchies

Role based access control has the notion of grouping users into roles that are associated with permissions. Instance based access control extends this notion to also group resources, either by type or by instances having certain other characteristics. Directory systems such as LDAP support group hierarchies which are tree structures. Hierarchies are very natural when modeling the real world. Data user hierarchies can correspond to organizational hierarchies. Data view hierarchies can correspond to composite data items, where one aggregate data item is composed of a number of other data items. Our model has evolved from using lists of data users and user groups, lists of data subjects, and singly rooted tree hierarchies of data resources to a consistent use of a Composite design pattern [13]. This allows either individual objects or groupings of objects to be the direct target of a rule, or composed into larger groupings. It also enables maximum reuse of groups, improving scalability and management. We apply this pattern to data users, data subjects, and data resources.

In particular, allowing a composite data view to be contained in more than one other data view improves reuse and usability. Data views in the composite hierarchy can represent both aggregate data entities and classification groups, or categories, to which these aggregate data entities belong. Consider the case of modeling P3P data groups as views. In P3P these groups may be associated with more than one P3P category. Our first approach leads to duplication, as seen in the case of Group2 and Group2' in Figure 1. Allowing n-n containment results in Figure 2 below, which eliminates duplication.

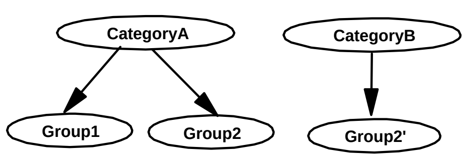


Figure 1. Singly rooted hierachy

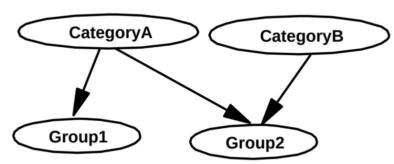


Figure 2. n-n containment hierarchy

The composite pattern is similarly useful for defining user groups corresponding to roles that are important in a privacy policy. These roles may not be the same roles or groups already defined for an enterprise's business processes. Composite data user groups can be defined to correspond to existing system groups. These system user groups can then be added to one or more of the policy composite user groups as desired. This leads us to the other important use of the composite pattern, maintaining separation of logical privacy policy from concrete deployment information. This is described in the next section.

Separation of logical policy from concrete deployment

Let's amplify the example above. An enterprise or user defines a logical privacy policy in terms of user roles that make sense in the context of protecting their privacy and in terms of data classifications desired for differing privacy rules. For privacy control, user roles might be specified in terms of relationships to the data subject: family, employer, trusted businesses (subject's bank), other businesses. Data classifications might be specified as medical, financial, contact, shopping preferences, demographic; or, might be specified along a completely subjective dimension such as very sensitive, somewhat sensitive, disclosed as needed, public. Privacy rules can be defined against these "logical" groupings. But, at some point specific data users, user groups, and data types or instances need to be placed in these logical groups. The Leaf objects of the Composite pattern serve this purpose. There can be different types of leaf objects for different concrete specifications of information. As new concrete data items and data users are encountered and need to be added to rules, this can be done simply by assigning new leaf objects to the existing composite objects, where the composite objects represent the logical groups the privacy rules are written against. The appropriate privacy rules will then apply to the new leaf objects. Figure 3 illustrates mapping concrete data items into the system by classifying them as Contact Information.

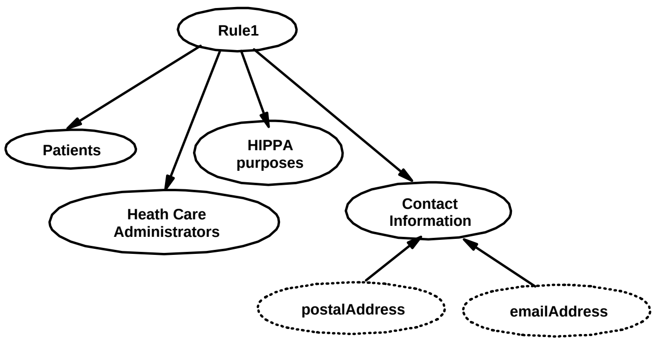


Figure 3. Deployment data mapping

In our model there are two leaf data view classes. One allows specifying data according to an object model for the data. Data can be specified by type, property of a type, instance, or property of an instance. The type hierarchy is applied. An example from the CPExchange [14] data model would be to have a rule that specifies "PartyRole", which would be applied to requests for any subtype of PartyRole data such as Employee or Customer data. The other leaf data view class in our model does not require knowledge of the data model, it just uses a naming convention for data items. Data item names are strings with "." separated substrings. Strings that match from the beginning are interpreted as describing portions of the same aggregate data item. For example, a "address.zipcode" is interpreted as part of an "address". A rule that referenced a composite containing an "address" would apply to an action on an "address.zipcode" data item.

