A Coreference Service for Enterprise Application Integration using Linked Data

Miguel Esteban-Gutiérrez, Raúl García-Castro,Nandana Mihindukulasooriya

Ontology Engineering Group, Center for Open Middleware
Universidad Politécnica de Madrid, Spain
{mesteban, rgarcia, nmihindu}@fi.upm.es

Abstract: The use of semantic and Linked Data technologies for Enterprise Application Integration (EAI) is increasing in recent years. Linked Data and Semantic Web technologies such as the Resource Description Framework (RDF) data model provide several key advantages over the current de-facto Web Service and XML based integration approaches. The flexibility provided by representing the data in a more versatile RDF model using ontologies enables avoiding complex schema transformations and makes data more accessible using Web standards, preventing the formation of data silos. These three benefits represent an edge for Linked Data-based EAI. However, work still has to be performed so that these technologies can cope with the particularities of the EAI scenarios in different terms, such as data control, ownership, consistency, or accuracy.

The first part of the paper provides an introduction to Enterprise Application Integration using Linked Data and the requirements imposed by EAI to Linked Data technologies focusing on one of the problems that arise in this scenario, the coreference problem, and presents a coreference service that supports the use of Linked Data in EAI systems. The proposed solution introduces the use of a context that aggregates a set of related identities and mappings from the identities to different resources that reside in distinct applications and provide different views or aspects of the same entity. A detailed architecture of the Coreference Service is presented explaining how it can be used to manage the contexts, identities, resources, and applications which they relate to. The paper shows how the proposed service can be utilized in an EAI scenario using an example involving a dashboard that integrates data from different systems and the proposed workflow for registering and resolving identities. As most enterprise applications are driven by business processes and involve legacy data, the proposed approach can be easily incorporated into enterprise applications.

1 Introduction

The purpose of Enterprise Application Integration (EAI) is to interconnect applications (including legacy ones) and data sources in order to enable business information interchange in enterprises. In the last years, the use of the W3C Semantic Web activity standards[1] and of the Linked Data principles[2] in EAI scenarios is increasing because of their support to

[1] http://www.w3.org/2001/sw/
[2] http://www.w3.org/DesignIssues/LinkedData.html

knowledge representation and data reuse and integration. However, EAI imposes different requirements over semantic and Linked Data technologies that have to be overcome in order to successfully deploy EAI systems in enterprises.

One of this requirements, which is the focus of this paper, is to solve the *coreference problem*. This problem appears when information about a certain entity is spread across different sources. It is a frequent problem in fields such as information science, where different repositories hold information about the same entity, or databases, where different databases include records describing the same entity [EIV07].

The coreference problem can be divided into three different sub-problems: finding equivalences between entities, representing such equivalences, and processing equivalences. And in the Linked Data field, different approaches have been proposed to cope with them.

However, these approaches are not enough for EAI scenarios because in these cases enterprise data are controlled and governed by their relevant authorities and they are required to be consistent and accurate.

The goal of this paper is to present a coreference service for Enterprise Application Integration using Linked Data, which satisfies the requirements imposed by the EAI scenario while complying with the Linked Data principles. The authors have put the proposed service into practice in the ALM iStack project to solve the coreference problem in a system that demonstrates the integration of Application Lifecycle Management (ALM) tools using the Linked Data principles.

This paper is structured as follows. Section 2 puts our work into the context of the EAI field and presents those requirements imposed by EAI to Linked Data technologies. Section 3 discusses other approaches in the Linked Data field to cope with the coreference problem and section 4 describes our proposal for a coreference service for EAI using Linked Data. Finally, section 5 draws some conclusions and presents future lines of work.

2 Using Linked Data for Enterprise Application Integration

With the growth of Information System infrastructures for managing business processes and the deployment of enterprise applications alongside legacy systems, there is a high demand for integrating applications without making significant changes to those applications or their underlying data models [SELB04, Qur05]. Enterprise Application Integration (EAI) is defined as "the unrestricted sharing of data and processing among any connected applications and data sources in the enterprise" [Lin00] and it "combines the technologies and processes that enable customer built and/or packaged business applications to exchange business level information in formats and contexts that each understand" [RWD99].

Requirements of EAI can be identified at five different levels: information integration, application connectivity, process integration, user interaction, and legacy-system integration [Juj09]. Out of the five levels, information integration and application connectivity are the basic requirements. In addition, there are high-level requirements that have to be met by enterprise applications that also apply for EAI: concurrent access, data integrity, transactions, performance, responsiveness, efficiency, reliability, extensibility, and scalability [Fow03, lHN12].

