Explicit Federal Relationship Management to Support Semantic Integration

Rob Brennan, Kevin Feeney, Brian Walsh, Hendrik Thomas and Declan O'Sullivan
FAME and Knowledge and Data Engineering Group, School of Computer Science and Statistics
Trinity College Dublin
Dublin, Ireland
{rob.brennan, kevin.feeney, walsheb, thomash, declan.osullivan}@scss.tcd.ied

***Abstract*—Integration of disparate information resources has long been a research topic. Semantic approaches can help significantly by allowing expression of concepts divorced from syntax and allowing rich, structured meta-data to be published in a form that is amenable to machine processing, reasoning and inter-domain concept mapping. However the creation of mappings, especially in a dynamic federation of autonomous entities is time-consuming and vulnerable to brittleness and high maintenance costs due to change at many levels in the system. In this paper we propose an approach to managing change and maximizing mapping reuse by building explicit models of the federal relationship context of mapping deployment. These descriptions enable automated support for mapping reuse suggestions and ease discovery of relevant mappings due to changes at the federation, peer domain, shared capability or local model levels.**

Keywords-Federation; Semantic Interoperability; Management

I. INTRODUCTION

Consuming data from multiple, heterogeneous sources and mapping it to a common representation or model is a common requirement of enterprise IT systems. For example, production scheduling and planning systems often need to import data from a wide-range of diverse suppliers' systems and map it to a common internal schema in order to enable consistent and coherent enterprise-wide views. Semantic descriptions, for example using OWL-RDF vocabularies or ontologies, can help significantly by allowing rich, structured meta-data to be expressed in a standardised syntax. However, even assuming the existence of full semantic descriptions, a considerable number of challenges remain in order to achieve real semantic inter-operability. In particular, the semantics of imported data must be mapped to the appropriate internal schema and these mappings must be maintained over time.

Research in semantic interoperability has tended to focus on the expression of semantic mappings and has tended to ignore the problem of dealing with the dynamism of both the data and the schemata that is characteristic of real-world integration problems [1] and the variability of adherence to publication guidelines by different sources, even where they are formally specified. The "semantic drift" of real world systems introduces new requirements for efficient consumption of the distributed data even when centralized, local models are used to mediate consistent data integration for local use. Once multiple autonomous sources are being considered, individual publication characteristics such as schema versioning or implementation quirks increase variability further. For systems that must deal with sustained consumption of information over time, even from a single publisher, semantic divergences tend to increase. Even if idealized mappings that changed appropriately over time were developed, the problem of adequately scoping their applicability in the presence of the diversity of publication behaviors remains, even when applied to common ontologies. For example, if we are importing datasets from two domains that use the same schema, but one of them changes their usage of a particular field, we need to be able to apply this change not to the imported schema, but to the particular instantiation of the schema in that domain.

The multi-factorial nature of dataset dynamics can create an explosion in the number of mappings to be deployed and managed, if the variable scoping of changes are not adequately captured by the mapping model. Yet current research focuses almost exclusively on the generation of mappings. New approaches are required to control change and keep the system up to date and working effectively in the presence of this variability. The practical experience of the database community shows that the creation [2, 3, 4, 5, 6] of transformation code or mappings is a complex and time-consuming task. Thus the description, discovery, tracking and most of all the re-use of existing mappings are of critical importance for integration or consumption of distributed data-sets [7]. Supporting faster and more frequent reuse of mappings could result in cost savings, enable runtime integration, leverage network effects and ultimately produce more flexible and effective systems in a world-wide web of structured data/knowledge [8]. One proven approach that supports maximizing mappings reuse, again from the large-scale database community [7], is to maintain suitable meta-data about the mappings themselves. Given the problem of managing change in a known domain of interest, rather than green-field integration of disparate information sources, it is important to note that there are enormous commonalities across the mapping sets required and hence reuse and appropriate refactoring of mappings can bring great benefits.

What are the appropriate sources of mapping meta-data in such a system? An analysis of the lifecycle of a specific ontology mapping [5] helps a user to understand the decisions and constraints involved in mapping creation or deployment, e.g. is the method used to select mapping candidates

This research is supported by the Science Foundation Ireland (Grant 08/SRC/I1403) as part of the Federated, Autonomic End to End Communications Services (FAME) Strategic Research Cluster (www.fame.ie).

appropriate for the reuse scenario? [9]. Such meta-data can facilitate the discovery and thus reuse of ontology mappings [2, 3] Support for automated reasoning about mappings, e.g. mapping classification, can also help with tool support for reuse decisions. Capturing mapping deployment context, an area largely ignored by current mapping meta-data, is very likely to have a big impact on realistic reuse decisions and could aid precise mapping discovery, thus increasing the chance of reuse. An important source of such context is the federal or inter-domain relationship that underlies the data consumption. Are there a set of pre-existing mappings deployed in that multi-domain federation? Have we relevant past experience of consuming data from that specific federation member entity? Does the specific information resource or service being consumed have known mappings, behaviours or constraints associated with it? How are specific mappings deployed as executable code, gateways or logical statements or rules and can these be re-used at a level of granularity that is finer than that of an individual concept or property correspondence expression? These questions are more specific and hence more revealing than the usual concentration on vocabularies, ontologies, datasets [10], mapping domain background knowledge or even schema-level co-reference identification [11].