Similarly, there could be different leaf objects for data users that supported identifying data users according to different user registries or tokens. There might be SAML [15] Subject leaf objects, LDAP distinguished name leaf objects, etc.

Dynamic groupings

Instance based access control introduces support for implicit user groups and implicit resource groups. We call this function "dynamic groups", and we allow the dynamically evaluated membership conditions to rely on any data in the system - not just properties of a user or a data item. This allows policies such as "my employer and department members can use my mobile device phone numbers to contact me". In this example, "my employer" and "my cell phone" might both be expressed by dynamically evaluated conditions. A dynamic group definition would include all users who are employers of the data subject. A data view definition would include phone numbers for mobile devices. Regardless of how the data subject then changes departments, the appropriate colleagues will be able to contact the data subject using their cell phone.

Supporting conditions on arbitrary data in the system allows our dynamic group support to also provide the function of the "relationship" dimension in instance based access control policies. The "relationship" in instance based access control is defined as a relationship that a user must have with a resource item. A similar notion is useful in privacy policies, but the relationship of interest is often between a data user and a data subject. For example, "only a patient's doctor can see his or her medical records".

When dynamic definitions are in place, new users , data subjects, and data can be added to the system and automatically covered by policy rules, without even the need to define new leaf objects. However, dynamic definitions can decrease the performance of the system because the detailed information values needed to evaluate the conditions must be accessed, and the condition evaluated.

Multiple personnas for data subjects

One of the major concerns of privacy advocates is the need to protect identity, and particularly to prevent a "global identity" for individuals. The concern is that while a few pieces of personal information in isolation may be fairly harmless, all the information about a person in aggregate would be present a huge loss of privacy. This is because, in such a situation, a few small pieces of information about a person could be enough to identity them with the total set of information. The resistance to use of social security numbers as general identifiers for individuals, rather than just tax identifiers, is based on this concern.

To alleviate these concerns, our model represents data subjects by "personnas", where a personna is some set of PII data. An individual may have more than one personna that is used in privacy rules. The data in these personnas can be distinct, or some data items may belong to more than one personna. The model does not require an association of data subject personnas to an identifiable person or user, although many systems will associated a personna with a registered user. In this case, privacy concerns can still be mitigated by allowing an individual to have multiple userids.

Superset of security access control

By allowing data views to be defined over any data, not just PII data and by allowing the data subject role to be ignored (or set to "everyone"), the model can be used to express general access control rules. These rules can be as expressive as any of the role based, template role, or instance based access control systems discussed in the background material. In addition, the data usage "purpose" can be used for task based access control that is often implemented at an application level. The dynamic role and dynamic view capability supports access control conditions on data users and data resources. Adding the notion of obligations from EPA, would also support provisional access control systems. In summary, we think this model is a powerful and interesting superset of access control system features.

Policy Evaluation Algorithm

A privacy policy is expected to be used to authorize requests for actions on personal data. We now describe the evaluation algorithm used to authorize requests according to individualized privacy policies.

Each request for authorization must include the authentication information of the requesting data user, a specification of the data subject personna whose data is being requested, the set of data resources being acted upon, the action to be performed on the data, and the privacy usage controls that the user agrees to apply to that data. We can represent a request input as a 5-tuple: (data user, data subject, action, data set, privacy usage), where user, data subject and action are single valued strings, and data set is a collection of multiple data types or instances, and privacy usage contains a set of privacy usage controls applying to some data items or types in the data set.

The user's authentication information is used to match the user to a data user group in the privacy rules. The identified data subject personna in the request must be matched to a data subject group in the rules. The data item in the requested data set should match the data views in the rules by either type or name - depending on the Leaf views used in the rules. The requested action must match the actions specified in the rules. The user privacy policy must match the rules' privacy usage control hierarchy.

Each request (data user, data subject, action, data set, privacy usage) is evaluated by the following steps:

1. Find all rules R1 that satisfy the given tuple(data user, data subject, action)
2. For each data item in the input data set, choose the most applicable rules from R1 based on precedence and /or specificity.

3. For each resulting rule in R2 from step 2 , match request (privacy usage) with the rule privacy usage control. All the privacy usages specified in the request have to be covered by a single rule to be matched.
4. For all rules obtained from step 3, make the authorization decision by conflict resolution.

Two-phase Rule Retrieval

To authorize a request, the first step is to retrieve all the rules that apply to the given tuple (user, data subject, action). Quite often, authorization rules are stored in relational database , LDAP or similar data management systems. Most of these systems have powerful and efficient optimization techniques for query retrieval. In our system implementation, DB2 is used to store our policy rules. To leverage the strong optimization mechanism that underline DBMS provides, we try to query the rule repository using a search criteria combining all constraints on table attribute values. However, our policy model provides the flexibility to express dynamic groups for data users, data users, and data views. The evaluation of dynamic groups may have to be performed dynamically using run-time context information, and thus it is impossible to search for satisfying rules using only a database query. To accomplish both performance and flexibility, we use a two-phase strategy for retrieving rules: rule-filtering and rule-refining. At the filtering phase, the rule repository is retrieved based on static constraints, and at the refining phase, the resulting rules from the filtering phase are evaluated based on dynamic constraints, specificity, and etc.

