Applying the Concept of Knowledge Blockchains to Ontologies

Hans-Georg Fill
Digitalization and Information Systems Group
Department Informatics - University of Fribourg
Boulevard de Pérolles 90, 1700 Fribourg, Switzerland

Abstract

In this position paper we would like to incite a discussion on how the concept of Knowledge Blockchains can be applied to ontologies. Knowledge Blockchains revert to blockchain technologies for enabling a transparent monitoring of knowledge evolution, for tracking the provenance of knowledge, for establishing delegation schemes, and for ensuring the existence of patterns in formal conceptualizations using zero-knowledge proofs. Based on their original application to enterprise models, we discuss which benefits arise from using the concept for ontologies. The paper concludes by outlining further research in this direction.

Introduction

The use of blockchains is currently discussed for many application fields (Iansiti and Lakhani 2017). Based on the success of Bitcoin and Ethereum, initiatives for investigating potential use cases have been launched in industry as well as in academia. Although the underlying technologies have been available for quite some time, the combination of a decentralized, tamper-proof storage together with trustworthy and equally decentralized consensus mechanisms for transactions has the potential to realize new forms of collaboration and business models (Aste, Tasca, and Di Matteo 2017). Whereas public blockchains, which do not regulate the access to the stored information, have contributed to the prominence of the technology, many use cases found in industry today focus on so-called permissioned blockchains, e.g. (Androulaki et al. 2018). For this type of blockchains, the aspect of decentralized storage and automated consensus mechanisms is maintained whereas the access to the blockchain is restricted to authenticated users. Permissioned blockchains thus enable interactions between actors who do not fully trust each other but still pursue common goals. Due to the required identification of the participants, their transactions can be traced back to physical persons thus easing also legal compliance requirements such as Know-Your-Customer (KYC) and Anti-Money-Laundering (AML) principles (Möser, Böhme, and Breuker 2013).

In a recent publication it has been discussed how these blockchain technologies can be applied to the domain of enterprise modeling (Fill and Härer 2018). Thereby, the core idea was to store the knowledge that has been made explicit in the form of visual conceptual models on a decentralized blockchain. The goals of this approach denoted as *Knowledge Blockchains* are as follows. It shall permit to track who has contributed which changes in the models and at what time and how concepts in the models have thus evolved. Further, the approach permits to establish delegation schemes so that operations on models can be delegated to other identities. Finally, the use of zero-knowledge proofs allows to proof the existence of patterns in models without having to disclose the content of the models, which is desirable for sensitive information. In the following this idea is extended to ontologies, which may be regarded as one type of enterprise models that stands for a shared, agreed-upon, formal, and machine-interpretable conceptualization of a domain (Fill 2017; Studer, Benjamins, and Fensel 1998).

The remainder of the paper is organized as follows. At first, the concept of Knowledge Blockchains is briefly described. Subsequently, we investigate, which of the components used for Knowledge Blockchains have already been discussed in the context of ontologies. Next, we illustrate how Knowledge Blockchains could be applied to ontologies and derive finally opportunities for further research activities.

Knowledge Blockchains

The concept of Knowledge Blockchain has been first presented in a recent publication by (Fill and Härer 2018). The main goal of Knowledge Blockchains is to store and process the knowledge that is made explicit in the form of various types of enterprise models using blockchain technologies. Enterprise models in this context are understood as schema-based information structures that are typically represented in a visual format and are specified in a semi-formal or formal manner (Bork and Fill 2014). Through the decentralized nature of blockchains, knowledge can thus be easily distributed. Based on the digital signatures used for signing information on a blockchain, it can be further traced who has contributed what to the enterprise models and at what time. In the case of permissioned blockchains this can be restricted to authenticated users and model information may

even be encrypted to prevent unauthorized access.

For storing information about the status of models, Knowledge Blockchains use a Merkle-tree-based structure together with the files that contain the model information in a format that can be interpreted by modeling tools. For this purpose, the attribute values of every model entity are hashed. Furthermore, all model entities are assigned a *universally unique identifier (UUID)* which is also hashed. The UUID permits to identify any model element that is created independently from other elements in decentrally stored copies of the blockchain. The concatentation of the two resulting hashes is hashed again and the resulting hash further concatenated with hashes resulting from other model entities and hashed until arriving at one single hash value, i.e. the *Merkle root*. The use of Merkle trees permits on the one hand to easily identify any changes that have occurred in the models. At the same time it enables the execution of so-called *zero-knowledge proofs*. Thereby, it can be proven that certain information parts are contained in an enterprise model without revealing the content of the model. To accomplish this, the information parts to be searched for in the model have to be hashed in the same way as described before. Subsequently, the hash values can be compared with the hash values in the Merkle tree of the models.

