Data Governance on EA Information Assets: Logical Reasoning for Derived Data

Bernhard Waltl, Thomas Reschenhofer, and Florian Matthes

Software Engineering for Business Information Systems
Department of Informatics
Technische Universität München
Boltzmannstr. 3, 85748 Garching bei München, Germany
b.waltl@tum.de, reschenh@in.tum.de, matthes@in.tum.de

Abstract. Today's companies face increased pressure regarding compliance to legal obligations. Regulations for the financial sector such as Basel II and III, Solvency II, or the Sarbanes-Oxley-Act explicitly demand various requirements. Many of those requirements address the governance and management of information assets, such as data. Companies need to report and track their information architecture, and furthermore have to provide accountability and responsibility information on their data to, e.g., supervisory authorities.
Additionally, the tracking of processed data becomes increasingly difficult since the software systems and their interactions throughout the enterprise are highly complex. This paper argues for a consistent and comprehensive assignment mechanism on data governance roles. Based on logical inferences, we are able to show how accountability and responsibility can be assigned throughout processed data. Thereby, we analyze the limitations of traditional logic, such as propositional logic, and exemplarily show how non-monotonic defeasible logic can be used to keep the assignment of roles on information assets consistent.

Keywords: Data Governance, Enterprise Architecture, EA Metrics, Logical Reasoning, Defeasible Logic

1 Introduction

The increasing organizational complexity of today's enterprises requires a strategic approach to align the enterprise's IT to its business [1]. In this sense, an Enterprise Architecture (EA) represents the holistic organization of an enterprise consisting of its components, relations, and environment [2]. Thereby, EA is understood as a descriptive and holistic model of the enterprise [3]. The corresponding management discipline—Enterprise Architecture Management (EAM)—is a function for planning, developing, and controlling an EA and its evolution, and thus to ensure its flexibility, efficiency, and transparency. Consequently, EAM fosters business IT alignment [4] and potentially improves the enterprise's business performance [1]. To cope with the increasing size and complexity of an EA, there are various tools supporting enterprise architects by providing methods for gathering, modeling, and analyzing EA data [5]. Analyzing an EA also includes the computation of EA metrics allowing a reliable assessment of the current state as well as the evolution of the EA [6]. Thereby, EA metrics generate derived EA data [7].

Due to EAM's goal to establish a shared understanding of the enterprise's current state among the whole organization, it involves a variety of stakeholders, e.g., enterprise architects, EA repository managers, and data owners [8]. However, there is an increasing pressure to legal compliance forcing companies to ensure transparency and provenance of their data. In the financial industry regulatory frameworks like Basel II [9], respectively Basel III, Solvency II and the Sarbanes-Oxley Act [10] were promulgated to ensure proper risk management and to reduce the vulnerability to systemic risks. Although the concrete measures vary throughout the different regulations, they share common principles regarding transparency and provenance of data. Additionally, the German implementation of Basel II in the banking act [11] has a strong focus on auditing of processes, data, business entities, etc. [12, 13]. In this sense, specifying and documenting roles, rights and responsibilities of all stakeholders is indispensable for today's companies.

There are various frameworks to document the roles of stakeholders with regard to specific entities of an EA. For example, the RACI matrix [14] links a stakeholder to certain activities or processes by specifying this person to be responsible (R), accountable (A), consulted (C), or informed(I) regarding the respective process. On another note, the Data Governance Framework (DGF) addresses responsibilities and accountability of stakeholders on data itself. Therefore, there are frameworks providing a way of explicitly specifying which persons are responsible and accountable for processes and data in EA. However, it is still an open question whether these frameworks also fit to derived EA data generated by EA metrics. Furthermore, since the data itself is derived through EA metrics, the question arises if roles for derived EA data are also derivable from the metric and its input. Trends in data science, such as big data, really push the necessity to provide consistent frameworks of semi-automatic determination of responsibility and provenance of data. Who is responsible, respectively accountable, for data that was automatically created by algorithms? Based on existing information about the assignment of roles of either processes and data, algorithms based on logical frameworks can deduce the roles for the derived properties. A consistent logical framework helps to generate reproducible and reliable results [15].