Consideration of the preceding issues leads to the research questions: Is it possible to enable consumption of data from multiple heterogeneous, noisy sources in a way that tends to minimize re-integration effort over time? Can models of mapping context that include meta-data on federal relationships assist in this process? What partitioning or structure of federal relationship context models will grant us the flexibility to deal with relationship change over time?

It is our belief that relationship context should be treated as a first class entity, held and managed separately from any information describing the mappings required to use shared capabilities themselves. In addition, it is our belief that managing the relationship context separately will bring benefits to the process of maintaining local data integration with remote information capabilities in the presence of both local and remote change. As most change takes place within known relationship contexts and many new relationships take place in existing federal contexts, there is a huge potential for reuse, In addition, by maintaining mapping state, mapping lifecycle information and meta-data related to executable mappings, it helps to make re-use easier and thus more likely.

Thus we extend our previous work [12] on a Federal Relationship Manager (FRM), see section V below, that maintains models of inter-domain relationships and their constituent external concepts such as capabilities (abstractions of information resources or information access services), domains and federations. Mappings of external capabilities to internal models are also maintained, along with relationship context, mapping lifecycle and executable mapping meta-data that facilitate the management and reuse of mappings in the presence of change. This means that relationship meta-data can be automatically generated and maintained for all mappings and that semi-automation or tool support is available for the process of maintaining data integration in the presence of change. For example, the FRM can automatically generate suggested mapping sets when a new capability import event occurs, simply by searching the set of available mappings and knowing the originator and the federal relationship context of the request. Our approach also facilitates the development of mappings independently of the data access service method, e.g. SPARQL endpoint vs RESTful, and the reuse of data-access service configuration details across multiple capabilities within one federation, domain or relationship context. In addition, in this paper we introduce support for "transformers" which realize executable mappings in our architecture.

The rest of this paper is structured as follows: section II provides a brief use case to motivate the work, in section III a set of requirements are derived for the desired system, then section IV provides a brief survey of existing solutions in the semantic integration and mapping re-use space, the main contribution of the paper is outlined in section V which details the approach and models used to apply the Federal Relationship Manager to semantic integration with efficient reuse of mappings and executable mapping components, in section VI we provide a snapshot of our prototyping activities to data and finally section VII provides some conclusions and plans for future work.

II. USE CASE

We propose a business intelligence application that regularly scrapes semi-structured information from the web, converts it to a structured form with explicit semantics, such as RDF or OWL, and also gathers structured information or knowledge from a number of business partners (resellers, suppliers) and wholly owned subsidiaries. Each of these information or knowledge streams exists, to a greater or lesser extent, within a specific relationship, with a known scope, typical terms of engagement or data access and a history of observed behaviors.

Within the application there is a number of evolving internal models or views that are selectively combined with aspects of the external information to extract business intelligence used for predicting behaviors or influencing local decisions. These model combinations are enabled by a set of schema-level mappings between the local and external models and a set of transformers (executable mapping components) used to transform instances of one schema or ontology into another. An inter-domain adaptor or gateway is formed from a set of transformer instances.

The presence of change at multiple levels – internal models, external models and external relationships - creates important challenges for maintaining a system like this, especially in an efficient manner that re-uses the existing effort in building mappings and transformers.

III. REQUIREMENTS

We divide the system requirements into two aspects – managing change at the level of a relationship and maximizing reuse. The resulting challenges are outlined below.

A. Managing Change at the Level of a Relationship

Given access to sets of internal models, external capabilities (capabilities are abstractions of information resources or information access services), mappings between them, ways to

execute or instantiate the mappings and known federal relationships, leads to the following challenges:

- Can we relate a mapping to a relationship in a way that facilitates mapping reuse?
- How do we relate a capability to a mapping set? (This enables the ability to efficiently discover and deploy the set of mappings required for a capability.)
- How to best relate a mapping to its execution method?
- Is it possible to automatically generate mappings and transformers when a federal partner shares a new capability with us?
- How do we relate a change in a relationship shared capability set to a potential change in a mapping set?
- What are the important relationship states or the relationship lifecycle phases that are relevant to managing relationship change?

B. *Maximising Re-use*

The primary reason for tracking lifecycle and mapping context is to maximize the potential for reuse of this important resource and thus ease integration of new sources with at least partially familiar contexts and to enable re-integration of sources with semantic drift. There are several levels on which there is potential for reuse and these must be supported by any viable approach:

- Reuse of data access service configurations, credentials or locations across relationships, domains or capabilities within a federation. For example [11] describes how "the discovery problem is determining which SPARQL endpoint(s) to consult in answering a given query."
- Reuse from the set of all mappings available to us. How do we maximize re-use in the context of a given system/FRM deployment? a FRM Domain? a Relationship or a Capability? How does mapping reuse at the level of vocabularies and data-sets fit into this? Ideally, it would be advantageous to be able to apply mapping rules in a cascading manner upwards from capability→relationship→domain→federation→FRM instance.
- Reuse of executable mapping components, which we call transformers. These components implement complex mapping logic, often for individual properties or concepts, in a variety of representations that may vary between Java code to sets of formal logic declarations.
- The availability of suitable tool and process support for reuse tasks like mapping discovery or automated mapping meta-data generation is also an important factor.