As described in the previous section, our policy model provides a lot flexibility in expressing data user and data subject dimensions for different application requirements. For example, either a data subject or data user can be an individual, or a static group list, or a dynamic group specified by a query. In an application supporting individualized privacy, such as calendar system or mobile device applications, the data subject dimension in the rules usually is an individual, and the data user dimension usually is an individual user or a user defined static group. In this case, retrieving rules that satisfies the given tuple (user, data subject, action) can be accomplished simply by querying the database, which is done in the rule-filtering phase.

On the other hand, retrieving rules with dynamic groups needs rule-filtering and rule-refining phases. A query based on action is performed at the filtering phase to find all rules for the specific action, and dynamic groups are evaluated at refining phase to eliminate false hits. Usually, dynamic groups are most useful in applications with a set of policy templates used for large set of users, such as enterprise privacy system. In such systems, there are usually a limited number of rules in the rule repository, and therefore, the number of rules that need to be refined is limited. As described later, rule caching and indexing techniques can help to improve performance.

Rule Precedence and Specificity Checking

Our model supports both rule precedence and view specificity to allow some rules to override others. To find the rules applied to a data item, the data view hierarchy is searched for views that include the data item. A data item can be identified by name, type or instance in a static leaf view, or identified in the result of a dynamic leaf view. The rules attached to such a leaf view, plus the rules attached to any composite view that contains the leaf view are applicable to this data item.

Rule precedence is a primary way to specify the priority of rules. Rules with higher precedence override rules with lower precedence. Introducing rule precedence is motivated by the need to allow certain management rules(such as legal rules) to have the privilege to override user defined rules. Rule precedence is also an easy way for users to understand and manage their rules. For example, a data subject DS1 can specify two rules to apply to the same data user group UG1. The first rule R1, with precedence 1, specifies that UG1 can read DS1's rolePlayer type information for business purpose, and can retain the data forever. The second rule R2, with precedence 2, specifies that UG1 can read DS1's social security number for business purpose, but "no-retention". Because RolePlayer type contains social security number, both R1 and R2 covers the usage of social security number. However, to authorize a request from a user in UG1 that asking about DS1's social security number, R2 will be considered to have higher priority than R1 because of its higher precedence.

In our policy model we allow views to be interpreted as more or less "specific", and to use the notion of specificity of views to support rule exceptions. Our exception mechanism attaches "specificity" to the position of views in the composition hierarchy. As described before, a view hierarchy specifies a composition of views. A composite view can contain multiple leaf views, and it is considered less specific than the leaf views. So if rule Rl is attached to a leaf view and rule Rc is attached to a composite view, R1 is considered to be more specific than Rc. The more specific rule is considered an exception to the less specific rule.

Note that when UML leaf views are used, the UML type hierarchy supports a very concise way to attach data with rules, but no specificity hierarchy is implied by the type UML hierarchy. If Rp targets a data item by some supertype, while Rc targets the data item by its actual type, then Rc and Rp both apply. Rp is just a shorthand for the group of all of its subtypes. Having more than one specificity mechanism for defining rule exceptions would be too complicated for users to manage, and make it difficult to understand which rules apply to a particular request. Therefore, we use only the view hierarchy for data specificity. It is possible to define different view hierarchies for different data subjects. The notion of rule exception by specificity could also be applied to other rule dimensions, such as data subject and data user.

The precedence is a primary way to specify the rule priority because of its clarity and simplicity. For two rules with the same precedence, we will use the view composition hierarchy for specificity checking. The specificity checking is implemented as an isolated model that can be easily replaced if an application wants different specificity checking.

Conflict Resolution

After the precedence and specificity checking, it is possible that there are more than one rule left in the final ruleset for a specific request. Since our policy model allows various authorization decisions, including "Allow", "Deny", "get consent", "notify", and etc., in the "decision" dimension, the rules in the final ruleset may result into conflict decisions. For example, two rules are in the final resultset, one is "Allow", while the other is "Deny". This raises an issue of how to manage conflicting rules, and how to resolve those rules. Checking to prevent rule conflicts in the rule repository could be difficult and time consuming. An easy way to avoid conflict is to require that each rule is given a different precedence, which can guarantee there is only one rule in the final ruleset. In our implementation, we allow users to specify conflicting rules, and we resolve the conflict when the evaluation produces a conflict final ruleset. We define a priority on the decisions, where "deny" overrides any other decisions, and "allow" is overridden by any other decisions. For any rules in the final ruleset with same decision, any one of them can be chosen.