Consequently, answering the following research questions constitutes this work's contribution: What is a framework for the specification of roles on EA data governance? And what is a logical framework to infer roles on derived EA data based on the corresponding EA metric and its input?

This paper describes an approach to cope with the assignment and derivation of roles on data, representing information about an enterprise architecture. Thereby, we briefly sketch prior and related work in Section 2. Section 3 will give an overview about models and roles for relevant information assets (e.g., data, metrics, and derived data) in the EA domain. The paper continues with the adaption of an existing and well studied logical framework, namely the defeasible logic, in Section 4. Finally, Section 5 summarizes its contribution and sketches further research directions.

2 Related Work

One of the most common tools for documenting the roles of persons on activities is the RACI (Responsible, Accountable, Consulted, Informed) matrix [16]. The RACI matrix was already applied in software development governance [16], but also in the domain of EAM. Thereby, Fischer et al. [8] used the RACI matrix for the EA maintenance process, i.e., they specify roles for EA stakeholders on activities related to the maintenance of the EA model. However, they did not discuss roles for derived EA information.

How to govern enterprise data was discussed by the Data Governance Institute (DGI). The DGI proposed the Data Governance Framework (DGF), which is a "logical structure for classifying, organizing, and communicating complex activities involved in making decisions about and taking action on enterprise data" [17]. The framework argues for an extensive system analysis to trace unclear accountabilities along the data flow. This manual tracing might work out for small programs with well defined data flows, but exceeds its applicability in complex enterprise structures.

As an extension to the DGF, Khatri and Brown [18] analyzed different facets of "data as an asset" with a strong focus on the provision of a framework for data governance. They also admit that data assets are moving more and more in the focus of legislative compliance and related reporting. The outcome of a case study they conducted in the insurance industry is a differentiation between five interrelated decision domains for data governance, namely data principles, data quality, meta-data, data access, and data life-cycle. For each of the provided domains the driving decisions are stated out and potential roles are given.

Yogesh et al. [19] argue that data provenance is crucial for the reuse of data. Thereby, data tracking can be ensured using meta-data, pertaining the complete derivation history starting from its original source. Analyzing the scientific workflows, they created a taxonomy of data provenance characteristics. The taxonomy differentiates between five main headings, namely application of provenance, subject of provenance, representation of provenance, provenance storage, and provenance dissemination. The differentiations are discussed regarding their characteristics, but miss a normative guideline how to derive the provenance information for data from its sources.

The usage of logical frameworks to enrich existing frameworks with consistent logical conclusions was also subject of prior research. Ninghui et al. [15] developed a logic-based language, namely delegation language, with the objective to represent policies, credentials, and requests allowing inferences based on logical conclusions. Their research focused on the support of decentralized aggregation of data, in which the derivation and inference on data plays a significant role.

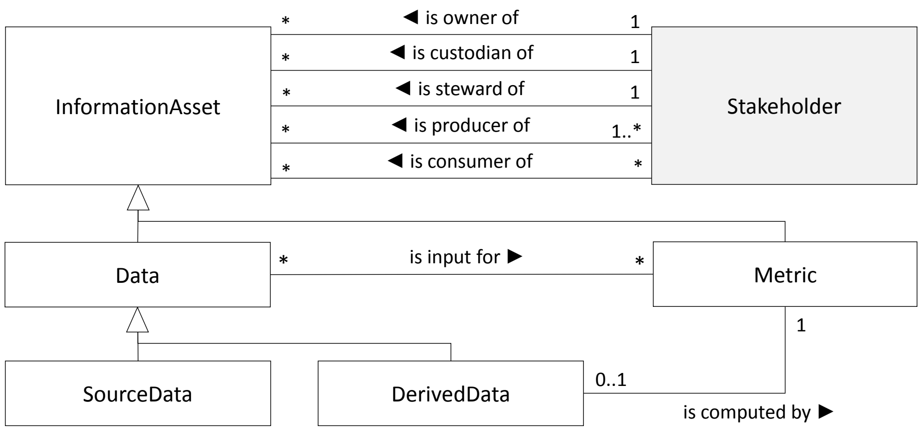

Fig. 1. A conceptual model of EA information assets including EA data, metrics, and derived data. Each information asset is associated with stakeholders through specific roles (see Table 1).

