Conceptual Schema Transformation in Ontology-based Data Access

D. Calvanese[1], *T. E. Kalayci*[2,1], M. Montali[1], A. Santoso[3,1], W. van der Aalst [4]

15 November 2018

21st Int. Conf. on Knowledge Engineering and Knowledge Management

[1]KRDB Research Centre for Knowledge and Data, Free University of Bozen-Bolzano (Italy)
[2]Virtual Vehicle Research Center, Graz (Austria)
[3]Department of Computer Science, University of Innsbruck (Austria)
[4]Process and Data Science (PADS), RWTH Aachen University (Germany)

Table of contents

Introduction

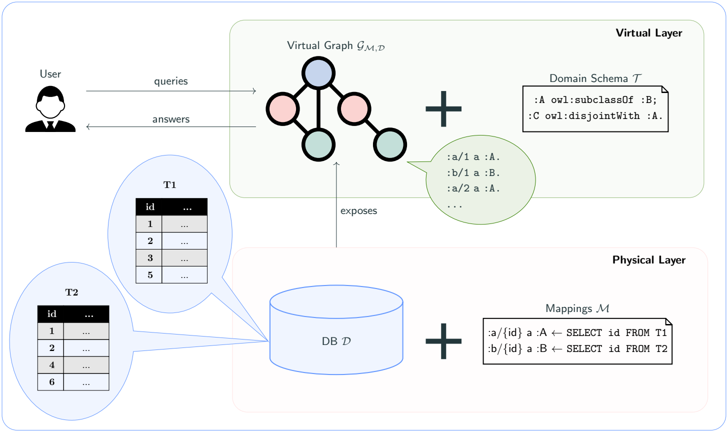
Conceptual Schemas

- To understand and document the relevant aspects of an application domain
- Used as live, computational artifacts
- Provides end-users with a vocabulary they are familiar with
- Masks how data are concretely stored
- Enrich (incomplete) data with domain knowledge

Mapping Specification

- To cover the abstraction mismatch between
  - domain schema
  - underlying data
- Declaratively links them to express how patterns in the data correspond to domain concepts and relationships

Ontology-based Data Access (OBDA)



Logical transparency in accessing data:

- does not know where and how data is stored;
- can only see a conceptual view of data.

Ontology-based Data Access (OBDA)

- Users do not need to code procedures for data extraction
- Domain experts autonomously interacts with legacy data without the manual intervention of IT savvy
- The actual data storage is completely transparent to end-users
- Data are not replicated and it is retrieved using the standard query engine of the information system
- From the foundational point of view, this is made possible [2]
  - by carefully tuning the expressive power of the conceptual modeling and mapping specification languages,
  - by exploiting key formal properties of their corresponding logic-based representations

On top of these foundations, several OBDA systems have been engineered, ontop is one of the main representatives in this spectrum [3] - `http://ontop.inf.unibz.it`

The Need of a Multi-level Approach to Data Access

When an OBDA specification is available

Certain types of users adopt reference models as an upper schema

- to understand the organization,
- to create reports, and
- exchange information with external stakeholders