IV. EXISTING SOLUTIONS

Systems or data integration is a fundamental issue for any deployed information system and hence is a long-studied topic. In the brief discussion that follows we focus on approaches to semantic integration, which is integration at a conceptual rather than syntactic level and the related issue of re-use of integration artifacts such as mappings, code, gateways and so on. To our knowledge there is no related work on relationship modeling as a source of context for semantic integration or as a means to maximize reuse of integration artifacts.

A. *Application of Ontologies to Data Integration*

Ontologies, when used as a means to express a machine-processable shared conceptualization or model of a domain, are a natural aid to semantic integration once there are suitable technologies and tools available (c.2000 for widespread systems). However ontology heterogeneity, even within a common domain, is a large problem, especially for federated systems. For much of the last decade the semantic mapping community has concentrated on methods to identify concepts in independently developed ontologies that relate to one another [13]. Nonetheless there has also been significant work to date on complete system solutions for semantic integration:

1) *Early Studies*

With *Observer* in 2000, Mena et al. [14] introduce a centralized architecture based on wrapper ontologies expressing the contents and capabilities of distributed repositories to support RDB-like queries across the federated system. The early use of description logics to describe the information resources was significant, as was the central role played by mappings, but ultimately the type of federation envisaged is more like corporate federated databases, which can support a mix of limited local autonomy and ultimate centralized control. This is very different to the loosely coupled and entirely autonomous federated systems we aim to support.

Conceptually, the approach of defining "upper ontologies" and using these as generic bridges between models is attractive [15] but the practical aspects of developing such large and widely agreed models has meant that little progress has been made on this method in realistic, open systems.

Bockting [16] introduced a more general "semantic translation service" architecture in 2004 that aimed to utilize ontologies and early semantic web technology to translate arbitrary message exchanges between entities. However he identified the immaturity of semantic mapping technology, even at the level of a lack of agreed mapping representation formats as a large issue. At around the same time, Jain and Zhao [17] proposed an integration approach that utilizes web service interfaces as a solution to the system heterogeneity issue and ontologies as a solution for semantic heterogeneity. However once again the focus is on uniform query processing and this approach side-steps the issues of mapping creation, maintenance and evolution and ultimately re-use, despite the centrality of mappings. The overhead of mapping creation and management for semantic integration is seen as a necessary and manageable cost in these systems and there is no specific consideration of the dynamic nature of real world information repositories [1, 7].

2) *Linked Data Developments*

In a 2009 DERI technical report [18] on the anatomy of a linked data application the issue of data integration for applications consuming linked data (structured or semantic data published on the web) is mentioned, but the underlying assumption seems to be that either the logic applied to this data is going to be so lightweight or an absence of local models reduces the integration problem to one of interlinking within the public data cloud itself. In fact consumption of the often loosely structured and .highly heterogeneous linked data cloud with many publishers raises many semantic integration issues (unless relying on a single data source) [11]. Chief among these is the identification of co-referents, due to lack of a unique name assumption in OWL and the realities of a decentralized, web-scale publication model. Nonetheless, in terms of reusable mappings this is a first order issue and the challenge of semantic integration with linked data systems will be with us for some time to come.

3) *Emergent and Probabilistic Semantics*

Within the agent research community the concept of a totally open system with negotiated or agreed semantics from first principles is a long held goal. However we do not consider such systems as likely or usable for practical applications in the short term. Despite this, there are relevant approaches, such as dynamic selection of ontological alignments that have emerged from research in this domain [19]. In addition there is important recent work on emergent semantics for web-scale distributed systems [20] that argues that, given the scale of the integration problem (for example in terms of numbers of mappings required) and the inherent dynamism of such a system, nothing more than a current best estimate for each mappings is attainable. An acceptance of change at the centre of such a system reinforces the importance of providing mechanisms to manage this. This probabilistic approach contrasts with our own by admitting a loss of fidelity while ensuring greater flexibility. In future work it will be important to quantify the overheads, advantages and limitations of these contrasting approaches and perhaps see if there is a hybrid approach possible that will take the best of each.

B. *Mapping Software Re-use*

Typically the development of executable code to automate mappings, which we call transformers, must be developed ad-hoc, by domain experts, and this process is expensive [4]. Furthermore, the data transformation code is often very similar to existing code and reusing this code could provide a great benefit [7]. The issue of reuse is therefore important. In [21] it is stated that "reusability implies some management of build, packaging, distribution, installation, configuration, deployment, and maintenance and upgrade issues". It is therefore important to follow some methodology to provide reusability. A six phase methodology is described in [22], with one of the phases being "Acquiring, instantiating, and/or modifying existing reusable components". The Transformers system that is part of our approach is intended to assist with this phase, by associating executable code with semantic maps in a structured way. By reasoning over the ontologies that have been associated with the executable code, it is hoped that reusable elements of the executable code can be discovered.