The possibility to express complex policies allows a fine granular declaration and delegation of data authority. The delegation language only supports monotonic reasoning, which was also identified as drawback by the authors. They later on worked on a non-monotonic delegation logic to handle with conflicting policies, but the manuscript was never officially published [20].

Gaaloul et al. [21] used a logical framework to reason about delegation events to model task delegation. The usage of formal logical engines, such as event calculus, allows the definition of delegation policies. Their overall objective is the automated assignment of delegation policies. Once the delegation policies are specified, a discrete event calculus reasoner can efficiently solve the problem by transforming it into a satisfiability (SAT) problem. They showed that formal reasoning increases the compliance regarding delegation changes in existing policies.

3 A Model of Roles for EA Data and Metrics

In the context of this work, we define derived EA data as the output of EA metrics, which performs a computation based on input data. Thereby, we differentiate between source data which has to be provided by one or more EA stakeholders, and derived data which is computed by an EA metric based on respective input data. In this sense, not only data itself is considered to be an information asset, but also the metric capturing the actual computation prescription. Figure 1 gives an overview of the information assets, and their relationships. In this context, a metric has an arbitrary amount of input data, and produces a derived datum. The corresponding roles are described bellow in Table 1.

For example, Schneider et al. [22] describe the metric *Average Functional Scope* computing the average number of function points of business applications

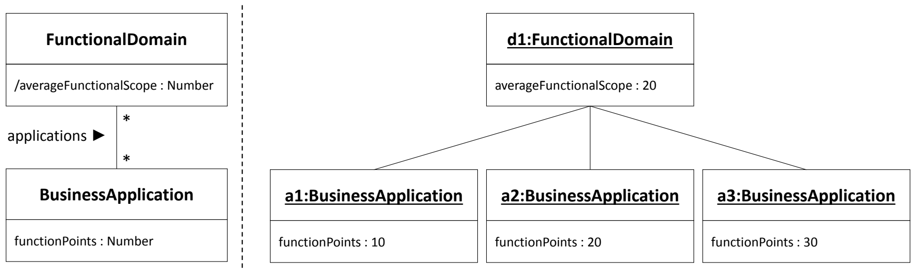

Fig. 2. An excerpt of the information model for the metric *Average Functional Scope* as described by Schneider et al. [22], as well as a corresponding and exemplary UML object diagram.

within a certain business domain. In this sense, function points are a quantification of how much business functionality a business application provides. An excerpt of the corresponding information model as well as a proper exemplary object diagram is depicted in Figure 2. A formal computation prescription for this metric can be expressed with the Object Constraint Language (OCL) as follows:

```
context FunctionalDomain::averageFunctionalScope:Number
derive:
  self.applications -> collect(functionPoints) -> sum()
  / self.applications -> size()
```

Based on the exemplary object diagram in Figure 2, the evaluation of the metric *Average Functional Scope* for the functional domain *d1* generates a derived datum, namely the value 20 for the attribute *averageFunctionalScope*.

To relate the exemplary information model in Figure 2 to the conceptual model of information assets as depicted in Figure 1, we consider the attribute values in Figure 2 to be objects of class *Data* as defined in the conceptual model. Furthermore, also whole entities are information assets. More specifically, the function points of business applications *a1*, *a2*, and *a3* are *SourceData*, and the derived attribute's value is *DerivedData*. The entities themselves (business applications *a1*, *a2*, and *a3* as well as functional domain *d1*) are *InformationAssets*. Moreover, the metric *Average Functional Scope* with its computation prescription is an object of class *Metric*. While this is a very concrete example of a model-based metric, we could use any other metric which is defined based on an information model.