For data analysis applications

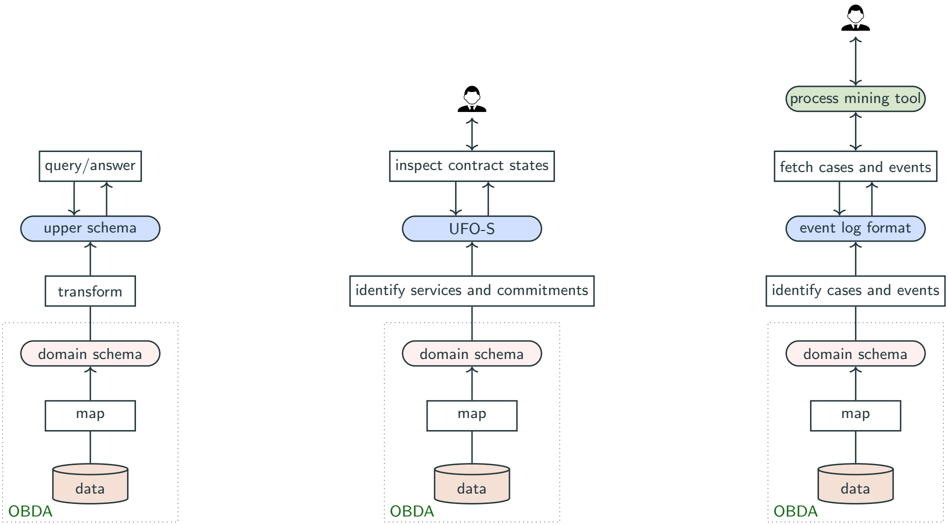
Data analysis applications are exploited to extract insights from legacy data

- The actual input for such applications consists of specific abstractions that may not be explicitly present in the legacy data, and
- Have to be represented according to the expected input data format

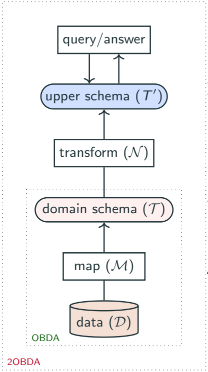
2OBDA Framework

- 2OBDA model is an elegant extension of OBDA
  - the conceptual transformation of concepts and relations in the domain schema into corresponding concepts and relations in the upper schema
  - 2OBDA specification can be automatically compiled into a classical OBDA specification that directly connects the legacy data to the upper schema, fully transparently to the end-users
- Supported by a tool-chain
  - End-users model the domain and upper schema, and specifies the corresponding transformations as annotations of the domain schema
  - Types and features of annotations are derived from the concepts present in the upper schema

2OBDA Framework



2OBDA Framework: Computing Certain Answers in 5 Steps



1. *rewrite* q to compile away the upper schema, obtaining $q'_r = \text{rew}(q, \mathcal{T}')$, which is a UCQ over the upper schema
2. use the schema transformation rules ($\mathcal{N}$) to *unfold* q'_r into a query over the domain schema, denoted by $q'_u = \text{unf}(q'_r, \mathcal{N})$, which turns out to be a UCQ
3. *rewrite* q'_u to compile away the domain schema $\mathcal{T}$, obtaining $q_r = \text{rew}(q'_u, \mathcal{T})$
4. use the mapping ($\mathcal{M}$) to *unfold* q_r into a query over relational database ($\mathcal{R}$), denoted by $q_u = \text{unf}(q_r, \mathcal{M})$, which turns out to be an SQL query
5. evaluate q_u over database instance, obtaining $\text{eval}(q_u, \mathcal{D})$

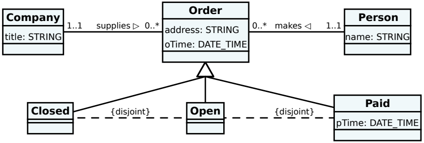
OHub Case Study

- An organization called OHub acts as a hub between companies selling goods and persons interested in buying those goods
- OHub takes cares of an order-to-delivery process that supports a person in
  - placing an order
  - paying the order
  - delivering the paid goods, etc.
- Employees of OHub use a legacy management system to handle orders, but they are not aware of
  - how the actual data about orders
  - how their involved stakeholders are stored
- OHub Managers wants to inspect
  - which commitments currently exist
  - in which state they are
- It is important for them to understand orders and their states in contractual terms

OHub Case Study: Challenges

- OHub managers cannot directly formulate queries of this form on top of the legacy data (vocabulary mismatch)
- A possible solution: create a dedicated OBDA specification that directly connects the legacy data to the UFO-S upper schema
    1. Unrealistic from the conceptual modeling point of view
    2. Reference models and upper ontologies are typically large
- Only a small portion of the whole reference model is needed to capture the commitments of interest in a specific application domain such as OHub

