Policy Creation for Enterprise-Level Data Sharing

Briesemeister, L. [1], Gustafson, W. [2], Denker, G. [1], Martin, A. [2], Martiny, K. [1],
Moore, R. [2], Pavlovic, D. [3], & St. John, M. [2]

[1]SRI International, Menlo Park, CA, USA
[2]Pacific Science & Engineering, San Diego, CA, USA
[3]University of Hawaii, Honolulu, HI, USA
MarkSt.John@pacific-science.com

Abstract. Enterprises, including military, law enforcement, medical, financial, and commercial organizations, must often share large quantities of data, some potentially sensitive, with many other enterprises. A key issue, the mechanics of data sharing, involves how to precisely and unambiguously specify which data to share with which partner or group of partners. This issue can be addressed through a system of formal data sharing policy definitions and automated enforcement. Several challenges arise when specifying enterprise-level data sharing policies. A first challenge involves the scale and complexity of data types to be shared. An easily understood method is required to represent and visualize an enterprise's data types and their relationships so that users can quickly, easily, and precisely specify which data types and relationships to share. A second challenge involves the scale and complexity of data sharing partners. Enterprises typically have many partners involved in different projects, and there are often complex hierarchies among groups of partners that must be considered and navigated to specify which partners or groups of partners to include in a data sharing policy. A third challenge is that defining policies formally, given the first two challenges of scale and complexity, requires complex, precise language, but these languages are difficult to use by non-specialists. More useable methods of policy specification are needed. Our approach was to develop a software wizard that walks users through a series of steps for defining a data sharing policy. A combination of innovative and well known methods is used to address these challenges of scale, complexity, and usability.

Key words. Cybersecurity, privacy, user interface design

1 Introduction

1.1 Enterprise Data Sharing

Enterprises are large, distributed, information-rich organizations, including military, law enforcement, medical, financial, and commercial organizations. Enterprises must often share large quantities of data, some potentially sensitive, with many other enterprises. However, data sharing is rarely, if ever, all or none. Rather, specific types of data are shared with specific partners to achieve specific objectives. Data sharing may be on-going to achieve long-term objectives, or it may be a one-time event to achieve a specific transient objective. For example, financial institutions may regularly share data with the government and with other financial institutions, or they may share data about a specific transaction with law enforcement to track an illegal transaction. Similarly, military organizations may regularly share some data with other militaries to coordinate mutual defense, or they may share specific data with a host nation to support a disaster relief operation. Once the relief operation is complete, the data sharing ceases.

Enterprise data sharing, and personal data sharing, as well, involves two central concerns: deciding whether to accept the risks of any particular sharing of data, and how to correctly specify that sharing of data, including the specific partners to receive the data, and any other parameters, such as a specific time period for data sharing to achieve the enterprise's objectives. Another concern is how to incorporate innovative cryptographic methods, such as multi-party computation and differential privacy, to obfuscate the details of shared data while still achieving the benefits of sharing data. For example, using multi-party computation, data could be shared in encrypted form and combined with encrypted data from other parties to compute results relevant to all the parties. Parties only see their own data, but everyone sees the results [1]. The framework described here provides a path forward for incorporating these technologies, and work on this capability is underway.

2 Data Sharing Risks and Benefits

Deciding whether to share specific data with specific partners involves evaluating two competing values: 1) the risk that the partner will exploit the shared data in unintended ways and cause harm, and 2) the benefit of sharing the data with the partner to achieve an objective. In most cases, the benefits should outweigh the risks in order for data sharing to be acceptable. Even if the risks are high, if the benefit is higher still, then sharing should typically proceed.

The risk of sharing data can be broken down into two factors: the sensitivity of the data, that is, the amount of harm its unintended exploitation could create, and the trust in the data sharing partner to not exploit, or further share, the data.

A simple equation for computing risk from sensitivity and trust is based on the decision making and actuarial formula that risk is the product of the chance of an event and the value of that event. Chance is translated to trust – the likelihood of the recipient keeping shared data safe and not exploiting it. Value is translated as sensitivity – the amount of harm that would result from exploitation. These concepts are captured in formula (1), where (1 – trust) is the likelihood of exploitation.

$$\text{Risk} = (1 - \text{trust}) * \text{sensitivity} \qquad (1)$$

Trust in partners, data sensitivity, and data sharing benefits are all subjective assessments. A consensus approach to assessing trust is to assess the ability, integrity, and beneficence of the data sharing partner [5]. Ability involves the technical abilities of the partner to keep the shared data secret. Integrity involves the degree to which the partner abides by law and order and upholds agreements. Beneficence involves the relationship between the enterprise and the partner, and the extent to which the partner is invested in maintaining that relationship by upholding the agreement not to exploit the data. These assessments are then combined into an overall assessment of trust in the partner.

A standard approach to assessing sensitivity is to assess the amount of harm that could be caused if the data were exploited. This harm can be estimated in terms of money, personnel, reputation, equipment lost, and the value of missed or thwarted objectives.

Data sharing benefits can be assessed as a combination of the value of the objective for which the data is being shared and the increased probability of success of the objective given that the data is shared. If sharing the data greatly increases the chances of success, and the objective is valuable, then there is high benefit to the sharing.

Developing reliable and valid methods for assessing trust, sensitivity, and benefits are important challenges. Here, though, we focus on the mechanics of data sharing and several technical challenges to specifying data sharing correctly and effectively.