There are several approaches to EAI: information-oriented, business process integration-oriented, portal-oriented, and service-oriented [Lin04], being the latter one the basis for the prominent solutions in the industry so far. These solutions utilize Service-Oriented Architectures and/or Message-Oriented Middleware (MOM) for integrating applications. According to the EAI functional model [MG08], these EAI solutions consist of three layers: Business Process Management Service Layer (Process Layer), Transformation & Routing Service Layer (Transformation Layer), and Core Integration Middleware Layer (Transportation Layer).

In any of the aforementioned approaches, information or data integration is one of the key challenges of Enterprise Application Integration. In this paper, we focus on one aspect of data integration that is the problem of coreference. This problem is also referred as reconciling data at instance level [HRO06], reference reconciliation [DHM05], object identification [TKM02], entity resolution [BG07], name matching [BMC+03], record linkage [BG04a], merge/purge [HS95] in the database, statistics, and EAI literature.

2.1 Linked Data and EAI

Linked Data is based on four rules defined by Tim Berners-Lee[3] with the intention of creating machine-readable data in the Web following the success of World Wide Web. In accordance with the third rule of Linked data, "When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL)", data are published using the Resource Description Framework (RDF) standards. RDF and Linked Data provide new opportunities to solve the complex problem of application integration.

RDF, with its simple and flexible data model, allows organizations to expose data that are normally locked in data silos and to facilitate integration by converting them to RDF and by publishing them using known vocabularies as private or public data, based on organizational needs [ADL+09]. RDF is specially suitable for data integration scenarios where (1) multiple-source data integration is required without the overhead of a large development effort; (2) data will be made available for reuse by stakeholders; (3) data are available in a decentralized manner, that is, no single stakeholder has responsibility for the entirety of data; or (4) enhanced use of large amounts of structured data is required (browse, query, match, extract, input) [OCH12]. The Linked Data Platform initiative[4] aims to standardize a protocol that can be used to read/write Linked Data which will facilitate application integration [lHN12]. In several domains including Application Life-cycle Management (ALM) and Financial Reporting, Linked Data is emerging as a new approach [OCH12, BLD+11] for facilitating Enterprise Application Integration.

However, there are some differences in data integration in enterprise applications that re-

[3] http://www.w3.org/DesignIssues/LinkedData.html
[4] http://www.w3.org/2012/ldp/

quire new approaches for data integration when using Linked Data. Enterprise Applications have a concrete data model and a defined business logic (processes) that has to be shared between integrated applications. Furthermore, enterprise data are controlled and governed by their relevant authorities and they are required to be consistent and accurate.

2.2 Coreference in EAI using Linked Data

One of the characteristics of Enterprise Applications is that they often handle complex data [Lin04]. Different divisions, different departments, or different companies engaged in enterprise business processes have information about the same entity scattered in many different places. Each of them often have a different view of the same entity and could contain partially or completely disjoint information about the same entity. This conceptual dissonance [Fow03] makes the coreference of EAI more complex and existing lexical and structural matching techniques are not much helpful.

Moreover, as the assignment of identifiers (i.e., URIs) is done by different organizational authorities that manage those specific views of the same entity, there is a lack of association between the identifiers of the same entity across different divisions [OCH12]. This makes it difficult to consolidate identifiers of the same entity thus leading to difficulties in data integration. However, the accuracy of the coreference identification is critical to the enterprise as they deal with monetary and other crucial information. Integrity and the consistency of those data have to be ensured during all the operations. As a result, the data managed by enterprises are well-controlled and governed by the relevant authorities.

In addition, in the context of Enterprise Application Integration, involved parties are not just interested in the contents of the entities integrated but also in its provenance, that is, to which application does the entity belong to, or more specifically, which view of the entity is handled by which application.

In an integration scenario, an application is defined by its *data model*, its *interface* (API), and its *business processes*. At run-time, an instance of the application exposes certain data –which conforms to the data model– that can be manipulated as required by the application business processes triggered when the application API is used. Thus, depending on the inputs specified when using a particular operation of the API, the business processes will operate over the data following the restrictions specified by the data model.

According to this scenario, in order to integrate two applications properly it is necessary to fully understand their data models, APIs, and business processes, so that the interactions between these applications can be done in the appropriate way.