OHub Case Study: Domain Schema and Data



type = p   type = c   final = 1

id	name	addr	type
pa	alice	bz-4	p
ce	eDvd	na-1	c
pb	bob	tn-3	p

Stakeholders

id	ctime	from	to	dest	final
o1	5	pa	ce	bz-5	0
o2	10	pb	ce	null	1
o3	20	pa	ce	null	1

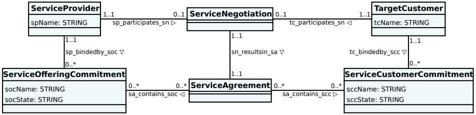
OrderData

id	order	ttime
t1	o2	15

MTransfers

- Each entry in the *OrderData* table corresponds to an order,
- Supplying company is obtained from the entry in *Stakeholders* pointed by the to column, and
- Making person is obtained from the entry in *Stakeholders* pointed by the from column
- the order is *open* if the corresponding entry in *OrderData* has final $= 0$
- *closed* if the corresponding entry in *OrderData* has final $= 1$, but no monetary transfer exists in *MTransfers* for the order
- *paid* if the corresponding entry in *OrderData* has final $= 1$, and there exists an entry in *MTransfers* pointing to the order

OHub Case Study: Defining OBDA Specification



- We can define an OBDA specification:
  - domain experts can forget about the schema of the legacy data, and
  - work directly at the level of the domain schema
- The domain schema can then employed to declare which concepts and relations define the UFO-S notions of
  - service provider, target customer, and corresponding offering and customer commitments
- We can declaratively specify that:
  - Each closed order gives rise to a *pending* customer commitment binding its making person (i.e., its target customer) to paying it.
  - Each paid order corresponds to a *discharged* customer commitment related to the order payment, and to a *pending* offering commitment binding its supplying company (i.e., its service provider) to delivering it

- Once the mapping and transformation rules are specified, OHub managers can express queries over UFO-S, and obtain answers automatically computed over the legacy data
- For example, upon asking about all the pending commitments existing in the state of affairs captured by the data in, one would get back two answers:
  1. one indicating that company `eDvd` has a pending commitment related to the delivery of order `o2`,
  2. one telling that person `Alice` is committed to pay order `o3`

type = p
type = c
final = 1

id	name	addr	type
pa	alice	bz-4	p
ce	eDvd	na-1	c
pb	bob	tn-3	p

Stakeholders

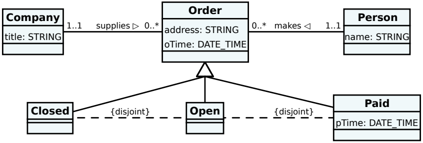
id	ctime	from	to	dest	final
o1	5	pa	ce	bz-5	0
o2	10	pb	ce	null	1
o3	20	pa	ce	null	1

OrderData

id	order	ttime
t1	o2	15

MTransfers

OHub Case Study: Conceptual Schema of Order System

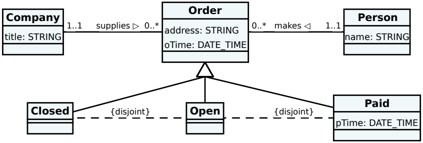


Intuitively concepts correspond to classes, roles to binary associations, and DL attributes to UML attributes

DL-Lite TBox assertions capture the conceptual schema

- *Open* and *Paid* are sub-concepts of *Order* ($Open \sqsubseteq Order$, $Paid \sqsubseteq Order$)
- *Paid* and *Open* are disjoint ($Paid \sqsubseteq \neg Open$)
- the domain of *name* is *Person* ($\delta(name) \sqsubseteq Person$)
- the domain and the range of *makes* are respectively *Person* and *Order* ($\exists makes \sqsubseteq Person, \exists makes^{-} \sqsubseteq Order$)
- orders are made by someone ($Order \sqsubseteq \exists makes^{-}$)
- the inverse of *makes* is functional ((funct $makes^{-}$))