3 Data Sharing Policy Specification Challenges and Solutions

The mechanics of data sharing can be addressed through a system of formal data sharing policy definitions and automated enforcement. The authors and others have collaborated to develop an experimental enterprise-level system for sharing data in effective, privacy-preserving ways, included tailored data access control based on explicit data sharing polices [6, 8]. In this system, data sharing partners make data requests. Incoming data requests are evaluated at a policy decision point. A policy decision engine assesses the data request against the data sharing policies in force at that time and returns a decision to allow or deny the request. Additionally, policies can be written to include constraints on data sharing, such as only share police record data for residents who are at least 18 years old, or only share a count of residents testing positive for tuberculosis rather than the names of each individual who has tested positive. The sharing decision, to allow or deny the request, and any constraints on sharing are forwarded to a policy enforcement point that creates policy-safe queries to retrieve the requested data from a database. This separation into a Policy Decision Point (PDP) and a Policy Enforcement Point (PEP) follows standard approaches (cf. for example XACML [12]) and ensures that policy decisions can be made without the need to access data---only the policy enforcement point requires accessing data instances.

A first challenge to precisely specifying data sharing policies involves the scale and complexity of data types. Enterprises typically have many different types of data, and these data types are not well represented by a flat list of types that users could scan through and check off to be shared. Further, data types cannot be organized naturally into a hierarchy. Rather, data types bear complex relationships with one another.

Since data types are typically not organized into a hierarchy, there is no root that could be used to start the exploration of data types. Instead, data types can be organized from many perspectives, such as types of entities (people, ships, money, resources) or domains (financial, personal, military). Second, there are loops in the relations among data types such that a data type would appear in multiple locations in a hierarchy (for example, persons are citizens of nations and nations have places of interest to which persons travel, meaning that both persons and nations would appear in multiple locations in a hierarchy). Instead, the relations among data types is better represented as a connected graph or network. A method is therefore needed to visualize a network of data types so that users can navigate among the data types and specify which types to add to a data sharing policy.

A second challenge involves the scale and complexity of data sharing partners. Enterprises typically have many partners involved in different projects, and there are often complex hierarchies among groups of partners that must be considered and navigated to specify which partners or groups of partners to include in a data sharing policy. For example, an enterprise may wish to share data all nongovernmental organizations (NGOs), only with health care NGOs, or only with Doctors without Borders and the Red Cross. Logical expressions, including "AND," "OR," and "NOT" provide a precise way to specify arbitrary sets of partners, but they can be difficult to use. More intuitive methods are needed.

A third challenge is that defining policies formally, given the first two challenges of scale and complexity, requires complex, precise language. This complexity and precision can be achieved in formal logic languages, such as Flora-2, but they are difficult to use by non-specialists, such as members of enterprises who need to define data sharing policies to accomplish their objectives.

Here, we describe three methods for addressing these three challenges. These methods involve 1) representing an enterprise's data as a semantic network and developing a tool that visualizes the network and allows users to navigate the network and designate which data types to share, 2) an attribute-based filter metaphor for specifying sets of partners with whom to share specific data, and 3) a software wizard that walks users through the steps of specifying a data sharing policy, including which data to share with whom, without requiring users to learn complex, formal languages for policy specification.

3.1 Specifying Data Sets

Throughout the system, including the Policy Decision Point and Policy Enforment Point, the data available for sharing is represented by a common data model (CDM). The CDM specificies the relationships among data classes, their subclasses, and their properties and is represented in Web Ontology Language (OWL). A CDM visualization was developed for the project, called the CDM Explorer, that reads in the CDM specification, and visualizes it as a network to users. Users can navigate around the network to select data classes and properties to add to a data sharing policy. While there are many network visualization tools, the CDM Explorer has several specific capabilities including visualization and navigation of a semantic network, specifically, ability to select data types and properties for inclusion in a data sharing policy, and ability to specify constraints on which data are shared.

To begin exploring the network, the user is first presented with a list of data classes. To keep the list to a manageable length, subclasses are hidden under their superclasses until the superclass is expanded (see **Figure 4**). The figure shows that the superclass Physical Entity has been expanded to show two subclasses, Mobile Entity and Stationary Entity. Mobile Entity has also been expanded. Currently, in the CDM, the only subclass of Mobile Entity is Ship.

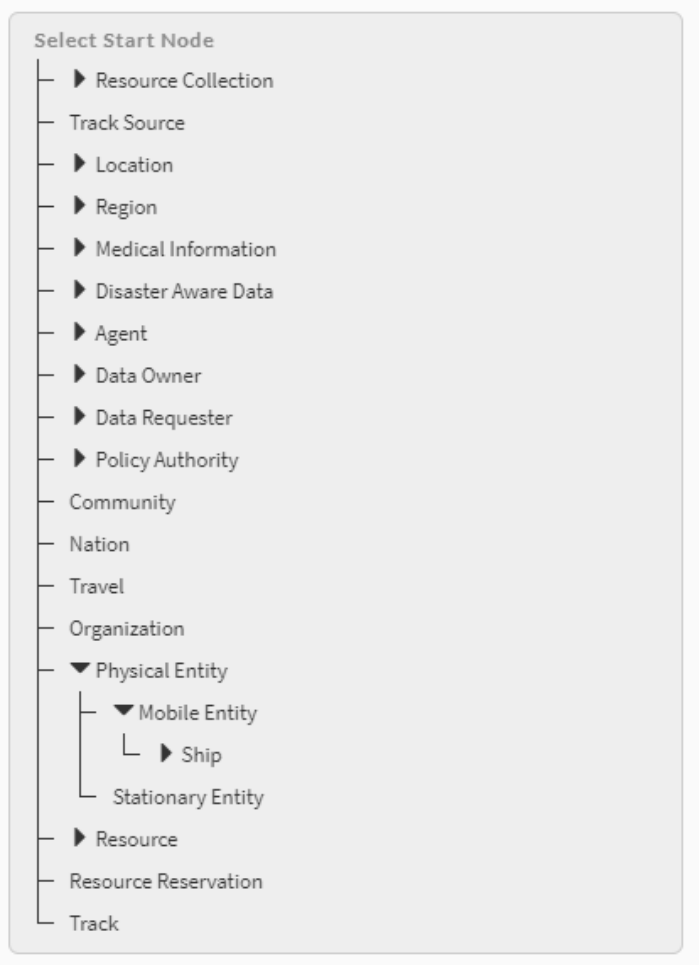


Figure 1. The entry view of the CDM Explorer showing a list of data classes arranged into superclasses and their subclasses.