Rule Caching and Indexing

In order to improve performance, we can cache and index authorization rules. What to base a cache on depends on common application or system scenarios. It is likely that the data being acted upon will change with each request, so caching on data subject, data user, action, and/or privacy usage makes the most sense. For many application request patterns, a rule cache can be based on data subject, or on data user, or on combination of them, such as, tuple (data user, data subject). Our system provides the capability for users to configure what keys the rule cache will use. Within a rule cache, each rule can be indexed by data views. This caching mechanism will greatly improve the performance for next request with same data subject and/or data user.

Application of Individualized Policies

In this section we give an example of a privacy policy that an employee, Mary, might define for controlling the release of personal contact information to co-workers, family, friends, and others. This policy might be managed as part of an employee service of an enterprise.

The privacyPolicy for managing contact information consists of two privacy rules, identified in the privacyRuleGroup by the object identifiers PR1, PR2.

```
<privacyPolicy>
       <policyName>contact info</policyName>
        <description>Policies for managing contact information</description>
        <privacyRuleGroup>
                <privacyRuleId>PR1</privacyRuleId>
                <privacyRuleId>PR2</privacyRuleId>
        </privacyRuleGroup>
</privacyPolicy>
```

The first rule PR1 grants permission to read telephone number , email, and legal name for the purpose of contact. The privacyRule element itself contains references to the other rule component elements, the data view (dataViewId), privacy usage (privacyUsageControlId), the data user (dataUserId), and the data subject (dataSubjectId). These component elements, which are defined below, can be shared by other privacyRules.

```
<privacyRule>
        <oid>PR1</oid>
        <ruleName>Contact information</ruleName>
        <description>telephone number and email for the purpose of contact</description>
        <decision>ALLOW</decision>
        <precedence/>
        <dataAction>READ</dataAction>
        <dataViewId>CV1</dataViewId>
        <privacyUsageControlId>PUC1</privacyUsageControlId>
        <dataUserId>CU1</dataUserId>
        <dataSubjectId>LS1</dataSubjectId>
</privacyRule>
```

The data view for privacyRule PR1 is a CompositeView, consisting of a LeafView (LV1) that includes the class email and the telephoneNumber property of the Location class plus a LeafView (LV2) that includes a specific legal name instance to be used for contact. This means that privacyRule PR1 applies to all of the data subject’s email addresses and telephone numbers, but only one specific name instance.

```
<compositeView>
        <oid>CV1</oid>
        <viewName>contact</viewName>
        <description>My friends and coworkers can contact me by email or phone</description>
        <containedViewGroup>
```

```
            <containedViewId>LV1</containedViewId>
           <containedViewId>LV2</containedViewId>
        </containingViewGroup>
</compositeView>

<uMLViewByType>
        <oid>LV1</oid>
        <classAndPropertyNameGroup>
            <classAndPropertyName>Email</classAndPropertyName>
             <classAndPropertyName>Location.telephoneNumber</classAndPropertyName>
       <classAndPropertyNameGroup>
</uMLViewByType>

<uMLViewItem>
        <oid>LV2</oid>
        <className>PersonName </className>
        <instance>NAME1</instance>
 </uMLViewItem>
```


The data subject for privacyRule PR1 is a DataSubjectPersonna, with roleName “Mary at work”. Only business information is included in this personna, for example business telephone numbers by not home phone numbers. The data users for privacyRule PR1 are the parties that may be granted permission to obtain the data for the data subject “Mary at work”. The users are specified in the compositeRole element. Users include any co-workers in the same department, plus friends.


```
<dataSubjectPersonna>
        <oid>LS1</oid>
        <roleName>Mary at work</roleName>
        <description>My work personna</description>
</dataSubjectPersonna>

<compositeRole>
        <oid>CU1</oid>
        <roleName>Users who can have access to my contact info</roleName>
        <description>
        <containedRoleGroup>
                <containedRoleId>LU1</containedRoleId>
                <containedRoleId>LU2</containedRoleId>
        </containedRoleGroup>
</compositeRole>

<dynamicUserGroup>
        <oid>LU1</oid>
        <query>Any employee in the same department</query>
</dynamicUserGroup>

<registeredUser>
        <oid>LU2</oid>
        <userid>George</userid>
</registeredUser>
```

The last component of privacyRule PR1 is the privacyUsageControl, which is match against a P3P policy of a user. The purpose of a user must be CONTACT to obtain the contact information of the data subject. Similarly, the retention, access, and recipient policy components must match as well.

```
<privacyUsageControl>
        <oid>PUC1</oid>
    <accessFlag>FALSE</accessFlag>
    <retention>INDEFINITELY</retention>
    <recipientGroup>
      <recipient enumtype>OURS</recipient>
    </recipientGroup>
    <purposeGroup>
      <purpose>CONTACT</purpose>
    </purposeGroup>
</privacyUsageControl>
```

The second privacyRule PR2 defined below denies access to all data for the purpose of telemarketing. The data users and data subject are the same as in privacyRule PR1. The privacyUsageControl and data view components are new. This rule is quite strong, it applies to all data, and no other rule currently overrides it. The dataView LV3 is a LeafView, which is the class named Distinguishable. Distinguisable is the parent class of all classes in this example, so it is the aggregate of all types of data. As long as Distinguishable is part of a LeafView with no children in the composite view hierarchy, no rule will override it by means of data specificity. Rule precedence is not being used in this rule set.