However, in the Linked Data world, there is no assumption nor restrictions regarding the provenance of linked resources. Thus, in order to leverage Linked Data techniques for solving Enterprise Application Integration scenarios, it is necessary to **control the scope of an entity**, as well as to be able **to discover the restrictions that apply to the underlying data model** so that its contents can be updated safely.

As the integration scenario already requires the understanding of the data model and its

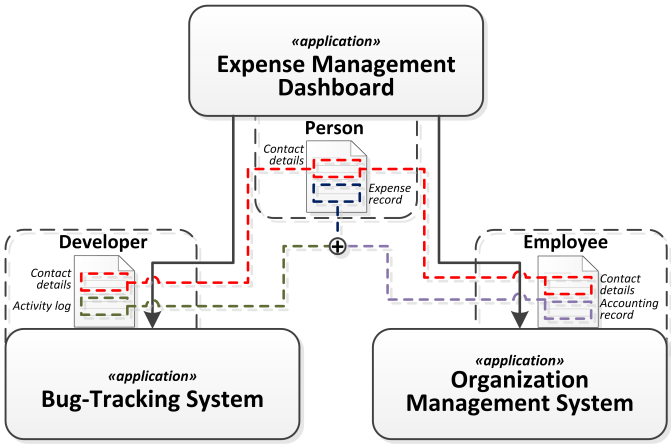

Figure 1: Integrating the Bug-Tracking System and the Organization Management System in the Expense Management Dashboard.

restrictions, we will focus on the entity scope issue and how it can be solved with a coreference service. The following section includes a running example showing the need as well as the requirements for this coreference service.

2.3 Sample EAI Scenario

Figure 1 shows a sample integration scenario where an **Expense Management Dashboard** (referred to as *EMD* from now on) integrates data from an **Organization Management System** (*OMS* from now on) and a **Bug Tracking System** (*BTS* from now on).

In this scenario, among the information exposed by the BTS are the details about the developers involved in the resolution of bugs. In particular, the contact information and an activity log that details how much effort do they timely put into solving bugs. On the other side, the OMS includes a full record of the employees, including not just their contact details but also their accounting records (i.e., cost per hour). Taking these facts into account, the objective of the EMD is to report the expenses of persons.

In order to integrate data between the BTS and the OMS it is necessary to relate the developers on the former system with the employees of the latter; that is, the EMD needs to know which developer of the BTS and which employee of the OMS are the same person.

In this simple, read-only scenario, the EMD can keep track of these relationships internally. However, moving to an scenario with more parties involved (for instance multiple bug-tracking systems), or with updates (i.e., reporting through the EMD), would increase the complexity of the EMD just for the sake of dealing with these relationships. In addition, if any of the other parties (BTS and/or OMS) were interested in integrating information, they would have to replicate the work of the EMD. As a result, it is a better practice to delegate these duties to a specialized third-party service that is capable of tracking these contextual relationships, which can be reused across applications.

3 Solving coreference for Linked Data

In the Linked Data field, and in the Semantic Web one in general, the problem of coreference appears because different URIs are used to describe the same non-information resource [JGM07]; and the main cause of this problem is the need for integrating data from different providers.

These data providers are responsible for assigning URIs to their information and non-information resources and, to do so, they use web domains over which they have control. Furthermore, data providers work under concrete assumptions and requirements and the heterogeneity of the provided data is prevalent at different levels, which makes the problem of coreference resolution harder [MVB08].

The coreference problem can be divided into three different sub-problems: finding equivalences between entities, representing such equivalences, and processing of equivalences. Next, we present existing approaches in the Linked Data field to solve that problems.

3.1 Finding equivalences

Different approaches cope with the problem of finding whether two entities are equivalent. Some of them exploit the semantics of the OWL language to do so (e.g., using inverse functional properties [HHD07]); others analyze whether entities share similar property-value pairs [FLM08, HPUZ10]; another one uses both the OWL language constructs and property-value pairs [HCQ11]; and in another equivalences are found through negotiations between agents [MDS08]. Furthermore, instance matching techniques [CFMV11] can also be applied to this respect.

In order to reduce the problem of equivalence finding, a proactive strategy can be followed to normalize identifiers before integrating data (in contrast with a reactive strategy as defined in [MDS08]). This strategy has been largely applied in fields such as information science through the use of controlled vocabularies.