If the user chooses Mobile Entity, the CDM Explorer opens a view of the CDM network (see **Figure 4**). The Mobile Entity class is shown on the left in blue, indicating that it is a selected data class. Mobile Entity is linked to four other data classes: Nation, Organization, Resource, and Track. These linked data classes are called object properties of Mobile Entity. Rolling over an object property, such as Nation, reveals the relationship between the two data classes. In this case, the relationship between Mobile Entity and Nation is "Owner Nation." This link specifies the nation, if any, that owns each mobile entity. Similarly, Organization specifies the organization, if any, that owns each mobile entity.

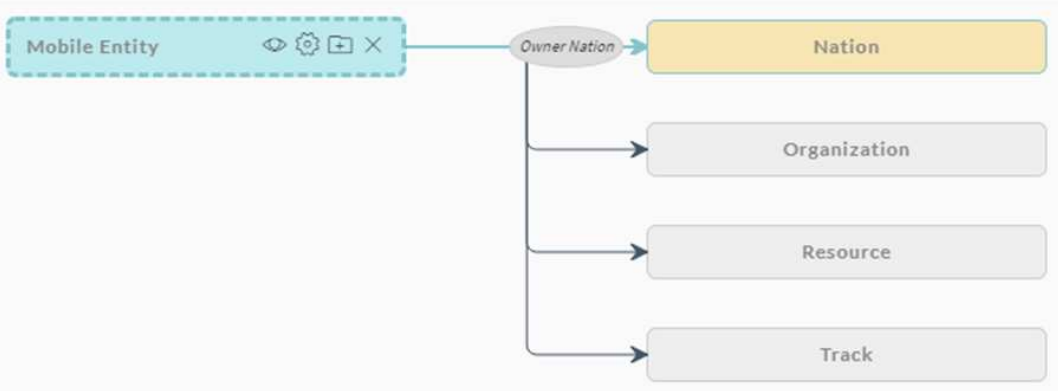


Figure 2. The view of the CDM after the user chooses Mobile Entity as the starting data class.

Another linked class is Track, which describes the location, course, speed, and a timestamp for any mobile entity. It is used both to locate the current location and course of an entity and a history of prior locations, courses, and speeds. If the user selects Track, then the Track data class turns blue, as well, and the CDM Explorer shows the object properties linked to Track: Track Source, Location, and Mobile Entity (see **Figure 3**). Note that Mobile Entity appears as a link on both sides of Track. In most cases, the user will not want to select the link to return to the prior class. If they do select that link, a warning dialog about creating a tight loop will appear.

As shown in **Figure 3**, Track has three data properties shown inside the Track box: Course, Speed, and Time. Track also has three object properties shown as links on the right side of the figure. Track is also selected for focus in the CDM Explorer, as indicated by the dashed lines around the box.

The location data for a track is specified through the link to the object property Location. The rationale for this CDM layout choice is that locations are in themselves complex data types. For example, locations can be expressed in different coordinate systems and units. Hence, we have added a hierarchy for these data types into our model. The examples shown here express geographic locations in latitude and longitude via a Location subclass called Geodetic2DLocation.

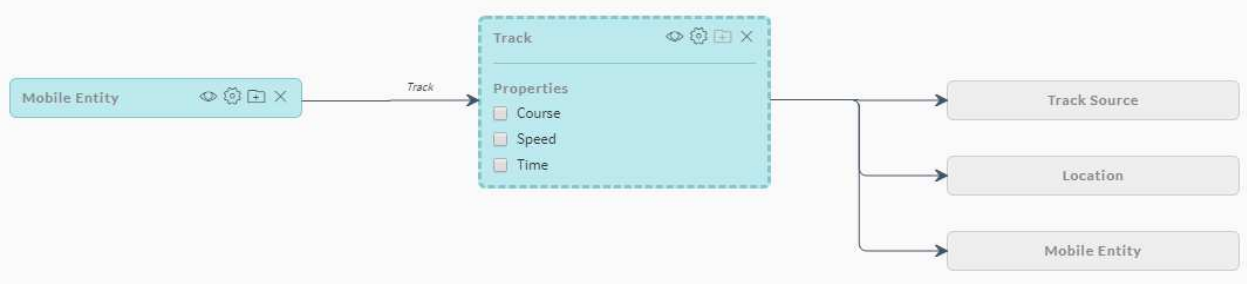


Figure 3. View of the CDM Explorer after Track is selected.

Users can back up and remove classes by selecting the X in a data class box. Users can also select subclasses by selecting the folder icon next to the X in a data class box. In **Figure 4**, the user has selected the Ship subclass of Mobile Entity, and then selected the folder icon again to show subclasses of Ship.