OHub Case Study: Mapping Assertions



type = p
type = c
final = 1

id	name	addr	type
pa	alice	bz-4	p
ce	eDvd	na-1	c
pb	bob	tn-3	p

Stakeholders

id	ctime	from	to	dest	final
o1	5	pa	ce	bz-5	0
o2	10	pb	ce	null	1
o3	20	pa	ce	null	1

OrderData

id	order	ttime
t1	o2	15

MTransfers

- Mapping assertion to populate *Person* concept with the corresponding attribute *name*, by selecting in the table *Stakeholders* entries for which the value of type equals 'p'

  `SELECT id as pid, name FROM Stakeholders WHERE type = 'p'`

  $\rightsquigarrow \quad \mathit{Person}(\mathbf{person}(\mathtt{pid})) \wedge \mathit{name}(\mathbf{person}(\mathtt{pid}), \mathtt{name})$

- Mapping assertion to populate the *makes* role with all pairs consisting of an order and the corresponding person who made the order:

  `SELECT OD.id as oid, S.id as pid FROM OrderData OD, Stakeholders S`
  `WHERE OD.from = S.id`

  $\rightsquigarrow \quad \mathit{makes}(\mathbf{person}(\mathtt{pid}), \mathbf{order}(\mathtt{oid}))$

OHub Case Study: The Rewriting of Queries

- Let's consider the query $q(x) = Person(x)$ that retrieves all persons
- Since $\mathcal{T}$ contains $\exists makes \sqsubseteq Person$ and $\exists name \sqsubseteq Person$, the rewriting of $q(x)$ w.r.t. $\mathcal{T}$ gives us the UCQ $q_r(x) = Person(x) \vee \exists y.makes(x, y) \vee \exists z.name(x, z)$,
- The unfolding of $q_r(x)$ w.r.t. $\mathcal{M}$ gives us the following SQL query $q_u(x)$:

```
SELECT id as x FROM Stakeholders WHERE type = 'p'
    UNION
SELECT S.id as x FROM OrderData OD, Stakeholders S
    WHERE OD.from = S.id
```

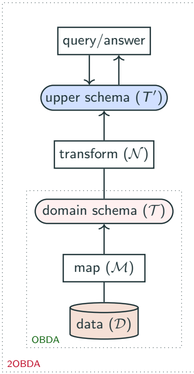
OHub Case Study: Computing Certain Answers

- The transformation rule $Person(x) \rightsquigarrow TargetCustomer(\mathbf{tc}(x))$ maps each instance of the domain schema concept *Person* into the upper schema concept *TargetCustomer*.
- By applying the 5 steps of computing certain answers, we get the OBDA mapping $q_u(x) \rightsquigarrow TargetCustomer(\mathbf{tc}(x))$, where $q_u(x)$ is the following SQL query

```
SELECT id as x FROM Stakeholders WHERE type = ‘p’
    UNION
SELECT S.id as x FROM OrderData OD, Stakeholders S
    WHERE OD.from = S.id
```

Achievements

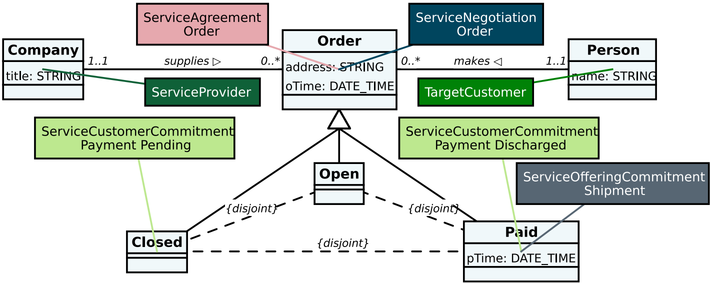
- Modularity and separation of concerns
    - If the underlying data storage changes, only the mapping to the domain schema needs to be updated, without touching the definition of commitments
    - If instead the contract is updated, the domain-to-upper schema transformation needs to change accordingly, without touching the OBDA specification
- The approach is driven by the actual querying requirements
    - only the aspects of the upper schema that are relevant for querying need to be subject of transformation rules
- The transformation rules also provide a way to customize the view over the data
    - even with the same upper schema, two different sets of transformation rules might provide different views over the data represented by the domain schema
- We might even go beyond that, and consider situations where several upper schemas are provided, each with different sets of transformation rules