In addition to the model information, Knowledge Blockchains also store information for *permission models*. These specify the rights for creating, modifying or deleting model information based on the digital signatures of actors contributing to the Knowledge Blockchain. In addition, they permit to delegate these rights to other actors. Permission models are hashed in the same way as described above.

The block header in Knowledge Blockchains then contains essentially the Merkle root hashes for the enterprise models and the permission models, the model and permission models themeselves, a timestamp, and the header signature. In the case of non-permissioned blockchains, a nonce can be added to the header as well. In the course of the mining it is checked whether the permissions specified in the previous permission model permit the intended operations. The approach has been realized as a prototype using the ADOxx metamodeling platform (Fill and Karagiannis 2013). For details on the mining algorithm and the implementation we refer to (Fill and Härer 2018).

Related Work

In the following we review approaches in the area of ontologies that are similar to the concepts used for Knowledge Blockchains. This will be followed by a discussion on how ontologies may be represented in Knowledge Blockchains.

Due to their nature as shared conceptualizations, ontologies are typically created in multi-user environments. This has traditionally been accomplished using platforms such as Stanford Protégé or ContentCVS that permit the collaborative editing of ontologies and the tracking of changes (Tudorache et al. 2008; Jiménez-Ruiz et al. 2009). In the according change records, the metadata on ontology changes is stored. This includes for example information about which user performed a change, a timestamp, or the entities on which the changes were performed (Walk et al. 2015). Today, this is accomplished using web-based environments, which eases the technical realization of multi-user collaboration platforms (Horridge et al. 2018). The availability of the data about changes in ontologies is essential for analyzing their evolution and deriving according strategies, e.g. for validating changes and their impacts (Zablith et al. 2013).

The use of digital signatures for signing ontologies has been discussed for RDF graphs, e.g. (Carroll 2003). Thereby, it can be verified who has created or modified a certain RDF document, which is essential to trace its provenance. Closely related to this is the hashing and subsequent signing of XML documents for determining which parts of an XML document have been updated (Maruyama, Tamura, and Uramoto 2000; Bartel et al. 2013). This approach may also be extended by using one-way hash functions and Merkle trees for applying zero-knowledge proofs to XML documents (Devanbu et al. 2001). Thereby, the existence of information in XML documents can be proven without revealing their content. The same procedures could be used for many ontology formats that are based on XML.

Another direction for the unique identification of resources has been proposed in (Kuhn and Dumontier 2014). Here, cryptographic hash values are included in unique resource identifiers (URIs) for enabling the identification and verification of resources or parts thereof on the web. As the cryptographic hash value is directly added to the URI, it does not require any additional data structures but can be easily processed. This approach thus does not regard XML documents or ontologies as a whole but is able to identify single resources.

In the context of Semantic Web, the use of blockchains has been proposed by Iancu and Sandu to realize the *trust layer* in the traditional semantic web stack (Iancu and Sandu 2016). However, in their paper they just store the entire ontology information as a file on the Openchain blockchain without making use of the typical data structures used for blockchains in the form of hash trees. Similarly, an approach for semantic internet of things reverts to the Hyperledger infrastructure for storing ontology information on a blockchain (Ruta et al. 2017). By extending the underlying APIs, semantic matchmaking and reasoning could thus be added to a blockchain-based application.

One of the most recent developments concerning the integration of ontologies and blockchains is the proposal of *GraphChain* (Sopek et al. 2018). In this work, the authors propose the creation of a linked chain of RDF graphs based on a computation of RDF digests with SHA-256 hash functions. The RDF digests are then stored in triple stores and the changes broadcasted to other nodes. Although the data structure strongly resembles the one in other blockchain approaches, consensus mechanisms or chain update strategies were not discussed.