The privacyUsageControl PUC2 matches any P3P policy that has the purpose TELEMARKETING. Since the retention value INDEFINITELY is the least restrictive possible value, any other retention in a P3P policy will be more restrictive, and therefore will match. Similarly, PUBLIC is the least restrictive value for recipient, and will match any recipient in a user's P3P policy.

```
<privacyRule>
        <oid>PR2</oid>
        <ruleName>no telemarketing</ruleName>
        <description>disallow telemarketing for all of my personal data</description>
        <decision>DENY</decision>
        <precedence/>
        <dataAction>READ</dataAction>
        <dataViewId>LV3</dataViewId>
        <privacyUsageControlId>PUC2</privacyUsageControlId>
        <dataUserId>CU1</dataUserId>
        <dataSubjectId>LS1</dataSubjectId>
</privacyRule>

<uMLViewItem>
        <oid>LV3</oid>
        <className>Distinguishable</className>
</uMLViewItem>

<privacyUsageControl>
        <oid>PUC2</oid>
    <accessFlag>FALSE</accessFlag>
```

```
    <retention>INDEFINITELY</retention>
    <recipientGroup>
      <recipient enumtype>PUBLIC</recipient>
    </recipientGroup>
    <purposeGroup>
           <purpose enumtype>TELEMARKETING</purpose>
    </purposeGroup>
</privacyUsageControl>
```

Conclusion

We have presented a very general privacy policy model, which is functionally a superset of current access control models. Our privacy policy model can be applied in customized ways to meet the needs of different applications. Application logic, APIs, and user interfaces can restrict the way the components of the model are shared or associated. At one extreme, it is possible to enable data subjects to define their own user groups, data resource view hierarchy, and create highly individualized policies. At the other extreme, an enterprise could author a single privacy policy, and data subjects could opt-in by being added to the data subject list of the policy. In between these extremes, data resource view hierarchies could be shared, as well as user groups and privacy usage controls. We believe that this kind of generality is necessary to meet the needs of new applications, the acceptance of which depend on solving fundamental privacy issues. Most importantly, individuals must be able to control the use of their personal information. Framework flexibility further enhances our ability to experiment and meet future needs.

Bibliography

[1] Organization for Economic Co-operation and Development, “Recommendation of the Council Concerning Guidelines Governing the Protection of Privacy and Transborder Flows of Personal Data”, Sept. 1980.

[2] Marchiori (editor), “The Platform for Privacy Preferences 1.0 Specification; W3C Recommendation”, April 2002, http://www.w3.org/TR/2002/REC-P3P-20020416/.

[3] Alan F Westin, Privacy and Freedom, Atheneum, New York NY, 1967.

[4] Karjoth, Schunter, Waidner, “Platform for Enterprise Privacy Practices: Privacy-Enabled Management of Customer Data”, 2nd Workshop on Privacy Enhancing Technologies, Lecture Notes in Computer Science. Springer Verlag, 2002.

[5] v-Go Single Signon; first release 1999, White Paper 2000, http://www.passlogix.com/media/pdfs/Usable_security.pdf

[6] Langheinrich (editor), “A P3P Preference Exchange Language 1.0, W3C Working Draft”, April 2002, http://www.w3.org/TR/P3P-preferences.html.

[7] Goodwin, Raina, Rajesekharan, Philip, Thomas, Nuzzo, Balakuranam, “Advances in Policy Based Authorization in Websphere Commerce Business Edition”, ???

[8] Goodwin, Gah, Wu, “Instance-level Access Control for Business-to-Business Electronic Commerce”, IBM Systems Journal, Volume 41, Number 2, 2002.