Depending on the level of granularity of data management and governance, various roles have to be specified to govern *InformationAsset*s (see Figure 1). According to Kathri and Brown [18], five interrelated decision domains are common (see Section 2), whereas the "data principles" domain is responsible for the linkage with the business. Consequently, the data principles domain clarifies the extent to "which data is an enterprise-wide asset, and thus what specific policies, standards and guidelines are appropriate." [18]. In order to achieve clarification, a differentiation according to roles for accountability for data is necessary. The differentiation as used in our research was derived from Khatri and Brown is provided in Table 1.

Role	**Description**
Data owner	The data owner has to ensure the data quality and develop and implement the data definition. Furthermore, he is responsible for interpreting and ensuring compliance to Federal, State and other policies.
Data custodian	The data custodian ensures that the access to the data is authorized and controlled. He is responsible for safe custody, transport, storage and implementation of business rules.
Data steward	The data steward ensures that each data element has a clear and unambiguous definition. He also has a respective documentation on usage.
Data producer	Everyone that creates data elements and persists them is a data producer. This is not necessarily a person but can also be an application.
Data consumer	The data consumer is the opposite role to the data producer. Data consumers read, transform, or process existing data elements.

Table 1. Roles for accountability and responsibility of information assets [18].

In the domain of EAM, Matthes et al. [23] proposed two different roles on metrics, namely the owner and consumer. As proposed by Kathri and Brown [18] and by considering metrics as information assets, we extend this existing set of two roles on metrics to enable comprehensive data governance regarding information assets. Thereby, we are adding the roles of the data custodian, data steward and data consumer to metrics. As shown in Figure 1, data as well as metrics can now be subsumed as information assets, which also holds the required relationships (data governance roles expressed by associations) between stakeholders and the information asset.

4 Defeasible Derivation of Roles for Derived EA Data

Just as Ninghui et al. [20] used a "delegation logic" to derive policies (see Section 2), we argue now for an existing logic system, namely "defeasible logic".

4.1 An Introduction to Defeasible Logic

Defeasible logic has been investigated extensively and is well known in the domain of artificial intelligence, especially its usage as argumentation logic [24]. Nute has shown the advantage of the defeasible logic over other logic systems [25]. The main advantage is the capability of non-monotonic reasoning. Monotonic logic systems fail if new information and conclusions contradicting prior reasoning results are added. Due to contradictions the logic framework gets inconsistent. This inconsistency does no longer allow the derivation of true and reliable results and can lead to undecidable problems for first order logic [25]. Therefore, the defeasible logic differentiates between different types of rules. Since it is not necessary to introduce and discuss all the possible rules and their relationships, we just restrict ourselves to the two most important:

$A \rightarrow \phi$ **(strict rule)** ... Strict rules can never be defeated. They do not have exceptions and consequently it is a *necessary* connection between antecedent (A) and consequence (ϕ), e.g., "Penguins are birds".

$A \Rightarrow \phi$ **(defeasible rule)** ... Defeasible rules represent weaker connections that can be defeated by a strict rule or a defeasible rule, e.g., "Birds can fly".

Without introducing all formalism required to fully understand the defeasible logic system, we will just briefly sketch out the symbols and predicates that we are using:

Predicate: P(X,Y) ... The predicate specifies the properties of some entity. For instance if someone wants to set the data owner of data entity d to the person p this can be denoted as dataOwner(d,p).

Logical conjunction: , ... The comma is an abbreviation for the logical conjunction (AND, &, $\wedge$). E.g., "isData(d), isData(d')" represents the fact that both d *and* d' are data entities.

Logical consequence: $\Sigma \vdash \phi$ ... The symbol represents the consequence relation. Informally spoken does this mean, that based on the given set of facts and rules Σ the logical system allows to derive the information ϕ. To express which information should be derived, it is possible to write a predicate, specifying the queried information.

Contradiction: $\perp$ ... To represent a contradiction in a logical system, for example inferred by conflicting rules or facts, the $\perp$ symbol (bottom) can represent this. This is usually a most unwanted state, since the logic system does not longer allow true and proofed inferences.

Inference: $\rightarrow$**,** $\Rightarrow$ ... To enable the derivation of new information, existing information has to be combined in rules. The two rule types are described above.