V. The Federal Relationship Manager

A. *Overview*

The Federal Relationship Manager (FRM) [12] is a software system designed to manage dynamic relationships between autonomous domains within and across organizations. It manages the lifecycles of these relationships and enables capabilities to be imported and exported between related domains. Capabilities incorporate cryptographic credentials and URI references to RESTful services which provide semantic descriptions of the schemata and the set of data access services that the capability represents. Imported capabilities can be associated with executable semantic mappings – mappings which are applied automatically to any data consumed from the source represented by the capability – providing automation support for the mapping of heterogeneous data sources to a common internal model. The FRM provides extensive support for the construction and distribution of secure, fined-grained data access policies between peered FRM instances, it can equally be applied to open data sources – simply by defining a domain within an FRM to represent the data-provider and allocating a capability to that domain that identifies the data-access modality and URL. This allows the FRM's semantic mapping management features to be applied to the data-source. .

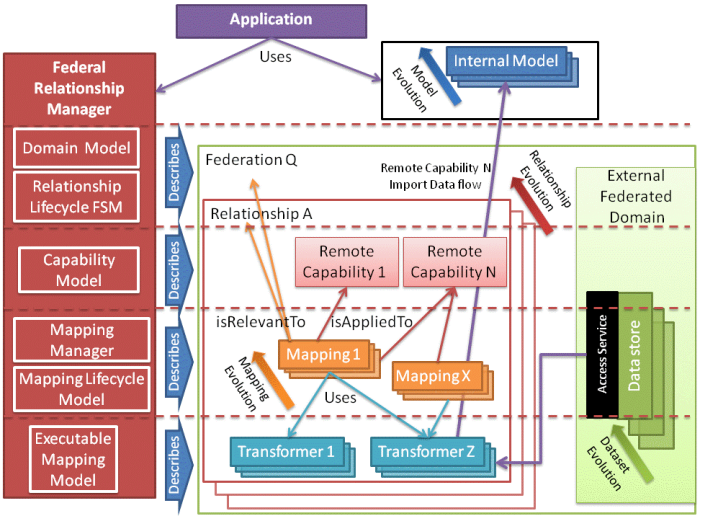

Figure 1. Models Maintained by the Federal Relationship Manager

The FRM has been designed with the requirements outlined in section III in mind, that is supporting management of change at the relationship level, and maximizing the opportunity for reuse. FRM maintains a set of internal models to support the management of the lifecycle of these relationships and the capabilities and mappings that are imported and exported between them. Figure 1 above shows on the left hand side the main models maintained by the system.

- The domain model captures the relationships between domains (hosted both locally and on remote FRMs).
- The capability model provides semantic descriptions of capabilities exported by local domains and a data access service providing capability comparisons – which is a key enabler of management delegation.

- The Mapping life-cycle model captures a wide range of meta-data associated with a particular mapping and is optimized to encourage re-use of mappings.
- The executable mapping (transformer) model defines automation processes for data made available through imported capabilities – it supports the definition of both schema transformation and onward process integration – allowing data imported from remote data sources to be automatically mapped to internal format and incorporated into internal process workflows.
- The relationship lifecycle evolves according to a Finite State Machine model, which can incorporate the negotiation and exchange of sharing policies and schemata. Figure 2 below shows the portion of the FSM that deals with relationship instantiation.

These models, between them, describe the evolution of the various inter-domain relationships and provide life-cycle management support to the administrators of those domains. The evolution of imported datasets, internal models, relationships and mappings is an integral part of the system.

B. *Modeling Federal Relationships as Mapping Context*

The set of models that the FRM contains allow imported data to be associated with mappings that operate on a variety of scopes, by associating mapping meta-data with model components. Specifically, mappings can be specified that operate at the system level, the remote-domain level, the local domain level, the relationship level and the capability level.

- *System scoped mappings* allow administrators to specify that a particular transformer should be applied to all imported capabilities that match a given schema. For example, we may wish to map all location data to a particular internal format. Specifying mappings for the various common internet geo-location standards at a system level achieves this without any need to replicate the mapping for particular sources or capabilities.
- *Remote-domain scoped mappings* allow administrators to specify that particular transformers should be applied to all capabilities imported from a given domain. For example, if a given provider uses a certain field in a schema in a non-standard way, this can be specified a single time in a remote-domain scoped mapping.
- *Local Domain scoped mappings:* FRM domains represent local organizational units (divisions, teams, etc..) that have some degree of management autonomy. These mappings allow administrators of local domains to specify their own domain-specific mappings.
- *Relationship scoped mappings* allow the specification of mappings that will apply to all capabilities imported by a particular local domain from a particular remote domain.
- *Capability scoped mappings* allow mappings to be specified that apply to only a particular imported capability (representing a particular remote data source).