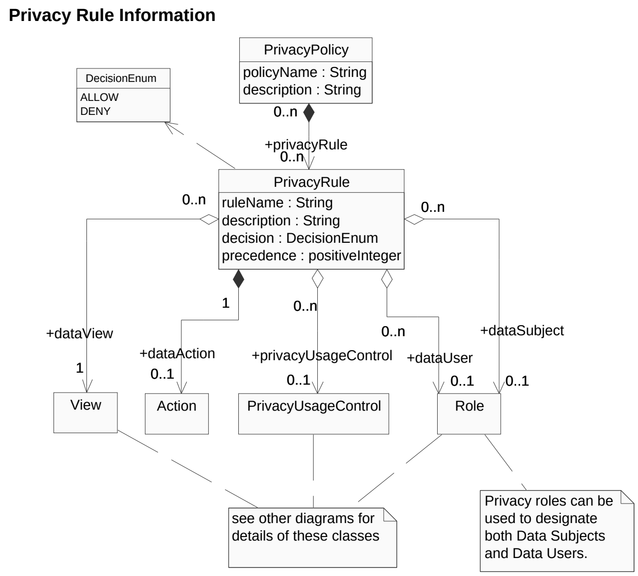
[9] Bohrer, Liu, Kesdogan, Schonberg, Singh, Spraragen, “Personal Information Management and Distribution”, 4th International Conference on Electronic Commerce Research, Nov. 2001.

[10] Bohrer, Kesdogan, Liu, Podlaseck, Schonberg, Singh, Spraragen. “How to go Window Shopping on the World Wide Web without Violating the User’s Privacy”, 4th International Conference on Electronic Commerce Research, Nov. 2001.

[11] European Parliament and the Council of the European Union, “On the protection of individuals with regard to the processing of personal data and on the free movement of such data”, Feb. 1995.
[12] Karjoth, Schunter, Waidner, “Privacy-Enabled Services for Enterprises”, IBM Research Report RZ 3391 (#93437) Jan. 21, 2002.
[13] Gamma, Helm, Johnson, Vlissides, “Design Patterns, Elements of Reusable Object-Oriented Software”, pages 163-173. Addison-Wesley, Reading, MA, 1995.
[14] K. Bohrer, B. Holland (eds), The Customer Profile Exchange (CPExchange) Specification, Version 1.0, International Digital Enterprise Alliance, Oct 20, 2000, (www.cpexchange.org).
[15] Halam, Baker (editors), OASIS Security Assertion Markup Language (SAML); Committee specification, April 19, 2002, http://www.oasis-open.org/committees/security/docs

Appendix: Complete Policy Model

Privacy rules



A privacy rule object is composed of five objects representing:

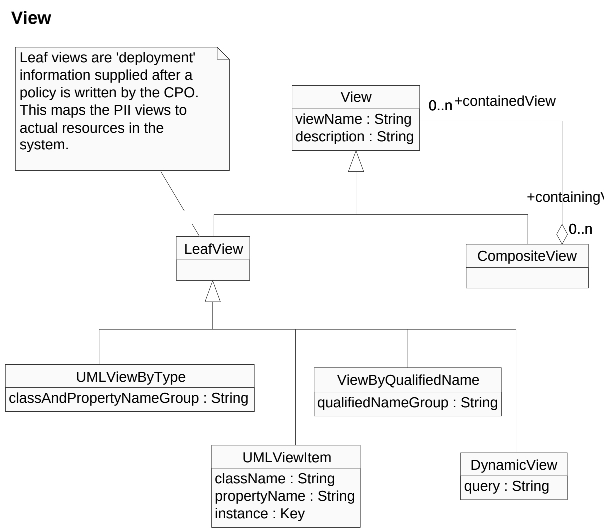
- A composite structure of data subjects whose PII data is governed by the rule.
- A composite structure of data views that identifies and classifies a subset of a data subject's PII included in the rule.
- A privacy usage control that describes usage constraints.
- A privacy action control that describes permissible data actions .
- A composite structure of data users that are authorized to take some action on the PII data in the view.

Rules can share data users, data subjects, data views, and usage controls. Data action objects are not shared. We now describe the detailed model for each of the five objects that make up a rule.

Data View Composition

The PII data covered by a privacy policy rules is specified in by reference to a View object. Authorization for an action will never be granted to personal data that is not included in a View. The referenced view object can be either a composite view object or a leaf view object. If it is a composite view object, the rule also covers all views contained in the composite view. Actual PII data items are identified in LeafViews that are contained in one or more CompositeViews. So, a rule covers all the PII data items described by any LeafView of any CompositeView contained directly or indirectly in the View referenced from the rule.