The applicability of the logic system is now discussed by a small example. Firstly, several facts are defined. In this case, there are two persons, namely p and q, and there are two data elements, namely d and d', whereas d' is derived data from d. Furthermore, the role of the data owner of d is kept by person p. We can now define the defeasible rule, that if a person is the data owner of some data element, and there is some other data element, that was derived from this data, then this person automatically becomes the data owner of the derived data element, i.e., dataOwner(d', X) $\vdash$ X = p. If someone now adds an additional fact, e.g., the data owner of d' is explicitly set to person q, then traditional logic systems, such as propositional logic systems, would determine a contradiction, which consequently would cause inconsistency and therefore the end of the logic engine as is. This inconsistency does not arise in defeasible logic, since the implications that can be drawn using defeasible rules, are "soft" and can be overwritten by other rules and facts. Consequently, person q is the data owner of d', i. e. dataOwner(d', X) $\vdash$ X = q. This example can be formally expressed as follows:

Facts:	isPerson(p), isPerson(q), isData(d), isData(d'), isDerivedData(d', d), dataOwner(d, p)
Def. rule:	isDerivedData(D, S), dataOwner(D,X) $\Rightarrow$ dataOwner(D, X)
Query 1:	dataOwner(d', X) $\vdash$ X = p
Fact:	dataOwner(d', q) ($\vdash \perp$, in traditional logic systems)
Query 2:	dataOwner(d', X) $\vdash$ X = q

To avoid problems of decidability, defeasible rules can and should be enriched with priorities clarifying the precedence between defeasible rules (see [25]). According to Nute, it is reasonable to prioritize regarding the specificity of rules. Thereby, one possible assignment could be *lex specialis derogat legi generali*. This means, that the more specific rule (*lex specialis*) takes precedence over the more general rule (*lex generali*). But there are also other ways to assign priorities to rules. Nute argues, just as legal sciences does in some cases, for the *lex superior derogat legi inferiori*, that higher-ranked rules (*lex superior*), such as the federal law, should have a higher priority than lower-ranked rules (*lex posterior*), such as state law. Different other priority assignments would be possible, depending on the concrete use case and implementation. Recent implementations of defeasible logic, e.g. Spindle [26], allow to explicitly assign priorities to express the conflict solution between rules.

4.2 Defeasible Derivation of Accountability and Responsibility Roles on Information Assets

As described in Section 3, every information asset needs to have roles governing the accountability on various levels. Although the number of roles, their name and their description may vary between enterprises, the need for assignment throughout dependent information assets remains. In the following we exemplarily show the applicability of defeasible logic as consistent and comprehensive method to assign roles with dependencies and resolve contradicting assignments.

Combining the roles as defined by Khatri and Brown (see Table 1), and non-monotonic reasoning of defeasible logic as described by Nute (see Section 4.1), it is possible to extend this reasoning to a comprehensive and consistent role derivation framework. Based on the assumption, that information assets of an enterprise, namely data, which can be differentiated into source data and derived data, and EA metrics, need to have a complete and unambiguous assignment of accountability and responsibility roles, we can model a situation as shown in Figure 3.

The model visualizes the situation as already described in Figure 2. We have three different business application entities, *a1*, *a2*, and *a3*, which have respective