In summary, previous approaches have already discussed the integration of ontologies and technologies necessary for blockchains. This concerns in particular approaches for hashing and signing RDF graphs and the identification of ontology resources. What is missing so far is the use of blockchain mechanisms such as consensus protocols re-

spectively mining algorithms for automatically determining whether concepts should be added. Furthermore, the multi-user-oriented design of ontologies and zero-knowledge proofs has so far not been yet considered in this context.

Ontologies and Knowledge Blockchains

Based on the insights gained in the previous two sections it will now be discussed how the concept of Knowledge Blockchains could contribute to the domain of ontologies. For this purpose, it first needs to be answered how ontologies can be represented as models according to the approach of Knowledge Blockchains. As shown in the upper part of Figure 1, Knowledge Blockchains extend the meta-metamodel constructs *model type*, *class*, and *relationclass* with UUID attributes.

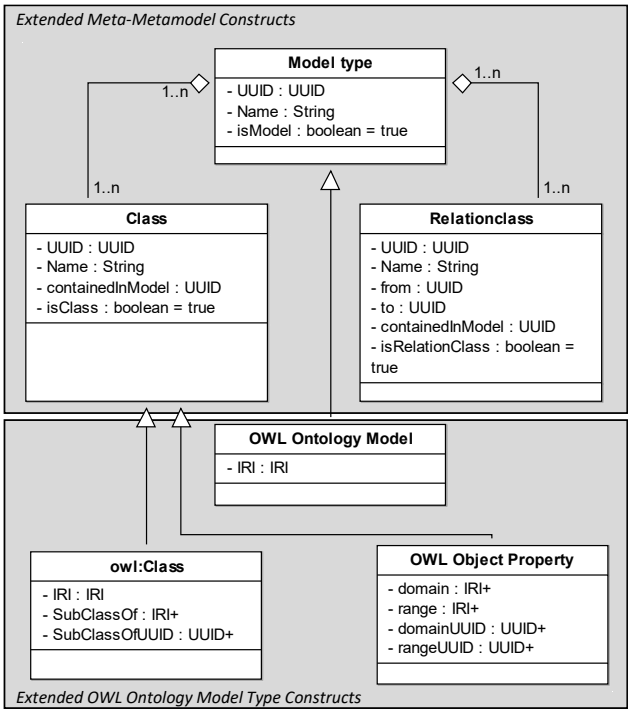

Figure 1: Representing Ontologies as Models in the Knowledge Blockchain Approach

From these meta-metamodel constructs we can derive the elements for OWL ontologies as shown exemplarily in the lower part of the figure. The representation of OWL ontologies in this way has been described earlier in more detail, e.g. (Fill 2017). However, as Knowledge Blockchains make use of UUIDs for identifying elements, the according reference attributes required in OWL ontologies have to be extended in this way as well. This is illustrated in the figure for the *SubClassOf* as well as the *domain* and *range* attributes. Instead of just using lists of IRIs (denoted as IRI+), Knowledge Blockchains would require lists of UUIDs in addition to take into account that several versions may exist for an element with the same IRI, e.g. due to an evolution of a class.

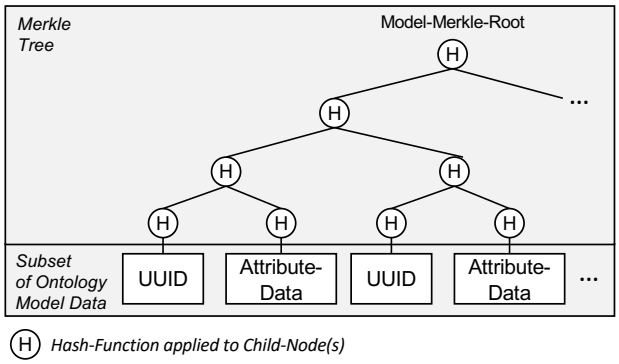

Figure 2: Creation of the Merkle Tree in Knowledge Blockchains

With the thus extended attributes of the ontology model elements, the hash values for the UUID of every element - as inherited from the meta-metamodel - and the hash value of the attribute data can be represented in a Merkle tree as shown in Figure 2. This will be the basis for conducting zero-knowledge proofs on the contents of the ontology.

Based on these data structures we will regard four areas of Knowledge Blockchains that could be beneficial for ontologies and that are not yet covered by previous approaches. These are: the monitoring of the evolution of ontologies and the tracking of the provenance of concepts, the use of permission and delegation schemes for the distributed design of ontologies, and the use of zero-knowledge proofs.