The mappings defined for any given data source are applied in a layered manner in the above order – each mapping can either replace or modify the above mappings, producing a composite transformation on imported data as output. The combination of layered mappings and the variety of scopings can greatly improve the efficiency and convenience of dealing with data-set dynamics in many common situations. For example, if we are importing data that uses a particular standard schema from a number of different sources, and one of them happens to make a minor modification to their interpretation of a particular field, we can create a mapping that simply changes the mapping applied to this particular field and give it a remote-domain or capability scope. We do not have to create a new full mapping and our model incorporates the underlying connection between the flavors of the imported schema automatically.

C. *Relationship Lifecycle Finite State Machine*

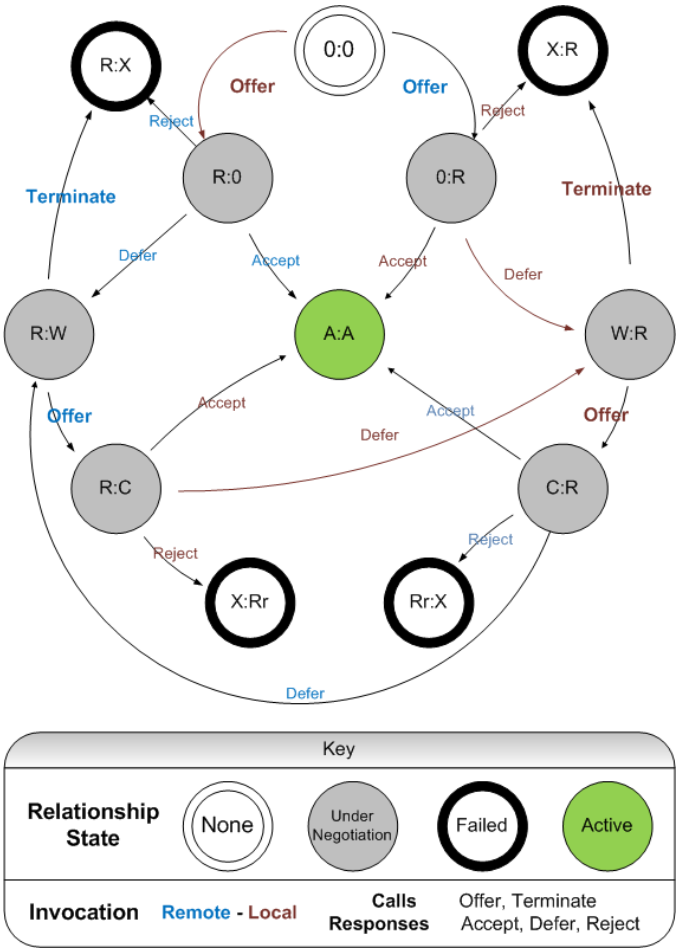

Figure 2. Relationship Lifecycle Model for Relationship Instantiation

The FRM incorporates support for the automated or semi-automated generation of mappings as part of its relationship lifecycle management functionality. When a new relationship is created with a remote domain, if a schema describing the data services provided by that domain is available, the FRM

can be configured to invoke a mapping generation process that will attempt to apply existing mappings to the imported data. When a new capability is imported, the FRM will attempt to use the capability description service (whose URL is included in the capability) to generate an appropriate mapping from existing ones.

The inter-FRM protocol, implemented as a RESTful API, also supports schema negotiation between domains managed by different FRM instances. Relationships are requested by making an *Offer* call to a remote domain. The payload of this offer can include a semantic description of the data-services provided by the domain. Figure 2 below shows the finite-state machine that governs the relationship negotiation between remote domains. Its design ensures that, in order to activate a relationship, a synchronous *accept* response must immediately follow an offer – by sending a *defer* response, a domain may make a subsequent counter offer, thus enabling support for negotiation of schema and relationship rules.