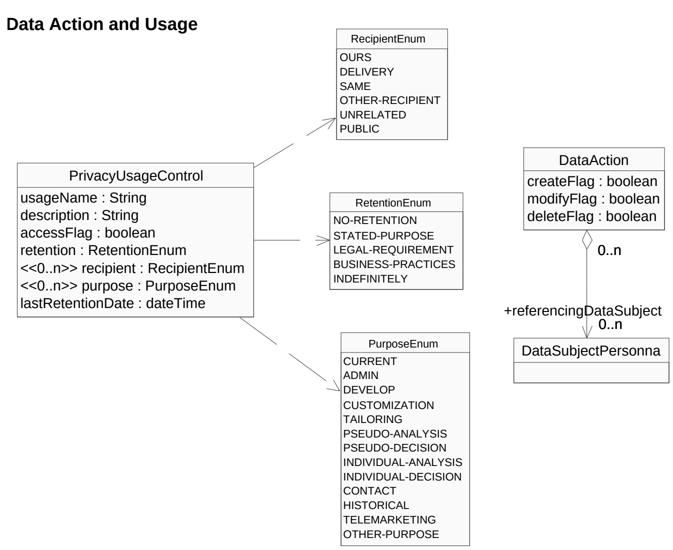
Several LeafView subtypes are supported. Other leaf view subtypes can be added, but require an extension to the evaluation engine to understand how to compare data action requests to the items described by the leaf views. The UMLViewItem allows a data items to be specified based on an underlying UML object model or equivalent XML Schema. A UMLViewItem can be specified by type, property of a type, instance, or property of an instance. A rule including a supertype would also apply to subtypes. The UMLViewByType provides a set of "classname.propertyname" strings for the UML classes, or properties of classes that are included. The ViewByQualifiedName specifies a set of data item names, where each name is a string composed of substrings separated by ".". Each substring is assumed to be a data item that is part of a larger aggregate data item represented by the preceding substring. No type hierarchy is assumed in interpreting these LeafViews, but a rule that specifies the name of an aggregate will apply to any data item within that aggregate. The name of a data item must always be given in a fully qualified form, that is all its containing aggregate data item names are given. For example, "zipcode" is not the same as "address.zipcode".



There is also support for a DynamicView. A DynamicView specifies query criteria on PII data, rather than listing view items by type or name. Any PII data that satisfies the query is part of the dynamic view. For example, a DynamicView could be defined to be "all telephone numbers with usage properties of "mobile" or "office". A DynamicView selects instance level data items based on related objects and property values. The objects and their property values must be available to the policy evaluation engine at runtime.

Data usage

A PrivacyUsageControl is associated with each rule. A PrivacyUsageControl specifies how the data being acted upon can be used or must be handled by the receiver of the data. Currently the PrivacyUsageControl properties are applied only to "read" actions, and extend the characteristics defined in the W3C Platform for Privacy Preferences (P3P) standard. The notion of *recipient* in P3P is extended to allow a list of data users to whom the data can be subsequently disclosed. The notion of *access* in P3P is extended to allow access to specified for either any data view. The notion of *retention* in P3P is extended to allow a specified end date past which the data may not be retained.



Data action

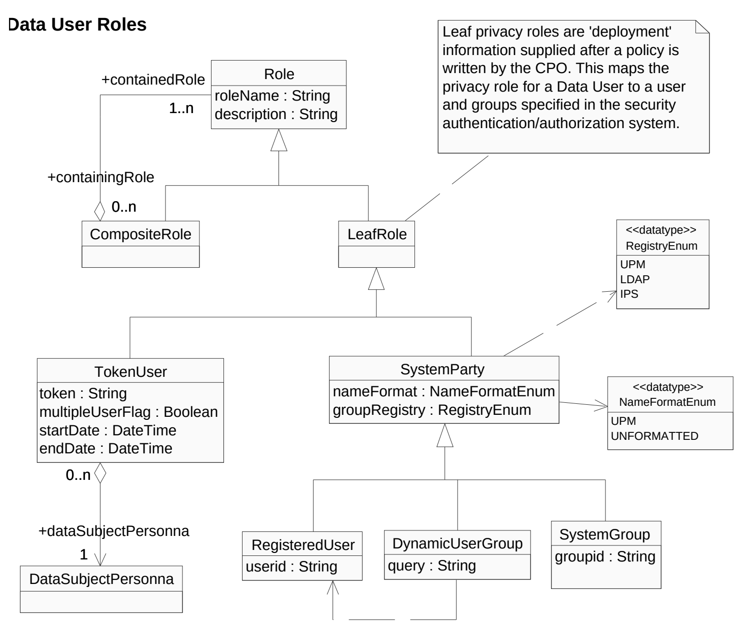
An DataAction object specifies the actions that can be performed on data. Our current implementation defines actions at the storage subsystem level: create, delete, modify, and query. A DataAction object can also specify a list of data subject "personnas" that are allowed to create links in that personna to the data subject PII data covered by this rule. This supports sharing of PII data across multiple personnas of the same or different people. For example, a husband and wife could share contact and financial information, or all the employees of a company could share the same employer company information. Note that creating a link to an object in personna A from personna B effectively adds the object to personna B.