Monitoring Ontology Evolution and Tracking the Provenance of Concepts

Due to their nature as immutable and distributed ledgers, blockchains seem well suited to act as a foundation for monitoring the evolution of ontologies and for tracking the provenance of the contained concepts. With the approach of Knowledge Blockchains, this can be accomplished in the folllowing ways. As every proposal for a modification in a Knowledge Blockchain must be *digitally signed*, it can be tracked who added which change and at what time. In case of qualified electronic signatures[1], these even refer to actual physical persons including all legal responsibilities. By using UUIDs for identifying elements in the ontology models, even different versions of the same IRIs may be stored so that alternative proposals for the realization of concepts can be recorded. At the same time, the UUIDs permit to unambiguously specify a particular version of a conceptualization and to track over time in the blockchain whether this version has been changed. For this purpose, the Merkle trees allow for an efficient identification of changes in ontology models based on the comparison of the hash values.

The decentralized nature of a blockchain further requires to establish so-called *mining algorithms* that decide upon the

[1]See for example the eIDAS regulation in the European Union: `https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32014R0910&from=EN#d1e791-73-1`.

inclusion of new blocks. Here, the mining algorithm originally proposed for Knowledge Blockchains could be extended for the case of ontologies to conduct certain sanity checks on the ontology before adding a block, e.g. to filter out changes that could lead to an inherent or inconsistent ontology or that do not satify certain domain or application-specific constraints, cf. (Zablith et al. 2013).

Establishing Permission and Delegation Schemes

In contrast to other approaches for aligning blockchains and ontologies, Knowledge Blockchains offer a mechanism for specifying *who* has *which* kind of access to which *parts* of a model - for details on these *permission models* we refer to (Fill and Härer 2018). This means, it can be defined who can edit which parts of an ontology as well as who is allowed to delegate rights to other persons. In this way, a *chain of trust* between different actors can be established. This could subsequently be aligned with previous approaches for defining ontologies that rely on multiple parties for deciding about the inclusion of concepts, e.g. (Vrandečić et al. 2005).

Using Zero-Knowledge Proofs

Zero-Knowledge proofs are typically used in blockchains for efficiently verifying the existence of information or transactions in a blockchain without having to reveal the actual data. For example, in cryptocurrency blockchains, this mechanism may be used to check that a particular transaction has been accomplished and is thus part of the current branch of the blockchain. The actual transaction data does however not need to be disclosed.

In Knowledge Blockchains, zero-knowledge proofs can be used to verify that certain patterns exist in models without having to give away the actual model data. This may be similarly applied to ontologies, e.g. to prove to an external actor that a confidential ontology contains certain elements without having to disclose the ontology. A use case for this could be to ensure the compliance of an ontology to legal regulations in a domain, e.g. that classes describing persons actually do require the specification of a social security number.

http://www.unifr.ch/ : OWL Ontology Model

IRI : IRI = http://www.unifr.ch/
UUID : UUID = e3874776-3398-42ec-bdf7-4c4cc6f2f646
Name : String = http://www.unifr.ch/
isModel : boolean = true

Animal : owl:Class

IRI : IRI = http://www.unifr.ch/#Animal
SubClassOf : IRI+ = http://www.unifr.ch/#Thing
SubClassOfUUID : UUID+ = 77965e01-3aef-490b-8875-5760d28659a9
UUID : UUID = 7e381016-810a-49c5-aacf-662353843940
Name : String = Animal
containedInModel : UUID = e3874776-3398-42ec-bdf7-4c4cc6f2f646
isClass : boolean = true

Figure 3: Excerpt of a Sample Ontology Model Using the Extended Knowledge Blockchain Constructs

For illustrating the basic working of this mechanism we present example instances of an OWL ontology model in Figure 3. Thereby, each element contains a *UUID* attribute for its unique identification in addition to its *IRI* attribute. The OWL class element *Animal:owlClass* further contains a *SubClassOf* reference to another class *Thing*, which is expressed both in IRI and UUID style. In Figure 4 it is shown how the hashing of these elements takes place. It is exemplarily shown for the lowest layer of the Merkle tree and two attributes *IRI* and *SubClassOf*. In every case, the UUID of the element is taken as the left leaf in the Merkle tree and the attribute's name and its value as the right leaf. This is a slight extension to the original conception for Knowledge Blockchains as it permits to access single attribute values in the proofs later. Each of the two leaf values is then hashed using the secure one-way SHA256 hashing algorithm.