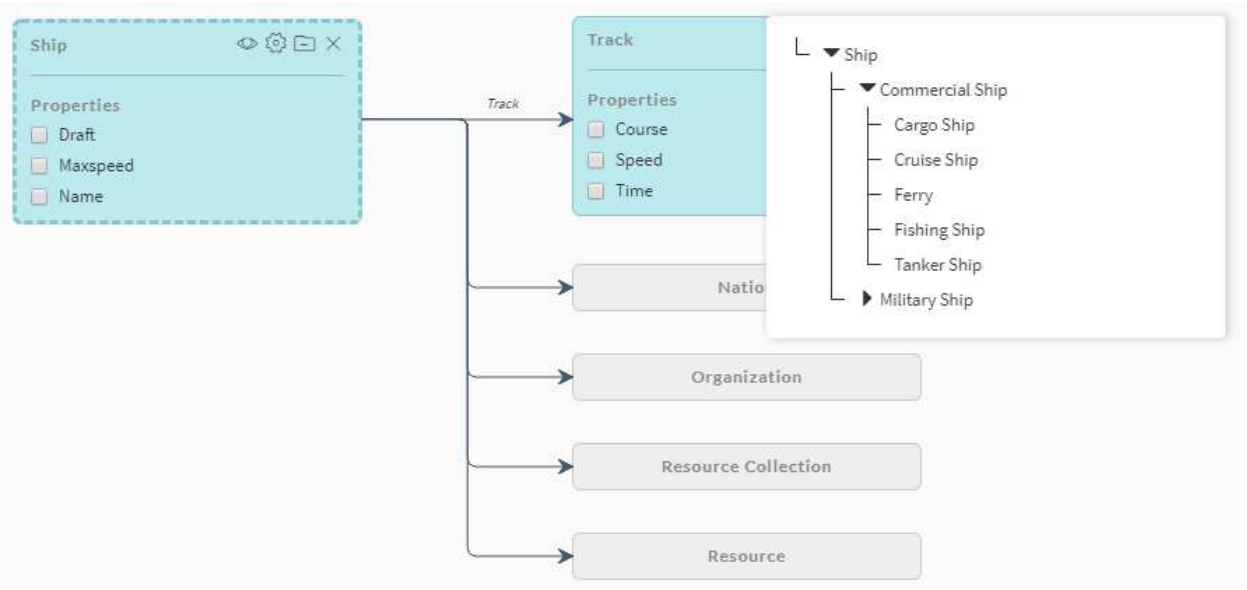


Figure 4. A snapshot from the CDM Explorer tool showing the Ship data class, it's three data properties (blue box on the left), multiple subclasses of Ship (white box on the right), and five links to other data classes (blue box for Track and grey links in the middle).

Because Ship is a subclass of Mobile Entity, it inherits each of the Mobile Entity object properties: Nation, Organization, Resource, and Track. If Mobile Entity had any data properties, Ship would have inherited those

properties, as well. Similarly, if the user selected the subclass Cargo Ship, then Cargo ship would inherit all five object properties and three data properties of Ship. Cargo Ship might also have its own object and data properties, such as Cargo Tunnage to add.

3.1.1 Joint data sets and the CDM. An important realization from the project was that data is almost always shared in joint sets, such as ship names and locations during a military or law enforcement operation or patient names and disease states during a humanitarian crisis. Locations alone have little or no meaning, as do ship names alone. It is only the combination of ship name and location that reveals information and may be sensitive. In other words, ship names alone cannot be exploited; nor can locations; but the combination of ship names and locations can be exploited and therefore are sensitive. For example the names of ballistic missile submarines and their locations is extremely sensitive. Furthermore, the data need to be correlated in the sense that ship names are correlated with their locations rather than simply providing two uncorrelated lists of names and locations. A joint data set specifies this correlated sharing of data.

While joint data sets is the correct level of analysis for assessing data sensitivity, assessing sensitivity is nontrivial. For example, while the names and locations of ballistic missile submarines is extremely sensitive, the names and locations of commercial fishing vessels is not typically very sensitive. One must think carefully about the instances of joint data within a joint data set and whether there are any sensitive instances. Constraints on a joint data set (see section 3.1.2 below) can significantly change the sensitivity of shared data depending of whether they allow sensitive instances to be shared or constrain them from sharing.

Finally, sensitive data can be inferred from shared data. An example is re-identifying individuals from personal data combined with other publically available information. A famous example of this type of inference was the re-identification of the governor of Massachusetts from anonymous medical records combined with public voter rolls [9]. In fact, the combination of data of birth, sex, and zip code can be used to identify 87% of individuals in the U.S. [9]. In some zip codes, the percentage can be even higher. Due to this inference potential, a joint data set's sensitivity should also be assessed with regard to what sensitive inferences might be drawn from the shared data.

Figure 5 shows the result of selecting Ship and Track, and then selecting Location and then the subclass of Location called Geodetic2DLocation. Additionally, the user has selected the Name of Ship and the Latitude and Longitude of Geodetic2DLocation. This specification creates a joint data set for sharing ship names along with their locations.

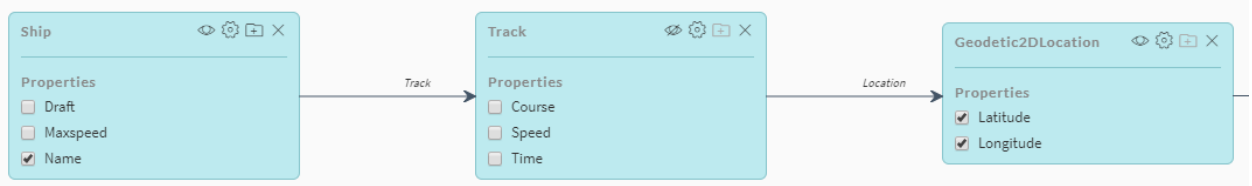


Figure 5. Selection of the data classes Ship, Track, and Geodetic2DLocation and the data properties (Ship) Name, Latitude, and Longitude.