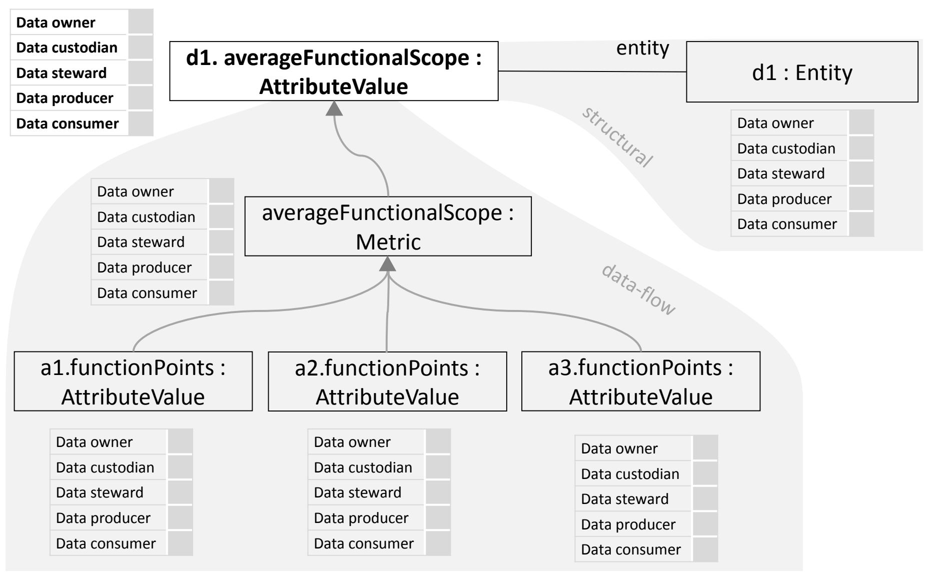
Data owner
Data custodian
Data steward
Data producer
Data consumer
d1. averageFunctionalScope : AttributeValue
entity
d1 : Entity
structural
averageFunctionalScope : Metric
data-flow
a1.functionPoints : AttributeValue
a2.functionPoints : AttributeValue
a3.functionPoints : AttributeValue

Fig. 3. Structural and data-flow influence factors of an attribute value.

function points as attribute values. The function points of those business applications are aggregated by an EA metric, which solely measures the mean value. The result of the metric is the value of the attribute *averageFunctionalScope* of the entity *d1* and consequently an information asset. Therefore, the holistic perspective on this particular attribute shows that it has two influence factors, i.e. data-flow and structure (see Figure 3). The data-flow arises through metric and their calculation, whereas the structural influence factor is given through the data model, since every attribute value belongs to an entity. Both, the entity and the attribute definition are considered to be information assets and therefore have well-defined roles.

The situation as described has some obvious challenges. The derived attribute, namely averageFunctionalScope, is an information asset itself and is also part of other information assets. As an information asset it is necessary to have well-defined roles regarding accountability and responsibility. Hereby three determination scenarios are possible:

1. The roles are derived from the entity to which the derived attribute belongs.
2. The roles are derived from the data-flow influence factors, namely the data sources and metric from which it was calculated.
3. The roles are manually assigned for each information asset.

Which of the three scenarios is going to be applied depends on various factors and the enterprise's data governance strategy. It could also be a combination of all of the mentioned strategies, depending on the situation and data. However, using defeasible logic it is possible to specify the derivation rules accordingly. The possibility to set priorities between roles, as explained in Section 4.1, can be used to create a logically consistent and comprehensive data governance derivation framework. The combination of strict and defeasible rules is then the key to set up a data governance framework that allows logical inference regarding roles, i.e. responsibilities. Although it is in principle possible to create contradictions within the logical system, by using strict rules and facts, this can and should be circumvented by the priority mechanism of defeasible rules, which is the fundamental idea of this non-monotonic reasoning.

The example bellow shows a possible scenario how defeasible logic can be used to update data governance principles, facts and interconnections between the data governance roles of information assets. At the beginning, facts are provided. Based on the facts, *Query 0* cannot determine who is the data owner of the attribute value *d1.avgFctScope*. The first rule allows the derivation of data ownership from the containing entity. The data owner from the entity is passed to the attribute values of the entity. *Query 1* now results in p as the data owner of *d1.avgFctScope*, because p is also the data owner of *d1*. However, a succeeding defeasible rule can now overrule this derivation. Based on the input values, the data owner of derived data may be determined. The facts define, that *d1.avgFctScope* is derived from *a1.functionPoints* and *a2.functionPoints*. So the data owner of the input variables is passed towards the derived data, which is then q (see *Query 2*). The metric, i.e. the combination of the input variables, might be more important than the source data, the defeasible rule 3 now allows the derivation of the data owner. The data owner of *d1.avgFctScope* will then be the same as the data owner of the metric *avgFctScope*, which is r (see Query 3).