D. OM²R, a Mapping Lifecycle Model Supporting Reuse

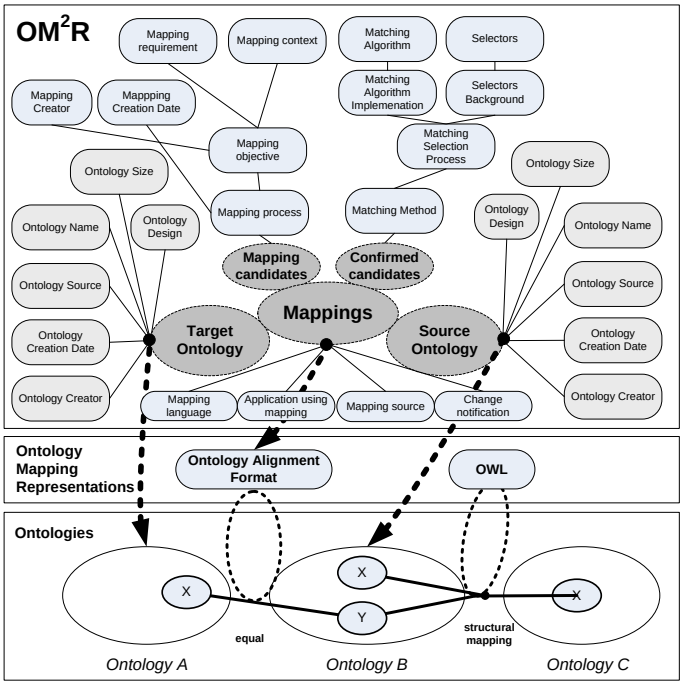

Figure 3. An overview of OM2R

OM²R (fig. 3) is designed as an independent meta-data layer describing mappings which means that it can simultaneously describe mappings created by a number of mapping generation systems. Its deployment concept is illustrated in figure 3. The lowest layer of the figure contains the ontologies which are the subject of mappings. The middle layer contains the mapping representations which specify the specific correspondences between the addressed ontologies. Any suitable mapping representation can be used like a rule based notation, XSLT or an OWL document. The highest layer is used to model meta-data independent of any specific mapping representation. The meta-data layer is composed of concepts, properties and individuals and provides a common vocabulary for documenting mappings. The highest layer concepts can be flexibly linked to the individual mapping representations or transformers that implement them.

The mapping context field is populated with the federal relationship properties illustrated in figure 1: isRelevantTo and isAppliedTo, these respectively link FRM domain/federation and capability models with OM2R mapping lifecycle models. These properties coupled with OM2R lifecycle meta-data allow tracking of which mappings relate to which capabilities and in what federal or domain contexts over time. This supports mapping discovery and reuse within specific federal relationship contexts.

E. Transformers (Executable Mappings) Meta-data Model

Transformers are used to realize executable mappings in our architecture. Transformers are described as instances of OWL classes. They combine executable code describing data transformations, with metadata to describe the nature of these transformations. This metadata helps us identify when the transformation can be re-used in a new mapping task. Transformers are referenced within the OM2R model as instances of ontology mapping representations. The following meta-data is recorded in each Transformer instance:

TABLE I. TRANSFORMER META-DATA[1]

Metadata Field	**Recorded In**	**Description**
Inputs	esm:usesProperty property of DataSource	An rdf:Property reference for each of the inputs of the transformer
Outputs	esm:usesProperty property of DataTarget	An rdf:Property reference for each of the outputs of the transformer
Heterogeneity Type	esm:correctsHeterogeneityType property of Transformer	The type of heterogeneity this transformer resolves. There are twelve broad types of heterogeneity, as identified in [1].
Language	esm:usesLanguage of TransformFunction	Language used in the transformation function. Eg. Python, XSLT
Dependencies	esm:dependsOn property of Transformer	A reference to any other Transformers that must be run before this one – the inputs and outputs of transformers may be chained together.
Date of Creation	esm:created property of Transformer	An xsd:date recording the time the transformer was created
Previous Versions	esm:perviousVersion property of Transformer, DataSource, DataTarget, or TransformFunction	A reference to any deprecated Transformer this one replaces.

[1] The esm namespace refers to http://kdeg.scss.tcd.ie/ExecutableSemanticMappings

VI. PROTOTYPING TO DATE

A. FRM Prototype

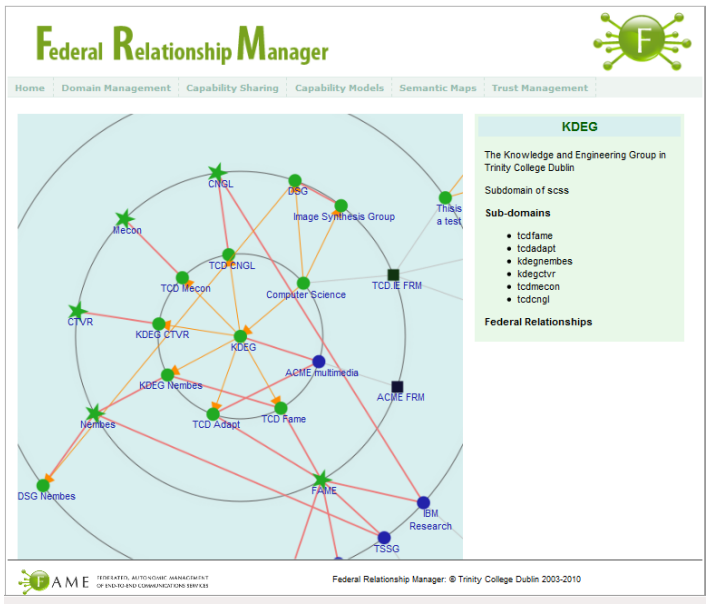

Figure 4. FRM Prototype Domain Management Interface

Figure 4 shows a screenshot from our FRM implementation. It allows domain administrators to manage the integration of data and services from multiple independent sources. The nodes in the diagram represent domains, while the lines represent relationships between domains. An FRM instance can manage multiple local domains (green nodes) and their relationships with other domains which may be hosted on remote FRM instances (blue and black nodes). The relationship management services of the FRM include authorization management, and policy-based automation as well as semantic mapping. The services are accessed through a RESTful interface.