Further, it is not sufficient to specify *that* different data items are correlated. It is also necessary to specify *how* these items are connected. For instance, if the names of nations are to be shared jointly with the names of persons, it is ambiguious how persons are to be related to nations (i.e., on what attribute persons and nations ought to be joined). For instance, among other relations, one can chose to share information about a nation's residents, or its citizens. In both cases the result is a join between the classes Nation and Person, but the affected data instances will usually not be identical sets because nations typically have residents who are citizens of a foreign country, and simultaneous have citizens who are residents in a foreign country. These considerations imply that the set of a nation's citizens and a set of a nation's residents would be considerably different from each other. Thus, to prevent any ambiguities, a joint data set needs to precisely specify how classes are linked by including the linking property (shown as the edge labels in the figures), such as "resident of" or "citizen of", in its specification. Consequently, a joint data set specification must always be expressed as a connected subgraph of the CDM.

3.1.2 Constraints. Constraints on data sharing can also be specified within the CDM Explorer. For example, a user could desire to share the ship name and location of just a single ship of interest, for instance, if it were suspected of some illegal activity. In this example, a constraint would be placed on the Name data property in the Ship data class. **Figure 6** shows how this constraint is specified. From the Ship data class, the user chooses the gear icon in the upper right corner of the data class box. This action opens a dialog box where the user chooses which property to constrain (Name), then specifies the nature of the constraint as the logical test Equal, and then defines a String with

the value of the name of the ship in question. For example, a user could specify that the ship's name equal "Damanzaihao," the name of the world's largest fishing vessel, which was recently detained by Peru for illegal fishing [7]. This approach to specifying constraints is necessary due to the separation of the architecture into decision and enforcement points. The policy decision point does not have access to any actual data, and thus it is not possible to specify policies directly on specific data instances (such as "location of Damanzaihao"). Instead, the policies need to be specified at the level of the data model (i.e., "locations of ships") and then the additional constraint on the ship's name allows the policy enforcement point to restrict shared data to specific instances, as intended.

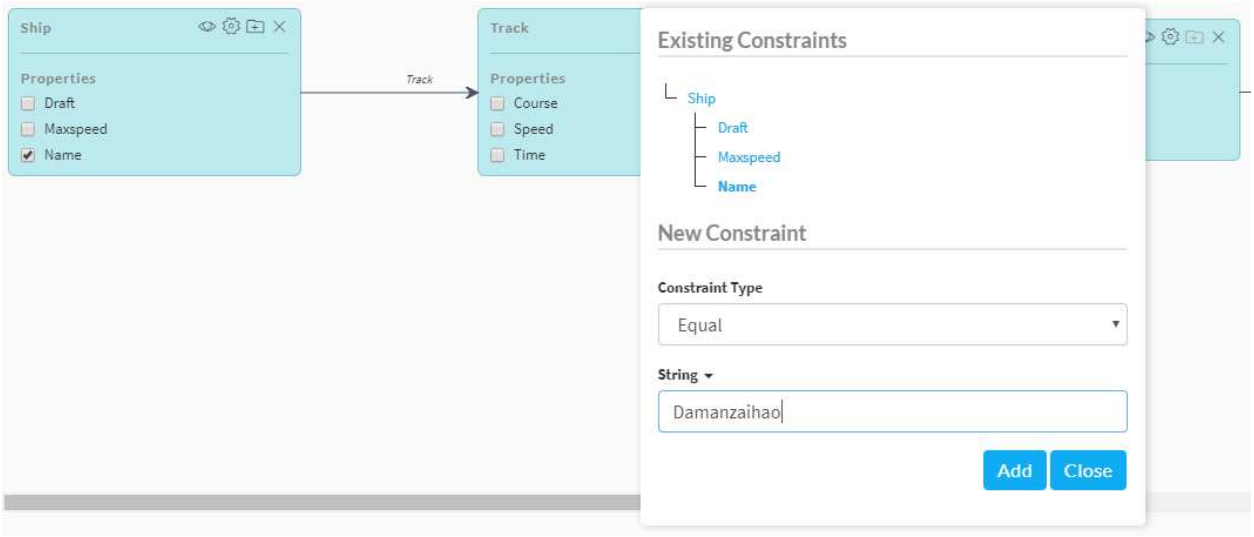


Figure 6. Adding a constraint on the name of a ship.

In the ship name example, Ship Name is a shared property and part of the joint data set. However, the constraint does not need to be a shared property. Any property can be used as a constraint. For example, a user may have a goal to share persons' names and arrest records but to exclude the arrest records of children. This constraint would involve placing a constraint on the DateOfBirth property in the data class Person. In this case, the constraint is placed on a property that is not shared as part of the policy. That is, if the policy is to share names and arrest records, but no other personal information, then DateOfBirth will not be shared. It can still be a source of constraint on shared data, however. Similarly, the sharing of the names of ships and their locations could be limited to a geographical region of interest or to a particular timeframe, such as immediately preceeding and following some event of interest.

3.1.3 Summary for Specifying Data Sets. As the CDM grows to include more data classes and properties, the CDM Explorer becomes a critical tool to visualize sections of the network and specify parts of it for sharing in a policy. Even so, familiarity with the CDM and where data classes are located is very helpful. Within a large CDM, it becomes possible to lose track of one's location and begin having difficulty navigating. For this reason, future versions of the CDM Explorer will contain a search tool where a user can specify a data class or property and have that term highlighted in the network. It may also be necessary to highlight related terms since terminology in very large CDMs could become complex. The user could then choose which highlighted instance of the term to explore further. It is also possible to specify two data classes, and have the CDM Explorer find the shortest path between them. This path is likely to be the desired connection path between the classes, and if not, it provides a useful basis for exploring for the desired path. Finally, a zoom capability should also help with navigation issues.

In sum, the CDM Explorer provides a rich and useful method for visualizating the relations among data within the CDM, specifying joint data sets to share, and specifying any desired constraints to place on that sharing.