Facts:	isEntity(d1), isEntity(a1), isEntity(a2), isMetric(avgFctScope), isData(d1.avgFctScope), isData(a1.functionPoints), isData(a2.functionPoints), isDerivedData(d1.avgFctScope, a1.functionPoints), isDerivedData(d1.avgFctScope, a2.functionPoints), isPerson(p), isPerson(q), isPerson(r), dataOwner(d1, p), dataOwner(a1, q), dataOwner(a2, q), dataOwner(avgFctScope, r)
Query 0:	dataOwner(d1.avgFctScope, X) $\vdash$ X = *undecided*
Def. rule 1:	isEntity(E), dataOwner(E,X), isDataOfEntity(D,E) $\Rightarrow$ dataOwner(D, X)
Query 1:	dataOwner(d1.avgFctScope, X) $\vdash$ X = p
Def. rule 2:	isDerivedData(D', D), dataOwner(D,X) $\Rightarrow$ dataOwner(D', X)
Query 2:	dataOwner(d1.avgFctScope, X) $\vdash$ X = q
Def. rule 3:	isMetric(M), dataOwner(M, X), calculates(M, D) $\Rightarrow$ dataOwner(D, X)
Query 3:	dataOwner(d1.avgFctScope, X) $\vdash$ X = r

This example shows the applicability of non-monotonic reasoning in the data governance process of EA information assets. The focus was rather the provision of a methodology than focusing on concrete rules, since those may vary throughout enterprises. However, the defeasible logic allows continuous adaption of rules, without being inconsistent, i.e. contradicting, at any time.

5 Conclusion

This paper is an attempt to support data governance in enterprises by using non-monotonic logic, i.e. defeasible logic. We developed a model for information assets in enterprise architectures and identified existing roles for accountability and responsibility. The synthesis with a respective meta-model that was used in prior research leads to a model on information and governance structure. The assignments of five different governance roles to each information asset assures tracking and well-defined accountability but can lead to a managerial overhead due to the amount of assignment.

The interconnectedness of data in enterprises and the derivation of data by metrics calls for a consistent framework that ensures the derivation of data governance roles. In this paper we argue for an existing logic, namely defeasible logic. Thereby it is not only possible to provide rules for derivation but also to continuously adapt the existing set of rules. In contrary to other logic frameworks, existing rules can be overwritten and priorities between rules can be specified. This allows seamless adaptation to changed data governance policies.

This paper proposes a framework for roles on EA data governance and shows how a defeasible logic can be used to support data governance, and thus answers the research questions as raised in Section 1. Open research questions could address the role of time in data governance principles, but could also focus on a more practical research such as using a defeasible logic engine to simulate and analyze the impact of changes in data governance policies.

References

1. F. Ahlemann, E. Stettiner, M. Messerschmidt, and C. Legner, *Strategic Enterprise Architecture Management.* Springer-Verlag, 2012.
2. International Organization for Standardization, "ISO/IEC 42010:2007 Systems and Software Engineering – Recommended Practice for Architectural Description of Software-Intensive Systems," Geneva, Switzerland, 2007.
3. J. A. Zachman, "A Framework for Information Systems Architecture," *IBM Systems Journal*, 1987.
4. S. Buckl, F. Matthes, I. Monahov, S. Roth, C. Schulz, and C. M. Schweda, "Towards an Agile Design of the Enterprise Architecture Management Function," *Proceedings of the Enterprise Distributed Object Computing Conference Workshops*, 2011.
5. F. Matthes, S. Buckl, J. Leitel, and C. M. Schweda, "Enterprise Architecture Management Tool Survey 2008: Technical Report," 2008.
6. I. Monahov, T. Reschenhofer, and F. Matthes, "Design and Prototypical Implementation of a Language Empowering Business Users to Define Key Performance Indicators for Enterprise Architecture Management," *Proceedings of the Trends in Enterprise Architecture Research Workshop*, 2013.
7. T. Reschenhofer, I. Monahov, and F. Matthes, "Type-Safety in EA Model Analysis," *Proceedings of the Trends in Enterprise Architecture Research Workshop*, 2014.