Taking this strategy to the limit, the Entity Name System (ENS) [Sto08] appeared as a global repository of entity identifiers. By requiring people to look for an existing URI in the ENS before creating a new one, it is expected that multiple URIs do not exist for a single resource.

3.2 Representing equivalences

Entity equivalences can be represented either at the data level or at the infrastructure one.

At the data level, on the one hand, equivalences are represented along data using modeling language constructs, such as RDF-S [BG04b] or OWL [MPSP12] properties (being *owl:sameAs* the frequently used one) or SKOS [MB09] mapping properties (e.g., *skos:closeMatch*, *skos:exactMatch*).

At the infrastructure level, on the other hand, equivalence information is separated from data and infrastructure services are in charge of resolving whether two URIs belong to the same entity. Examples of such services are the Co-reference Resolution Service (CRS) [JGM07] and the Co-reference Knowledge Service (CKS) [MDS08].

3.3 Processing equivalences

Apart from coreference resolution, another required operation is coreference registration. Coreferences can be registered by the same resolution service, which is the case of the CRS where the same service uses different algorithms to identify equivalent resources, or by external services, which is the case of the CKS where agents register the coreferences resulting from their negotiations.

Another useful operation is the discovery of coreference resolution services. To enable such discovery, in the CRS entity descriptions contain properties that identify the corresponding resolution service.

Next, we discuss the differences between these approaches and the coreference service that we present in this paper.

In terms of equivalence finding, in our scenario application interactions are driven by the concrete business logic of the EAI system, which will be the one that ensures that no inconsistencies appear in data spread along applications while allowing different applications to use different URIs to describe a resource. Hence, this scenario does not require finding equivalences between entities, since equivalences are defined in a controlled manner.

As mentioned above, the ENS also removes the need for equivalence finding. However, the ENS approach poses another problem: when dereferencing a URI that identifies an entity, how can the user be assured that the contents that will be retrieved match those required in his application context? That is, if there is a single URI that identifies an entity whose definition is spread across different data sources, the user might be interested in the contents managed by a particular data source and no other. To cope with this problem, we introduce the notion of application contexts, to allow applications controlling the different scopes of an entity.

The CRS also supports managing different contexts by having different coreference services each for managing a different context. In our case, multiple contexts can managed by a single service.

Regarding equivalence representation, similarly as in the CRS and in the CKS, we represent equivalences at the infrastructure level. However, this does not remove the possibility of exporting equivalences to the data level if required.

One advantage of managing equivalences at the infrastructure level is that it enables developing advanced services for equivalence management. In our case, we have defined context-aware registration and resolution services.

4 A Coreference Service for EAI using Linked Data

As explained in section 2.2, in order to solve the coreference problem in EAI using Linked Data it is necessary to keep track of the contextual relationships among different views of the same entity, in particular which applications control which views, and what do these views contain.

The objective of the *Coreference Service* is to serve as a middleware facility that provides the means for tracking these contextual relationships in a domain independent manner so that the service can be reused across domains.

The rest of the section is devoted to present different aspects of the *Coreference Service* (data model, architecture, and implementation) as well as to show how the proposed service can be used for solving the example integration scenario presented section 2.3.

4.1 Coreference Model

In order to solve the coreference problem, the *Coreference Service* needs a model capable of representing the elements involved in the scenario as well as the constraints they must meet. This model is the *coreference model* and, as Figure 2 shows, consists of the following elements:

- **Resources**: represent ground data entities.
- **Applications**: represent software systems that expose *resources* using controlled URIs.
- **Identities**: represent sets of characteristics that allow the unambiguous identification of resources where these characteristics depend on the *context* at hand.
- **Entities**: represent collections of *resources* that have the same *identity*.
- **Contexts**: represent collections of *entities*.

The introduction of contexts that define the scope of identities allows to handle the coreference problem in a more modular manner. All these elements are identified by URIs.

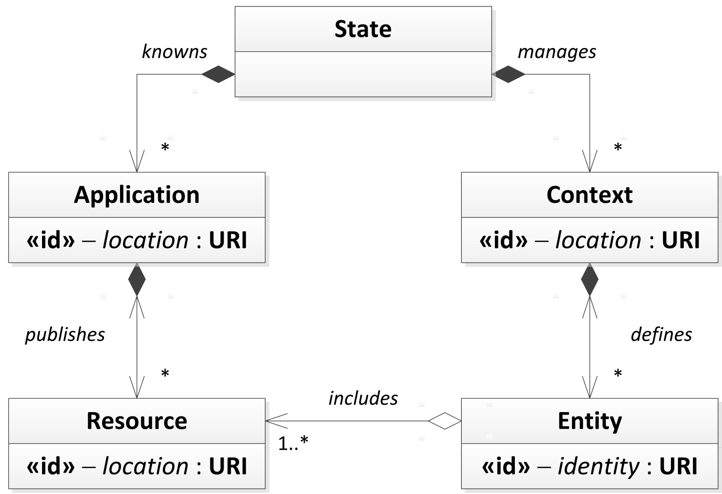

Figure 2: Elements of the *Coreference Model*.