With this structure, it can now for example be proven that an element with the IRI *http://www.unifr.ch/#Animal* is contained in the blockchain data just by giving access to the hashes in the Merkle tree without the underlying data. This is accomplished by calculating the SHA256 hash value for '*IRI:http://www.unifr.ch/#Animal*' and then searching for it on the lowest level of the hash tree. Subsequently, the hash of the UUID can be found, which may be used for further queries, e.g. to prove that the same element is a subclass of *http://www.unifr.ch/#Thing*.

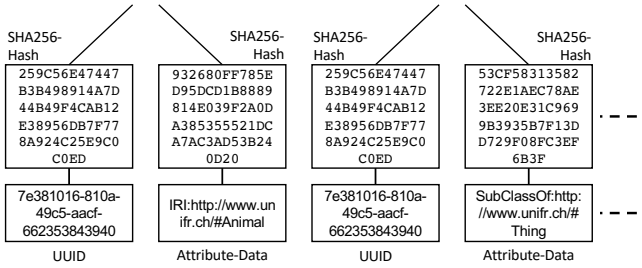

Figure 4: Excerpt of a Merkle Tree for the Sample Ontology Model

Opportunities for Further Research

Although this position paper only intends to incite a discussion on using the approach of Knowledge Blockchains for ontologies, we can derive several opportunities for further research. First, it needs to be investigated in detail which data structures are most adequate for the representation of ontologies in the context of blockchains, cf. (Fill and Johannsen 2016). This is closely related to the application of zero-knowledge proofs and which advantages can be gained from their application. In this respect, additional benefits may arise from a combination of zero-knowledge proofs and reasoning, e.g. to automatically expand the scope of matches when searching for a concept in a Merkle tree based on information derived throuh reasoning.

Second, the use of UUIDs may not be an optimal solution for ontologies although they provide several benefits in terms of a distributed and thus independent creation of elements. For this purpose it would be a next step to evaluate whether the approach described by (Kuhn and Dumontier

2014) for *Trusty URIs* could be used instead.

Third, for realizing the application of Knowledge Blockchains to ontologies, existing editors either from the area of ontologies or conceptual modeling would need to be adapted. In particular, it needs to be evaluated how the requirements for a multi-user-based editing of ontologies can be optimally aligned with the concept of permission models. In the current conception of permission models only the level of model elements is considered. As a lot of essential information in ontologies is stored in the level of attributes, it may need to be taken into account to expand the permission and delegation specification to attributes.

References

Androulaki, E.; Barger, A.; Bortnikov, V.; Cachin, C.; Christidis, K.; Caro, A. D.; Enyeart, D.; Ferris, C.; Laventman, G.; Manevich, Y.; Muralidharan, S.; Murthy, C.; Nguyen, B.; Sethi, M.; Singh, G.; Smith, K.; Sorniotti, A.; Stathakopoulou, C.; Cocco, S. W.; and Yellick, J. 2018. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of 13th EuroSys Conf.*, 1–15. ACM.

Aste, T.; Tasca, P.; and Di Matteo, T. 2017. Blockchain Technologies: The Foreseeable Impact on Society and Industry. *IEEE Computer* (September):18–28.

Bartel, M.; Boyer, J.; Fox, B.; LaMacchia, B.; and Simon, E. 2013. XML Signature Syntax and Processing Version 1.1. Report, W3C Recommendation. https://www.w3.org/TR/xmldsig-core/ (accessed 2018-11-02).

Bork, D., and Fill, H.-G. 2014. Formal Aspects of Enterprise Modeling Methods: A Comparison Framework. In *47th Hawaii International Conference on System Sciences*, 3400–3409. IEEE.

Carroll, J. 2003. Signing RDF Graphs. In Fensel, D.; Sycara, K.; and Mylopoulos, J., eds., *International Semantic Web Conference*, volume LNCS 2870, 369–384. Springer.

Devanbu, P.; Gertz, M.; Kwong, A.; Martel, C.; Nuckolls, G.; and Stubblebine, S. 2001. Flexible authentication of xml documents. In *8th ACM conference on Computer and Communications Security*, 136–145. ACM.