3.2 Specifying Data Sharing Partners / Data Requesters

An enterprise is likely to have many data sharing partners, hereafter called data requesters because they make requests for data that are then checked against data sharing policies. Specifying one or more data requesters for a data sharing policy can become unwieldy. For instance, if each data requester were simply listed and then checked off for inclusion in a policy, that list would potentially be very long, and checking requesters would be prone to errors.

Instead, policies are likely to refer to groups of data requesters who perform similar tasks, therefore, choosing requesters based on their attributes could substantially simplify specifying sets of data requesters for a data sharing policy. Useful attributes include the user's role or tasking, nationality, and organization.

However, using logical expressions, such as "[attribute-1 AND (attribute-2 OR attribute-3)] but NOT attribute-4" to write attribute-based specifications is likely to exceed most users' training. The general public is well known to have difficulty both reading and writing complex logical expressions including the correct use of "AND," "OR," and "NOT" [2-4]. Instead, the user interface needs to use a more readily usable metaphor to specify attribute-based specifications of data requesters. We chose a fairly common method of specifying search filters that is more natural and well known for most users. As described in more detail below, users select one or more attributes to filter the set of data requesters. They can also select among members of an attribute, if specific members need to be included or excluding from a policy. For example, a user could select epidemiologists as the role and all nations as the nation. Or a user could select specific nations or even exclude specific nations from a data sharing policy. This method provides similar expressability as formal logical expressions, but within a more intuitive metaphor for users.

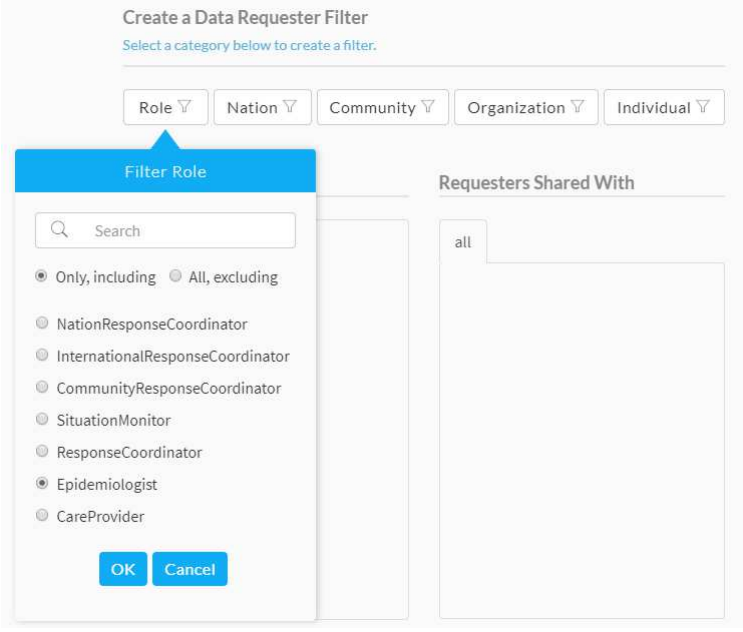
Finally, membership in user groups changes over time, yet policies should still apply to the current membership of a group. For example, if the Centers for Disease Control bring on a new epidemiologist, any policy that shared data with CDC epidemiologists should share data with that new epidemiologist without requiring a change to the policy. This requirement is different from standard uses of search filters. Search filters are typically used to identify an enumerated set of things. Here, the set of search filters itself becomes the data requester specification within a policy, and the enumerated list of things it identifies may change over time as group memberships change.

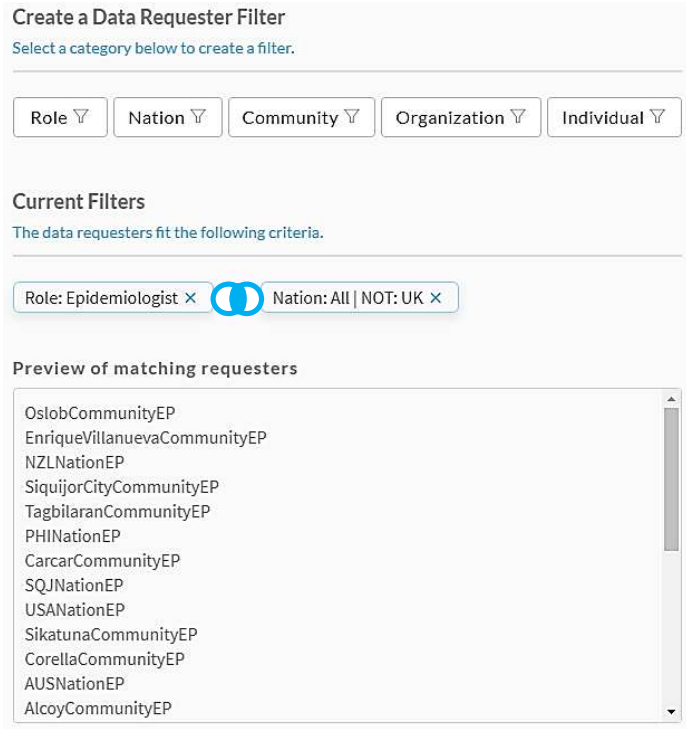
In sum, there are four requirements that data requester specification must address

- Many data requesters in the system
- Attribute-based specification
- Alternative user interface or metaphor to formal logical expressions
- Dynamic specifications as members of user groups change over time

3.2.1 Details of Search Filters. **Figure 7** shows the user interface developed for specifying data sharing partners via attribute-based filters. The attributes, such as role, nation, and organization, are shown in the top row. Users can open an attribute and select all or some of the choices. For example, under "role," there are choices for care providers, response coordinators, situation monitors, epidemiologists, and other more specific roles. These roles were developed for a use-case involving humanitarian assistance, specifically a pandemic. A more generic interface would have many additional roles and include the ability to search for relevant roles.

a)





b)

Figure 7. Data requester search filters. The top figure (a) shows the selection of Epidemiologists from the Role attributed-based filter. The bottom figure (b) shows the creation of two filters, one for Epidemiologists and a second for all Nations except UK. The lower half of the bottom figure shows the list of data requesters who match the two filters.