However, whereas the URIs of resources and applications are not controlled by the *Coreference Service*, the URIs of identities, entities, and contexts are under its control.

The state of the *Coreference Service* consists of the set of applications and the set of contexts defined for the service. At any point in time, the following invariant holds for these sets: (1) the same resource cannot be exposed by different applications; (2) as identities are context-dependent, entities are confined within a single context; and (3) in a given context, a resource can only have a single identity; thus, in a given context, resources can only belong to a single entity.

4.2 Service Architecture and Implementation

The *Coreference Service* provides three differentiated ports, as shown in Figure 3: the **Identity Resolution** port, the **Application Management** port, and the **Context Management** port. Separation of these three concerns of managing different aspects facilitates a wide variety of usage patterns. Applications that are concerned about registering their resources, resolving references or aggregating data can use the appropriate ports in a loosely coupled manner.

The *Identity Resolver* port provides the *IIdentityResolver* interface, which defines the operations required for: (1) resolving the identity of a resource in a given context; and (2) discover the application-specific resource for a given identity.

On the other hand, the *Application Management* port and the *Context Management* port provide the *IResourceRegistry* and the *IContextRegistry* interfaces respectively. These interfaces provide granular creation, update, retrieval, and deletion operations for applications and contexts, respectively.

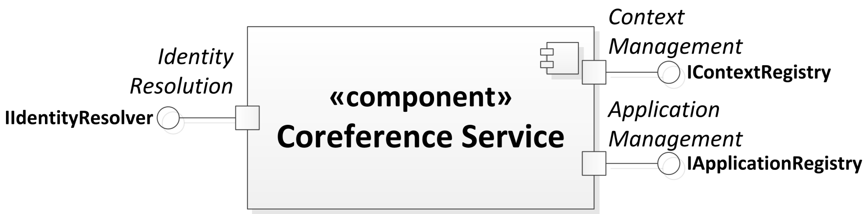

Figure 3: Ports exposed and interfaces provided by the *Coreference Service*.

To better address these concerns, the responsibilities of the *Coreference Service* are divided among three components, as shown in Figure 4: the **Application Manager**, the **Context Manager**, and the **Identity Resolver**.

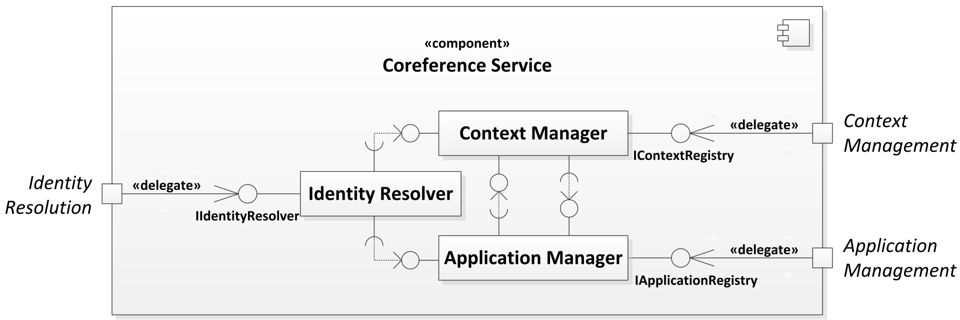

Figure 4: Components of the *Coreference Service*.

The *Application Manager* controls the life-cycle of applications and resources, according to the messages sent to the *Application Management* port of the *Coreference Service*. The processing of these messages occasionally requires the notification of state changes to the *Context Manager* (i.e., resource unpublishing).

Similarly, the *Context Manager* controls the life-cycle of contexts, identities, and entities, according to the messages sent to the *Context Management* port of the *Coreference Service* and/or the state change messages sent by the *Application Manager*.