Fill, H.-G., and Härer, F. 2018. Knowledge Blockchains: Applying Blockchain Technologies to Enterprise Modeling. In *51st Hawaiian International Conference on System Sciences*, 4045–4054. AIS.

Fill, H.-G., and Johannsen, F. 2016. A Knowledge Perspective on Big Data by Joining Enterprise Modeling and Data Analyses. In *IEEE HICSS'2016*. IEEE.

Fill, H.-G., and Karagiannis, D. 2013. On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform. *Enterprise Modelling and Information Systems Architecture* 8(1):4–25.

Fill, H.-G. 2017. SeMFIS: A Flexible Engineering Platform for Semantic Annotations of Conceptual Models. *Semantic Web* 8(5):747–763.

Horridge, M.; Goncalves, R.; Nyulas, C.; Tudorache, T.; and Musen, M. A. 2018. WebProtégé 3.0 Collaborative OWL Ontology Engineering in the Cloud. In Van Erp, M.; Atre, M.; Lopez, V.; Srinivas, K.; and Fortuna, C., eds., *ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks*, volume Vol-2180. CEUR-WS.

Iancu, B., and Sandu, C. 2016. A Cryptographic Approach for Implementing Semantic Webs Trust Layer. In Bica, I., and Reyhanitabar, R., eds., *Int. Conf. for Information Technology and Communications*, 127–136. Springer.

Iansiti, M., and Lakhani, K. 2017. The truth about blockchain. *Harvard Business Review* (January-February):119–127.

Jiménez-Ruiz, E.; Grau, B. C.; Horrocks, I.; and Llavori, R. B. 2009. Building ontologies collaboratively using contentcvs. *Description Logics* 477.

Kuhn, T., and Dumontier, M. 2014. Trusty URIs: Verifiable, Immutable, and Permanent Digital Artifacts for Linked Data. In Presutti, V.; D'Amato, C.; Gandon, F.; D'Aquin, M.; Staab, S.; and Tordai, A., eds., *European Semantic Web Conference*, volume LNCS 8465, 395–410. Springer.

Maruyama, H.; Tamura, K.; and Uramoto, N. 2000. Digest Values for DOM (DOMHASH). Report, Internet Engineering Task Force (IETF). https://www.ietf.org/rfc/rfc2803.txt (accessed 2018-10-08).

Möser, M.; Böhme, R.; and Breuker, D. 2013. An inquiry into money laundering tools in the Bitcoin ecosystem. In *APWG eCrime Researchers Summit*. IEEE.

Ruta, M.; Scioscia, F.; Ieva, S.; Capurso, G.; and Di Sciascio, E. 2017. Semantic Blockchain to Improve Scalability in the Internet of Things. *Open Journal of Internet of Things* 3(1):46–61.

Sopek, M.; Gradzki, P.; Kosowski, W.; Kuzinski, D.; Trojczak, R.; and Trypuz, R. 2018. GraphChain A Distributed Database with Explicit Semantics and Chained RDF Graphs. In *World Wide Web Conference - Workshop on Linked Data & Distributed Ledgers*, 1171–1178. ACM.

Studer, R.; Benjamins, R.; and Fensel, D. 1998. Knowledge Engineering: Principles and methods. *Data & Knowledge Engineering* 25:161–197.

Tudorache, T.; Noy, N.; Tu, S.; and Musen, M. A. 2008. Supporting Collaborative Ontology Development in Protégé. In *International Semantic Web Conference*, volume LNCS 5318. Springer.

Vrandečić, D.; Pinto, S.; Tempich, C.; and Sure, Y. 2005. The diligent knowledge processes. *Journal of Knowledge Management* 9(5):85–96.

Walk, S.; Singer, P.; Noboa, L.; Tudorache, T.; Musen, M. A.; and Strohmaier, M. 2015. Understanding How Users Edit Ontologies: Comparing Hypotheses About Four Real-World Projects. In *International Semantic Web Conference*, volume LNCS 9366, 551–568. Springer.

Zablith, F.; Antoniou, G.; d'Aquin, M.; Flouris, G.; Kondylakis, H.; Motta, E.; Plexousakis, D.; and Sabou, M. 2013. Ontology evolution: a process-centric survey. *The knowledge engineering review* 30(1):45–75.