8. R. Fischer, S. Aier, and R. Winter, “A Federated Approach to Enterprise Architecture Model Maintenance,” *Enterprise Modelling and Information Systems Architectures*, vol. 2, no. 2, pp. 14–22, 2007.
9. Basel Committee on Banking Supervision, “International Convergence of Capital Measurement and Capital Standards: A Revised Framework Comprehensive Version,” June 2006.
10. P. Aleatrati, M. Hauder, and S. Roth, “Impact of Solvency II on the Enterprise Architecture of Insurances - A Qualitative Study in Germany,” *Multikonferenz Wirtschaftsinformatik (MKWI 2014)*, 2014.
11. D. Bundesbank, “Banking Act: 2009.”
12. J. Bretz, *Prüfung IT im Fokus von MaRisk und Bundesbank: Verstärkter IT-Fokus in Sonderprüfungen.* Finanz Colloquium Heidelberg, 2012.
13. B. Waltl, A. W. Schneider, and F. Matthes, “Deriving and Modelling Compliance Requirements from Legal Audits,” *23rd Annual EICAR Conference: Trust and Transparency in IT Security*, 2014.
14. J. E. Hallows, *The Project Management Office Toolkit.* New York: American Management, 2002.
15. L. Ninghui, N. G. Benjamin, and F. Joan, “Delegation logic: A logic-based approach to distributed authorization,” *ACM Trans. Inf. Syst. Secur.*, vol. 6, no. 1, pp. 128–171, 2003.
16. A. Kofman and T. Klinger, “Roles, Rights, and Responsibilities: Better Governance Through Decision Rights Automation,” *Proceedings of the Workshop on Software Development Governance*, 2009.
17. Data Governance Institute, “The DGI Data Governance Framework,” 2009.
18. V. Khatri and C. V. Brown, “Designing data governance,” *Commun. ACM*, vol. 53, no. 1, pp. 148–152, 2010.
19. L. S. Yogesh, P. Beth, and G. Dennis, “A survey of data provenance in e-science,” *SIGMOD Rec*, vol. 34, no. 3, pp. 31–36, 2005.
20. N. Li, B. N. Grosof, and J. Feigenbaum, “A nonmonotonic delegation logic with prioritized conflict handling,” Technical Report, 2000.
21. K. Gaaloul, H. A. Proper, E. Zahoor, F. Charoy, and C. Godart, “A logical framework for reasoning about delegation policies in workflow management systems,” *International Journal of Information and Computer Security*, vol. 4, no. 4, p. 365, 2011.
22. A. W. Schneider, T. Reschenhofer, A. Schütz, and F. Matthes, “Empirical Results for Application Landscape Complexity,” *Proceedings of the Hawaii International Conference on System Sciences*, 2015.
23. F. Matthes, I. Monahov, A. W. Schneider, and C. Schulz, “Towards a Unified and Configurable Structure for EA Management KPIs,” *Proceedings of the Trends in Enterprise Architecture Research Workshop*, 2012.
24. K. D. Ashley, “Toward a computational theory of arguing with precedents,” in *Proceedings of the 2nd international conference on Artificial intelligence and law.* Vancouver, British Columbia, Canada: ACM, 1989, pp. 93–102.
25. D. Nute, “Defeasible Logic,” in *Web Knowledge Management and Decision Support*, ser. Lecture Notes in Computer Science, O. Bartenstein, U. Geske, M. Hannebauer, and O. Yoshie, Eds. Springer Berlin Heidelberg, 2003, vol. 2543, pp. 151–169. [Online]. Available: http://dx.doi.org/10.1007/3-540-36524-9_13
26. H.-P. Lam and G. Governatori, “The Making of SPINdle,” in *Rule Interchange and Applications*, ser. Lecture Notes in Computer Science, G. Governatori, J. Hall, and A. Paschke, Eds. Springer Berlin Heidelberg, 2009, vol. 5858, pp. 315–322. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-04985-9_29