Specifying Schema Transformations ($\mathcal{N}$)

- An approach based on annotations are used to generate the rules
- UML class diagrams is employed as a concrete language for conceptual data modeling, and we rely on their logic-based encoding in terms of OWL 2 QL [2, 7]
- We assume to work with OWL 2 QL compliant ontologies, the available types of annotations are automatically deduced from the upper ontology based on this assumption
- We have developed an editor for annotating the domain ontology with upper ontology concepts that dynamically builds the annotation types accordingly

Specification Schema Transformation ($\mathcal{N}$)



Payment pending service customer commitment

Payment discharged service customer commitment

Shipment service offering commitment

Tool Support

- We provide onprom tool-chain[1] that supports the various phases of the 2OBDA design
- It implements the automated processing technique for annotations and consists of the following components
  - A UML Editor to model the domain and upper ontologies as UML class diagrams, and to import from and export to OWL 2 QL
  - A Dynamic Annotation Editor to enrich the domain ontology with annotations extracted from the upper ontology, which are automatically translated into corresponding SPARQL queries
  - A Transformation Rule Generator automatically processes the annotations, and generates rules between the domain and upper ontologies
    - implements the described mapping synthesis technique leveraging the state-of-the-art ontop[2] framework [3] for mapping management and query rewriting and unfolding
- We do not have native tool support for specifying the mapping assertions currently, it can be realized manually or by exploiting third-party tools (such as mapping editor in the ontop plugin for Protégé[3])

[1] http://onprom.inf.unibz.it

[2] http://ontop.inf.unibz.it

[3] http://protege.stanford.edu/

Conclusions

- A framework is proposed for accessing data through different conceptual schemas, which is formalized in terms of 2OBDA
- It is possible to exploit an existing OBDA specification for the domain schema, together with conceptual mappings between the domain and the upper schema, to automatically derive an new OBDA specification for the upper schema
- The framework can be realized through schema annotations, and accordingly we implemented a tool-chain supporting annotation based approach
- Finally, 2OBDA framework and the results can be easily generalized to *multiple-levels*, where schema transformations are specified between multiple conceptual schemas

Questions and Thanks

Web Site

Please visit for more information, related papers, to download onprom and to watch screencasts:

http://onprom.inf.unibz.it

Acknowledgement

This research is supported by the Euregio IPN12 *KAOS (Knowledge-Aware Operational Support)* project, funded by the "European Region Tyrol-South Tyrol-Trentino" (EGTC), and by the UNIBZ internal project *OnProm (ONtology-driven PROcess Mining)*.

References

References i

- G. Xiao, D. Calvanese, R. Kontchakov, D. Lembo, A. Poggi, R. Rosati, and M. Zakharyaschev, "Ontology-based data access: A survey," in *Proc. of IJCAI*, AAAI Press, 2018.
- D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodríguez-Muro, and R. Rosati, "Ontologies and databases: The *DL-Lite* approach," in *RW Tutorial Lectures* (S. Tessaris and E. Franconi, eds.), vol. 5689 of *LNCS*, pp. 255–356, Springer, 2009.
- D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro, and G. Xiao, "Ontop: Answering SPARQL queries over relational databases," *Semantic Web J.*, vol. 8, no. 3, pp. 471–487, 2017.
- J. C. Nardi, R. de Almeida Falbo, J. P. A. Almeida, G. Guizzardi, L. F. Pires, M. J. van Sinderen, N. Guarino, and C. M. Fonseca, "A commitment-based reference ontology for services," *Information Systems*, vol. 54, pp. 263 – 288, 2015.
- A. Scherp, C. Saathoff, T. Franz, and S. Staab, "Designing core ontologies," *Applied Ontology*, vol. 6, no. 3, pp. 177–221, 2011.