Finally, the *Identity Resolver* handles the messages sent to the *Identity Resolution* port of the *Coreference Service*, querying the *Application Manager* and the *Context Manager* as required in order to serve the request.

4.3 Using the Coreference Service

Going back to the example integration scenario described in section 2.3, lets see how a developer available in the BTS and an employee in the OMS that have matching contact details –that is, they refer to the same person and thus can be integrated– would be integrated using the *Coreference Service* (see Figure 5).

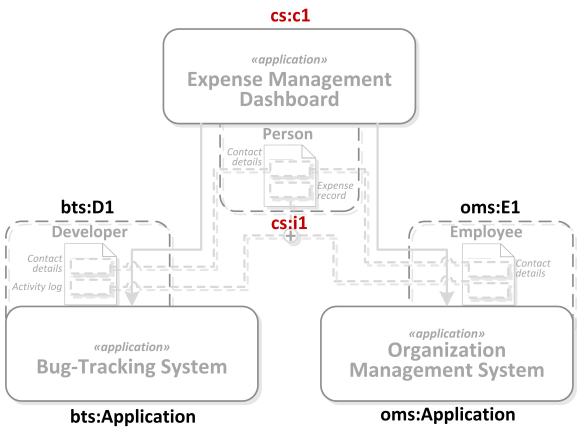

Figure 5: Example integration case.

The integration of the BTS and the OMS carried out by the EMD using the *Coreference Service* requires implementing a three stage process.

The first stage consists in populating the *Coreference Service* with the details about the resources used for exposing these data entities (see Figure 6).

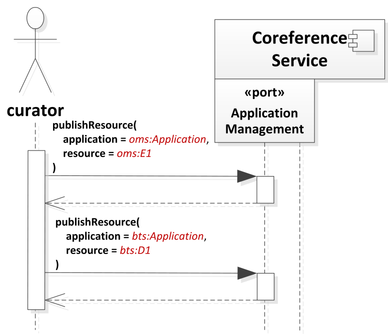

Figure 6: Populating resource details in the *Coreference Service*.

The second stage consists in creating an integration context for the EMD, and populating the details about the integrated entity within that context (see Figure 7).

Once this is all set-up, whenever the EMD is requested the expense records about the employee (or the developer), the EMD would be able to calculate them after resolving the resource using the *Coreference Service*, as shown in Figure 8.

It is worth pointing out that the first two stages can be carried out by different parties. In the example shown, a third party (the curator) takes care of populating the *Coreference Service*. This perfectly matches a legacy data integration scenario, where the information

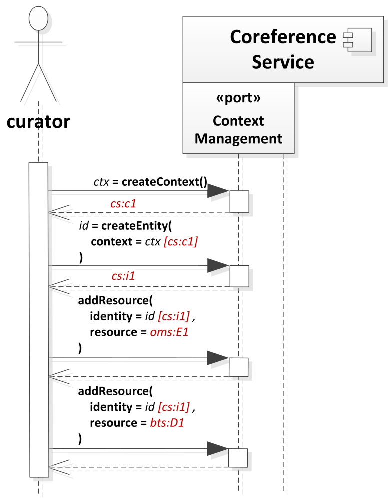

Figure 7: Managing the integration context.

about the individuals is already available and can be populated in advance. This population process can be performed in a manual, semi-automatic, or automatic fashion depending on the difficulty for identifying individuals.

However, this process could also be carried out on demand by the integrating party (the EMD in the example) when queried about an unknown individual. In this situation, the application would explore the integrated applications looking for the individual, and populate the *Coreference Service* for later usage. Then, the application would try to find matching individuals in the rest of the integrated applications, creating the required entities within its context in the *Coreference Service* as needed. In this case the identity resolution stage is implicit in the process.

5 Conclusions and future work

Linked Data provides new opportunities for data integration and enterprise applications can apply Linked Data principles to solve the EAI issues. This paper has described a coreference service that supports the use of Linked Data in EAI systems and that manages coreferences at an infrastructure level, while dealing with those requirements imposed by EAI systems.

Apart from being used in EAI scenarios, the coreference service could be used in other scenarios where data requires being controlled and accurate.