Other actions could be supported. For example, the higher-level IBM Enterprise Privacy Architecture actions can be mapped to the more storage actions with restrictions on whether the parties involved in the action are data subjects or data users as follows:

- release : create PII from data provided by data subject
- update : modify PII
- delete : delete PII
- utilize : query PII by data user within organization holding the PII
- disclose : query PII by an data user outside the organization holding the PII
- access : query PII by data subject
- notify : inform data subject about PII policies
- add consent : modify by data subject of their consent PII
- withdraw consent : modfiy by data subject of their consent PII
- depersonalize : query PII in order to transform it so it is no longer PII
- repersonalize : query depersonalized data (and PII key information) in order to transform data back into PII
- anonymize : query PII in order to transform data so it is no longer PII, and can't be repersonalized

Data users and groups

The data users to which a rule applies is specified by a Role. A Role may be a composite of other roles, or may be a LeafRole. A Role may be part of more than one CompositeRole. Privacy rule can be written in terms of CompositeRole objects that correspond to logical entities relevant to privacy concerns. CompositeRoles can be used to model groups of users, where the users are represented by LeafRoles. A system often already has a directory service used for authentication and authorization that defines system users and system groups of those users. These system groups generally do not correspond exactly to the roles required for the privacy rules. But, system groups can also be mapped to LeafRoles that are then assigned to the various privacy roles.



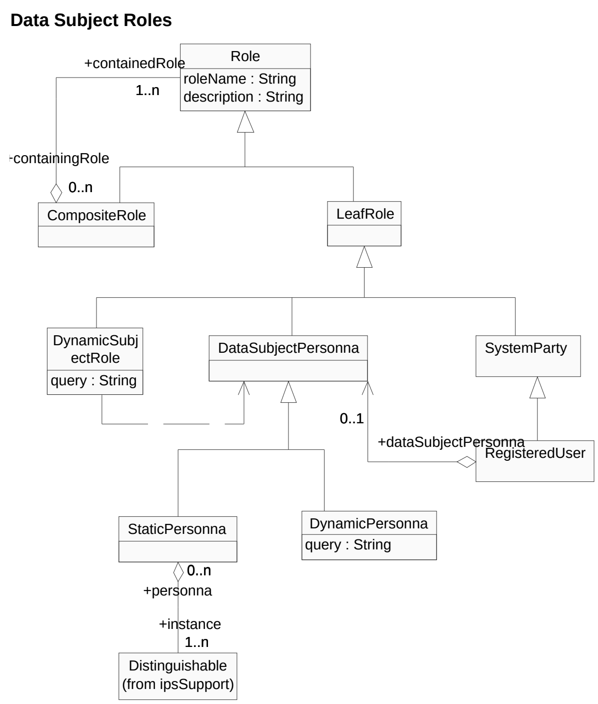
Three LeafRole subtypes are defined in the model. Others could be added, if the evaluation engine is extended to handle these. The SystemParty LeafRole represents registered system users and system groups. The RegisteredUser class represents a single user. A SystemGroup represents a single user group. The DynamicGroup represents the set of registered users selected by executing the specified query.

A fourth support LeafRole does not depend on registered user information. Instead the TokenUser LeafRole assumes that the data user will present a "token" that is a credential issued by the data subject. Token users may be limited in how many times they may use the token to act on the data subject's data, and may also be valid for a specified period of time.

Data subjects

Each privacy rule applies to View data belonging to one or more *data subjects*. Each data subject is represented by a DataSubjectPersonna object. The DataSubjectPersonna class is a subtype of LeafRole. The DataSubjectPersonna specifies a set of PII data items. Generally, a DataSubjectPersonna is associated with a individuals, but organizations could have personnas if their data needs to be protected by policy rules. An individual could have multiple DataSubjectPersonna objects with separate PII data. PII data can also be shared by more than one DataSubjectPersonna. DataSubjectPersonnas can represent anonymous or pseudononymous individuals, as well as identified individuals.

A rule specifies a data subject Role that can be either a CompositeRole or a LeafRole. A CompositeRole would be used if the rule applies to more than one data subject personna. The set of data subject personnas is essentially setting the scope of PII data to which the View applies. Or, conversely the View determines what subset of a data subject's PII data is covered by a rule.



A set of data subject personnas can also be defined using a DynamicDataSubjectRole, instead of a CompositeRole. The DynamicDataSubjectRole supports selecting a set of DataSubjectPersonna objects using the specified query. For example, "all customers with a residence in Canada".

In addition to directly specifying LeafRoles as DataSubjectPersonna or DynamicDataSubjectRole objects, the model also allows reusing SystemParty LeafRoles to define data subjects. In that case, the SystemPart LeafRoles must decompose to a set of RegisteredUsers that each have an association to a DataSubjectPersonna object. Supporting RegisteredUser objects as a way to specify data subject personnas allows the same groups defined for data users to be reused as data subjects. For example, if an enterprise is setting up rules it may want to use its "employee" system group for both the data subject and data user roles of different rules. That is, there may be rules that protect employee privacy differently from customer privacy - in that case the "employee" group would be the data subject role of a rule. In other cases, employees may be allowed to access other employees' business phone numbers. In that case, the "employee" group would be used as the data user role of a rule.