B. MooM – a Framework for Management of Ontology Mappings

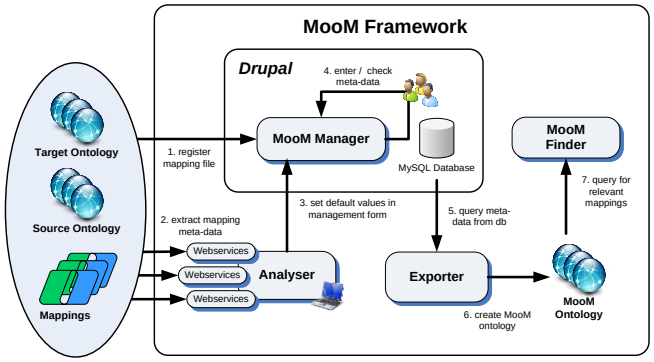

Figure 5. The MooM Mapping Management Framework

An important gap in the state of the art is tools and methods to generate, maintain and apply meta-data for ontology mappings. For this purpose we designed the MooM framework for the management of meta-data for ontology mappings. It uses the OM^{2}R model described above to model the mappings and their federal relationship context. Currently MooM consists of the components shown in Figure 5 and discussed below. Note that in the discussion the numbers referenced relate to the arrows shown in Figure 5.

MooM Management System: This system is the core of the framework. It provides a graphical interface for uploading and maintaining the relevant meta-data for ontology mappings. It is based on the modular content management system Drupal[2]. A new Drupal module implementing a custom multi page form was developed that enables a user to register an ontology mapping into the system (1) and to enter all necessary meta-data (4). Also an overview of all currently available mappings is provided.

MooM Analyser: All relevant meta-data fields can be entered manually by the uploading user. However, in order to simplify the editing process, a set of web services has been developed which analyses a given mapping or ontology in order to automatically extract relevant meta-data fields (2). These values are then shown in the form as default values (3) and can be checked by the editing user. For example, the web service getOntoLang() automatically identifies the language of an addressed ontology or getAuthor() returns the author of the mapping if previously specified. All web services are based on the JAVA web services framework AXIS 2.

MooM Exporter: All meta-data on ontology mappings is stored in a MySQL database. The MooM Exporter connects to the database (5) and transforms the meta-data into the OWL based OM^{2}R ontology (6). By exposing the semantics of the meta-data in an ontology, they can be more easily reused and queried.

MooM Finder: This service is used to discover existing ontology mappings based on the OM^{2}R ontology (7). With a set of appropriate SPARQL queries the MooM Finder can easily support 15 common mapping discovery tasks and custom queries allow full flexibility.

C. Executable Mappings Framework

The Transformer Framework is used at the lowest level of reuse and supports the work of the FRM. It is designed to allow executable code, for example relating to a remote repository access service, to be associated with individual semantic mappings. Each semantic mapping may also be associated with a set of reusable Transformer objects which perform the more low-level operations needed to share data between domains.

These transformers are based on the well known *Decorator* design pattern. In a Transformer object, functionality for extracting information from a source ontology is delegated to DataExtractor objects, all functionality for processing this information is delegated to a TransformFunction, and the transformed output is sent to a target ontology by a DataTarget object. Using this decorator design pattern allows us to easily store and retrieve Transformers as elements in an OWL Ontology. This facilitates reasoning about Transfomer components to categorise them and to provide appropriate

[2] http://www.drupal.org

suggestions for reuse or customization when a new mapping is deployed in a particular federal context.

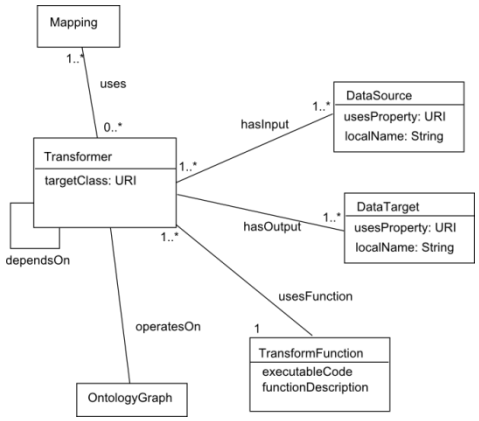

Figure 6. Transformer Structure

VII. CONCLUSIONS

We have presented a novel approach to generating context for semantic mappings to support their reuse in efficient semantic integration of multiple federated data sources over time. This is based on utilizing explicit models of the federal relationships to scope mapping deployments and hence likely reuse candidates. An additional level of reuse is ensured by connecting individual mappings to sets of executable mapping components called transformers that provide atomic instance transformation code that can be used by multiple mappings.

Central to the issue of consumption of data from multiple heterogeneous, noisy sources is the idea that there will be commonalities in the subject domain of interest, the autonomous domains involved and the federations or relationships that frame this consumption. Each of these axes of commonality gives a potential for re-use and hence efficient re-integration. Thus they are all considered in our mapping context models.