The currently specified set of filters is shown in the center of the display. Epidemiologist was chosen as the role, and all nations, excluding the United Kingdom, was chosen as the nation. Each filter is represented as a "pill," and the two filters are combined as an intersection. The explicit use of logical operators "AND" and "OR" was avoided, and icons for intersection and union were used in their stead. While these icons are unambiguous, they are more obscure, therefore, a simple explanation and example was provided when a user rolled over them.

The set of data requesters to which the filters resolve is shown at the bottom of the display. This list is provided to give users feedback on the effects of the filters they specify and combine. As filters are specified and combined, the list grows and shinks accordingly. Importantly, however, it is the set of filters and their combination that is provided to the policy rather the enumberated list of partners. By providing the filters, the list is evaluated fresh each time the policy is assessed. As the membership of Epidemiologist changes over time, for example, the list will change as well. This design ensures that a policy consistently refers to the current membership of attributes rather than the membership in effect when the policy was created.

Attribute-based filtering is a fairly common user interface method for narrowing a search. **Figure 8** shows an example of filters taken from the website Amazon.com. In the example, attribute categories are shown in bold with their choices shown underneath. Checking multiple boxes within a category is treated as a union, while checking boxes between categories are treated as an intersection, e.g. checking Baldwin and 5 foot leads to just those items that fit both attributes. Selecting a link, an attribute in a category that does not have a checkbox, is treated an a intersection, e.g. under Doorknob function, one choses either privacy or passage. The revised list of products that meet the filter specification would be shown to the right of the figure.

Young and Shneiderman [10] developed an attributed-base filter method that used the filtering of flowing water as a metaphor. Users could create the equivalent of logical queries by selecting attributes and values of attributes, and dragging the attributes to locations on the screen to create a sequence of filters. Their method allowed arbitrary nesting while our simpler method does not.

Figure 8. Example filters from Amazon.com.

3.2.2 Summary for Specifying Data Sharing Partners. This method of attribute-based filters avoids formal logical expressions and instead uses a version of a common method for filtering search results. It scales far better than an enumerated list of data sharing partners and allows a clear way to specify meaningful groups of partners.

3.3 Defining Policies

Once the to-be-shared data and the data sharing partners have been specified, there is still the matter of writing the policy so that it can be evaluated by the policy decision engine when requests for data arrive. The engine requires a formal, precise representation of the policy, and there are formal languages designed for this type of specification. Flora-2 (http://flora.sourceforge.net/) is a good example.

However, Flora-2 is a complex language, and as with any formal language, it has a complex and detailed syntax. Only a highly trained user would be able to code a data sharing policy directly into Flora-2 without error, and it would still be a time-consuming process. For example, a host country might decide to share disease status data about its residents with international response coordinators. However, the host country may want to protect its residents' privacy by only providing aggregate counts of how many of its residents have the disease. For this example, the host nation is a fictious county named Bohol, and the Bohol nation policy authority is the individual who has authority to create data sharing policies concerning the Bohol nation's data. In English, the policy could be stated as "Bohol nation policy authority allows sharing statistically accurate aggregations at the level of nation of disease state data of its residents with any Response Coordinator." Figure 9 shows how this policy could be written in Flora-2. Any typo could break all policy specifications in the system.

```
@!{BoholNationAllowsNationAggregationToRC}
?pa [ po#allow_sa(?req, ?dd) ] :-
?pa = combined#BHLNationPA,
?req [ po#requester -> ?r ],
((?r = po#isDR_type.po#getDR(prime#ResponseCoordinator))),
?polData = ${
?var4 : prime#Person [
prime#medicalInformation -> ?var5 ],
?var5 : medical#DiseaseStatus [
medical#state -> ?var6 ],
?var3 : prime#Community [
prime#resident -> ?var4 ],
?var0 : prime#NationPolicyAuthority [
prime#nation -> ?var1 ],
?var1 : prime#Nation [
prime#name -> ?var2,
prime#community -> ?var3 ]
},
implies_sharing(?polData, ?req.po#requestFormula, ?constr),
?id = "BoholNationAllowsNationAggregationToRC"^^\string,
?descr = "Bohol nation PA allows sharing statistically accurate
aggregation of disease state data of its residents at the level of
nation with any Response Coordinator"^^\string,
?prio = 0,
?dd = decisionDetails(?pa, ?req, ?constr, ?startTime,
?endTime, ?id, ?descr, ?prio).
```

Figure 9. An example of a data sharing policy written in Flora-2.

The alternative method developed by our project was to create a software wizard that walks users through a process of specifying the components of a data sharing policy, including specifying the joint data set and the data requesters. The user interface of the wizard is designed to support a user who is not trained in logical languages to accurately express a desired data sharing policy. The wizard then turns the policy specification into Flora-2 code for evaluation.