Future lines of work include analysing potential modifications and extensions of the coreference service when some of the restrictions imposed by the EAI scenario are relaxed or when the EAI system needs to integrate data from external data sources that do not comply

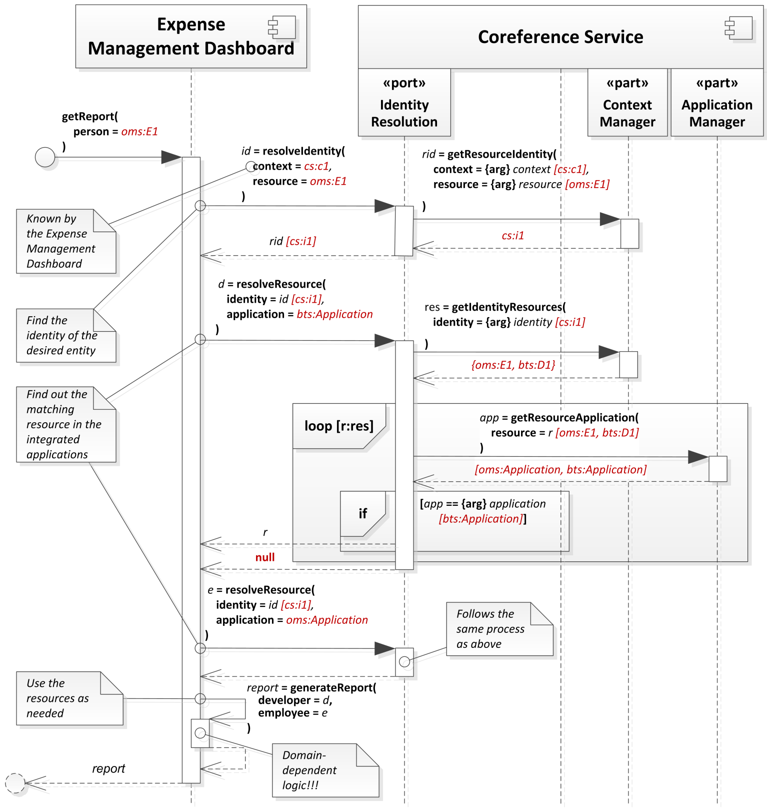

Figure 8: Leveraging the *Coreference Service* for resolving an individual.

with those restrictions (e.g., the Linked Open Data cloud).

Acknowledgments

The authors are partially supported by the ALM iStack project of the Center for Open Middleware.

References

[ADL+09] S. Auer, S. Dietzold, J. Lehmann, S. Hellmann, and D. Aumueller. Triplify light-weight linked data publication from relational databases. 2009.

[BG04a] I. Bhattacharya and L. Getoor. Iterative record linkage for cleaning and integration. In *Data Mining And Knowledge Discovery: Proceedings of the 9 th ACM SIGMOD* *workshop on Research issues in data mining and knowledge discovery*, volume 2004, pages 11–18, 2004.

[BG04b] D. Brickley and R.V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation 10 February, 2004.

[BG07] I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):5, 2007.

[BLD+11] O. Berger, S. Labbene, M. Dhar, C. Bac, et al. Introducing OSLC, an open standard for interoperability of open source development tools. In *ICSSEA*, 2011.

[BMC+03] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg. Adaptive name matching in information integration. *Intelligent Systems, IEEE*, 18(5):16–23, 2003.

[CFMV11] S. Castano, A. Ferrara, S. Montanelli, and G. Varese. *Ontology and Instance Matching*, volume 6050 of *Lecture Notes in Computer Science*, pages 167–195. Springer, 2011.

[DHM05] X. Dong, A. Halevy, and J. Madhavan. Reference reconciliation in complex information spaces. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 85–96. ACM, 2005.

[EIV07] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate Record Detection: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 19(1):1–16, January 2007.

[FLM08] A. Ferrara, D. Lorusso, and S. Montanelli. Automatic Identity Recognition in The Semantic Web. In *Proceedings of the 1st International Workshop on Identity and Reference on the Semantic Web (IRSW2008)*, volume 422 of *CEUR Workshop Proceedings*, Tenerife, Spain, June 2 2008.

[Fow03] M. Fowler. *Patterns of enterprise application architecture*. Addison-Wesley Professional, 2003.

[HCQ11] W. Hu, J. Chen, and Y. Qu. A self-training approach for resolving object coreference on the Semantic Web. In *Proceedings of the 20th International World Wide Web Conference (WWW2011)*, pages 87–96, Hyderabad, India, March 28 - April 1 2011. ACM.