Due to a lack of experimental results at this time it is not possible to provide firm evidence for the claim that these federal relationship models provide useful context for reuse but given the phenomenon of semantic drift and the reports [1] of circa 80% static behavior by evolving data publishers it indicates at the very least a strong potential for reuse within the context of a specific consumer-publisher relations

Dealing with relationship change over time is the ultimate motivating factor for this work since static (or pre-deployment) systems rarely exhibit the complexity that stems from systems evolution. Future field evaluations will have to report on this issue.

REFERENCES

[1] Towards Dataset Dynamics: Change Frequency of Linked Open Data Sources- J¨urgen Umbrich, Michael Hausenblas, Aidan Hogan, Axel Polleres, Stefan Decker

[2] Shvaiko, P., Euzenat, J.: Ten Challenges for Ontology Matching. In Proceedings of OnTheMove Federated Conferences & Workshops, 2008.

[3] Thomas, H., O'Sullivan, D., Brennan, R.: Ontology Mapping Representations: a Pragmatic Evaluation. In: 21st International Conference on Software Engineering and Knowledge Engineering (SEKE) Boston, pp. 228 – 232, 2009

[4] Euzenat,J., Scharffe, F., Zimmermann, A.: Expressive alignment language and implementation. Deliverable 2.2.10, Knowledge Web NoE, 2007.

[5] O'Sullivan, D., Wade, V., Lewis, D. Understanding as We Roam, In: IEEE Internet Computing, 11, (2), 2007, p26 - 33 DOI: http://doi.ieeecomputer society.org/10.1109/MIC.2007.50

[6] Moser, T, et al.: Semantic Service Matchmaking in the ATM Domain Considering Infrastructure Capability Constrains. In: 21st International Conference on Software Engineering & Knowledge Engineering. SEKE 2009, 1-3.06, Boston, 2009, pp. 222 – 227

[7] Weidman, S., Arrison, T. : A Steps Toward Large-Scale Data Integration in the Sciences, ISBN: 0-309-15443-X, 2010.

[8] Jennings, B., Brennan, R., Donnelly, W., et al.: Challenges for Federated, Autonomic Network Management in the Future Internet, 1st IFIP/IEEE International Workshop on Management of the Future Internet, New York, 5 June 2009, IEEE, 2009

[9] Yang, K., Steele, R.: A Framework for Ontology Mapping for the Semantic Web, In: Proc. of the International Conference on Information Technology in Asia, http://www-staff.it.uts.edu.au/ ~kayang/download/AFOMSW.pdf, 2007.

[10] Alexander, K. et al., Describing Linked Datasets, LDOW2009, April 20, 2009, Madrid, Spain

[11] Millard, I., Glaser, H., Salvadores, M. and Shadbolt, N.: Consuming multiple linked data sources: Challenges and Experiences. In: *First International Workshop on Consuming Linked Data (COLD2010)*, Shanghai, 2010.

[12] Kevin Feeney, Rob Brennan, John Keeney, Hendrik Thomas, Dave Lewis, Aidan Boran, Declan O'Sullivan, Enabling Decentralised Managem*ent through Federation, to appear in Elsiver Computer Networks* 2010

[13] Shvaiko P. and Euzenat J.: *A Survey of Schema-based Matching Approaches*, Journal of Data Semantics, Long version in DIT Technical Report DIT-04-87, 2004

[14] Eduardo Mena, Arantza Illarramendi, Vipul Kashyap, and Amit P. Sheth. Observer: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. Distrib. Parallel Databases, 8(2):223{271, 2000.

[15] Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., and Hübner, S. (2001) Ontologybased Integration of Information - a Survey of Existing Approaches, Proceedings of the IJCAI Workshop on Ontologies and Information Sharing, pp. 108--117.

[16] S. Bockting , A Semantic Translation Service using Ontologies Advances inWeb-Age Information Management Lecture Notes in Computer Science, 2004, Volume 3129/2004, 249-258

[17] Jain, Hemant and Zhao, Huimin, "Federating Heterogeneous Information Systems Using Web Services and Ontologies" (2004). AMCIS 2004 Proceedings

[18] Hausenblas, M. , Linked Data Applications, DERI Technical Report 2009-07-26, July 2009.

[19] Paul Doran and Valentina Tamma and Terry R. Payne and Ignazio Palmisano, Dynamic Selection of Ontological Alignments: A Space Reduction Mechanism,

[20] [R12] Philippe Cudré-Mauroux, Emergent Semantics: Rethinking Interoperability for Large Scale Decentralized Information Systems, Self-Organizing Knowledge Systems Symposium, VU Amsterdam, April 29, 2010

[21] P.S. Sandhu, "A Survey on Software Reusability," 2010 International Conference on Mechanical and Electrical Technology (ICMET 2010), 2010, pp. 769-773.

[22] K.C. Kang, S. Cohen, R. Holibaugh, J. Perry, and A.S. Peterson, "A Reuse-Based Software Development Methodology," Application of Revisable Software Components Project Special Report, 1992