- G. Guizzardi, "On ontology, ontologies, conceptualizations, modeling languages, and (meta)models," in *Proc. of DB&IS*, pp. 18–39, IOS Press, 2006.
- D. Calvanese, T. E. Kalayci, M. Montali, and A. Santoso, "OBDA for log extraction in process mining," in *RW Tutorial Lectures*, vol. 10370 of *LNCS*, pp. 292–345, Springer, 2017.
- W. van der Aalst *et al.*, "Process mining manifesto," in *Proc. of the BPM Int. Workshops* (F. Daniel, K. Barkaoui, and S. Dustdar, eds.), vol. 99 of *LNBIP*, pp. 169–194, Springer, 2012.
- D. Calvanese, T. E. Kalayci, M. Montali, and S. Tinella, "Ontology-based data access for extracting event logs from legacy data: The onprom tool and methodology," in *Proc. of BIS*, vol. 288 of *LNBIP*, pp. 220–236, Springer, 2017.
- D. Calvanese, T. E. Kalayci, M. Montali, and A. Santoso, "The onprom toolchain for extracting business process logs using ontology-based data access," in *Proc. of the BPM Demo Track and BPM Dissertation Award, co-located with BPM 2017*, vol. 1920 of *CEUR*, 2017.

- IEEE Computational Intelligence Society, "IEEE standard for eXtensible Event Stream (XES) for achieving interoperability in event logs and event streams," Std 1849-2016, IEEE, 2016.
- M. Montali, D. Calvanese, and G. De Giacomo, "Verification of data-aware commitment-based multiagent systems," in *Proc. of AAMAS*, pp. 157–164, 2014.
- J. Euzenat and P. Shvaiko, *Ontology Matching*.
  Springer, 2nd ed., 2013.
- M. Lenzerini, "Data integration: A theoretical perspective.," in *Proc. of PODS*, 2002.
- D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati, "Tractable reasoning and efficient query answering in description logics: The DL-Lite family," *JAR*, vol. 39, no. 3, 2007.
- C. Daraio *et al.*, "The advantages of an ontology-based data management approach: Openness, interoperability and data quality," *Scientometrics*, vol. 108, no. 1, pp. 441–455, 2016.

- G. Mehdi *et al.*, "Semantic rule-based equipment diagnostics," in *Proc. of ISWC*, vol. 10588 of *LNCS*, pp. 314–333, Springer, 2017.
- A. Calì, D. Calvanese, G. De Giacomo, and M. Lenzerini, "On the expressive power of data integration systems," in *Proc. of ER*, vol. 2503 of *LNCS*, pp. 338–350, Springer, 2002.
- T. Catarci and M. Lenzerini, "Representing and using interschema knowledge in cooperative information systems," *JICIS*, vol. 2, no. 4, pp. 375–398, 1993.
- E. Kharlamov *et al.*, "Ontology based data access in Statoil," *J. of Web Semantics*, vol. 44, 2017.
- B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue, and C. Lutz, "OWL 2 Web Ontology Language profiles (second edition)," W3C Recommendation, W3C, Dec. 2012.
- F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, eds., *The Description Logic Handbook: Theory, Implementation and Applications*. **CUP, 2003.**

- A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev, "The *DL-Lite* family and relations," *JAIR*, vol. 36, pp. 1–69, 2009.
- A. K. Chopra and M. P. Singh, "Custard: Computing norm states over information stores," in *Proc. of AAMAS*, pp. 1096–1105, 2016.
- A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati, "Linking data to ontologies," *J. on Data Semantics*, vol. X, pp. 133–173, 2008.