[HHD07] A. Hogan, A. Harth, and S. Decker. Performing Object Consolidation on the Semantic Web Data Graph. In *Proceedings of the WWW2007 Workshop I^3: Identity, Identifiers, Identification, Entity-Centric Approaches to Information and Knowledge Management on the Web*, volume 249 of *CEUR Workshop Proceedings*, Banff, Canada, May 8 2007.

[HPUZ10] A. Hogan, A. Polleres, J. Umbrich, and A. Zimmermann. Some entities are more equal than others: statistical methods to consolidate Linked Data. In *Proceedings of the 4th Workshop on New Forms of Reasoning for the Semantic Web: Scalable & Dynamic (NeFoRS2010)*, Heraklion, Greece, May 31 2010.

[HRO06] A. Halevy, A. Rajaraman, and J. Ordille. Data integration: the teenage years. In *Proceedings of the 32nd international conference on Very large data bases*, pages 9–16. VLDB Endowment, 2006.

[HS95] M.A. Hernández and S.J. Stolfo. The merge/purge problem for large databases. In *ACM SIGMOD Record*, volume 24, pages 127–138. ACM, 1995.

[JGM07] A. Jaffri, H. Glaser, and I. Millard. URI Identity Management for Semantic Web Data Integration and Linkage. In *Proceedings of the 3rd International Workshop On Scalable Semantic Web Knowledge Base Systems (SSWS2007)*, pages 1125–1134, Vilamoura, Portugal, November 25-30 2007. Springer.

[Juj09] Z. Jujian. Apparel enterprise application integration model based on service-oriented architecture. In *Automation and Logistics, 2009. ICAL'09. IEEE International Conference on*, pages 1374–1377. IEEE, 2009.

[lHN12] A. le Hors and M. Nally. Using read/write Linked Data for Application Integration–Towards a Linked Data Basic Profile. *Linked Data on the Web (LDOW2012)*, 2012.

[Lin00] D.S. Linthicum. *Enterprise application integration*. Addison-Wesley Professional, 2000.

[Lin04] D.S. Linthicum. *Next generation application integration: from simple information to Web services*. Addison-Wesley Professional, 2004.

[MB09] A. Miles and S. Bechhofer. SKOS Simple Knowledge Organization System, Reference. W3C Recommendation 18 August, 2009.

[MDS08] C. Meghini, M. Doerr, and N. Spyratos. Managing Co-reference Knowledge for Data Integration. In *Proceedings of the 18th European-Japanese Conference on Information Modelling and Knowledge Bases (EJC2008)*, volume 190 of *Frontiers in Artificial Intelligence and Applications*, pages 224–244, Tsukuba, Japan, June 2-6 2008. IOS Press.

[MG08] T. Mahmoud and J.M. Gómez. Integration of semantic web services principles in soa to solve eai and erp scenarios; towards semantic service oriented architecture. In *Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on*, pages 1–6. IEEE, 2008.

[MPSP12] B. Motik, P. F. Patel-Schneider, and B. Parsia. OWL 2 Web Ontology Language, Structural Specification and Functional-Style Syntax (Second Edition). W3C Recommendation 11 December, 2012.

[MVB08] A. Morris, Y. Velegrakis, and P. Bouquet. Entity Identification on the Semantic Web. In *Proceedings of the 5th Workshop on Semantic Web Applications and Perspectives (SWAP2008)*, volume 426 of *CEUR Workshop Proceedings*, Rome, Italy, December 15-17 2008.

[OCH12] S. O'Riain, E. Curry, and A. Harth. XBRL and open data for global financial ecosystems: A linked data approach. *International Journal of Accounting Information Systems*, 2012.

[Qur05] K.A. Qureshi. Enterprises application integration. In *Emerging Technologies, 2005. Proceedings of the IEEE Symposium on*, pages 340–345. IEEE, 2005.

[RWD99] K. Ring and N. Ward-Dutton. Enterprise Application Integration: Making the Right Connections. *Ovum Ltd*, 1999.

[SELB04] A. Sharif, T. Elliman, P. Love, and A. Badii. Integrating the IS with the enterprise: key EAI research challenges. *Journal of Enterprise Information Management*, 17(2):164–170, 2004.

[Sto08] H. Stoermer. *OKKAM: Enabling Entity-centric Information Integration in the Semantic Web*. PhD thesis, University of Trento, January 2008.

[TKM02] S. Tejada, C.A. Knoblock, and S. Minton. Learning domain-independent string transformation weights for high accuracy object identification. In *Conference on Knowledge Discovery in Data: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, volume 23, pages 350–359, 2002.