The components of a data sharing policy are:

1. Policy Authority – person who had authority to create a policy and created this policy (*individual)*
2. Decision – permissive or restrictive policy *('allow sharing' or 'disallow sharing')*
3. ID – a unique identifier for each policy *(string)*
4. Name – a short natural language name for the policy *(string)*
5. Description – a natural language description of the policy *(string)*
6. Requesters – which data sharing partners are addressed by this policy *(Disjunctive Normal Form (DNF) of RequesterAttributes)*
7. Data – which data are addressed by this policy *(joint data set)*
8. (optional) Constraints of a sharing decision *(Special Constraint Vocabulary)*
9. (optional) Start and end time for this policy *(DateTime)*
10. Precedence – A number that determines which other policies this policy can override and which policies can override it *(integer)*

Step 1 of the wizard is determined by credentials established while logging onto the system. A policy authority has the ability to set data sharing policies only over a specified set of data that is managed through their credentials. Steps 5, 6, and 7 of the wizard are described above. The remaining steps of the wizard are straight-forward dialogs.

In sum, a software wizard including the CDM Explorer and the attribute-based data requester specifier allows a non-specialist to create detailed and unambiguous data sharing policies.

4 Summary and Conclusions

Enterprises need precise and sophisticated methods for specifying which data to share with which of their partners. A useful approach is to write data sharing policies that explicitly and precisely define data sharing. Three challenges to this approach are 1) the scale and complexity of specifying the data, 2) the scale and complexity of specifying the data sharing partners, and 3) users' limited training in using formal languages to write data sharing policies.

Here, we describe three methods for addressing these three challenges. The scale and complexity of specifying the to-be-shared data is addressed by representing data types owned by an enterprise within a network, or graph, formalism called a common data model (CDM). A CDM Explorer tool visualizes the network and allows users to explore the network and designate a connected subgraph of data types to share. The CDM formalism suits the nature of the relations among data types, and the CDM Explorer provides a graphic way to visualize the network and specify data types to share. The CDM Explore also provides a method for specifying constraints on which data to share.

The scale and complexity of specifying data sharing partners, also called data requesters, is addressed by developing an attribute-based filter metaphor for specifying sets of data requesters. While writing logical expressions to specify sets of data requesters would be difficult for many users, attribute-based filters are a common metaphor for searching and filtering items, for example, on Amazon.com. Unlike most such filter tools, however, the filters themselves are written into the data sharing policy rather than the resulting enumerated list of items. Using the filters makes the policy dynamic and sensitive to changes in group membership over time so that a policy will resolve to the correct set of data requesters over time.

Users' limited training in formal languages is addressed by developing a software wizard that incorporates the CDM Explorer tool and the attributed-based filter tool and walks users through the steps of specifying a data sharing policy. The resulting specification is then rewritten into a formal language for evaluation by the policy decision engine.

The structure of the CDM and its visualization within the CDM Explorer and the attribute-based data requester filters scale better than enumerated lists of data types and requesters, and they are much easier to use than formal languages

and logical expressions. Together, these tools create an innovative and highly useful system for specifying data sharing policies for enterprises. Given the prevalence of data sharing among enterprises and the risks of inadvertent sharing and exploitation, effective and usable tools to support precise data sharing are essential.

The approach should generalize well. Any types of data and relationships among data types can be represented in a CDM and then visualized in the CDM Explorer. Additionally, there are no in-principle limits to the types of constraints that could be imposed on which data to share, though new constraint types would have to be added to the constraint dialog. The approach is also general in terms of data requesters, groups of requesters, and attributes, though new attributes would have to be added to the dialog, or a method would have to be developed to automatically add attributes from a description. Consequently, the approach should be useful for a wide variety of different types of enterprises and data sharing purposes.

5 Acknowledgements

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. Approved for Public Release, Distribution Unlimited.

6 References

1. Du, W., & Atallah, M. J. (2001, September). Secure multi-party computation problems and their applications: a review and open problems. In *Proceedings of the 2001 workshop on New security paradigms* (pp. 13-22). ACM.
2. Essens, P. J., McCann, C. A., & Hartevelt, M. A. (1991). An experimental study of the interpretation of logical operators in database querying. *Acta psychologica*, *78*(1-3), 201-225.
3. Greene, S. L., Devlin, S. J., Cannata, P. E., & Gomez, L. M. (1990). No IFs, ANDs, or ORs: A study of database querying. *International Journal of Man-Machine Studies*, *32*(3), 303-326.
4. Ogden, W., & Kaplan, C. (1986, September). The use of AND and OR in a natural language computer interface. In *Proceedings of the Human Factors Society Annual Meeting*(Vol. 30, No. 8, pp. 829-833). Sage CA: Los Angeles, CA: SAGE Publications.
5. Mayer, R. C., Davis, J. H., & Schoorman, F. D. (1995). An integrative model of organizational trust. *Academy of management review*, *20*(3), 709-734.
6. Myers, K., Ellis, T., Lepoint, T., Moore, R. A., Archer, D., Denker, G., Lu, S, Magill, S. & Ostrovsky, R. (2017). Privacy technologies for controlled information sharing in coalition operations. *Proceedings of the Symposium on Knowledge System for Coalition Operations*. Los Angeles, CA.
7. Seafoodsource (July 9, 2018). Found at www.seafoodsource.com.
8. St. John, M., Moore, R., Martin, A., Gustafson, W., Jaramillo, M., Denker, G., Martiny, K., and Briesemeister, L. (July 2018). Enterprise-level private data sharing: Framework and user interface concepts. In *proceedings of the 2018 Applied Human Factors and Ergonomics conference*. Orlando, FL: AHFE.
9. Sweeney, L. (1997). Weaving technology and policy together to maintain confidentiality. *The Journal of Law, Medicine & Ethics*, *25*(2-3), 98-110.
10. Young, D., & Shneiderman, B. (1993). A graphical filter/flow representation of Boolean queries: a prototype implementation and evaluation. *Journal of the American Society for Information Science*, *44*(6), 327-339.
11. OASIS Standard, “eXtensible Access Control Markup Language (XACML) Version 3.0,” January 2013, https://www.oasis-open.org/committees/xacml