A Process for Reverse Engineering of Aggregate-Oriented NoSQL Databases with Emphasis on Geographic Data

Angelo Augusto Frozza[1,2], **Ronaldo dos Santos Mello**[1] **(advisor)**

[1]PPGCC-INE - Federal University of Santa Catarina (UFSC)
Florianópolis – SC – Brazil

[2]Federal Institute Catarinense (IFC)
Camboriú – SC – Brasil

angelo.frozza@ifc.edu.br, r.mello@ufsc.br

Level : Doctorate **Qualification Exam** : June 2018
Admission : March 2016 **Defense Forecast** : February 2020

Completed Steps : Bibliographic review, Progress Seminar, Schemas extraction for document-oriented NoSQL DBs, Qualification exam.
Future Steps : Schema extraction for key-value and columnar NoSQL DBs, mapping JSON Schemas to RDF and validation.

Resumo. *Bancos de dados (BD) NoSQL não têm esquemas ou permitem esquemas flexíveis. Esta característica é adequada para vários domínios de aplicações, tais como, redes sociais, aplicações Web e Internet das Coisas. Porém, é crescente o interesse em manipular esquemas de BD NoSQL, p.ex. para tarefas como integração de BD. Este trabalho propõe um processo para extração de esquemas de BD NoSQL orientados a agregados (com possíveis dados geográficos) e a produção de visões RDF a partir dos esquemas criados. Pelo nosso conhecimento, não há trabalhos com uma abordagem tão abrangente ou que tratem dados NoSQL geográficos, demonstrando que ainda há muito a ser feito nessa área. A partir da análise dos trabalhos relacionados, apresentamos a nossa proposta e alguns resultados preliminares.*

Abstract. *One main advantage of NoSQL DBs is having no schemas, or allowing flexible ones. Such feature is suitable for various application domains, including social networks, Web applications, and Internet of Things. However, there is a growing interest in the management of NoSQL DB schemas, e.g., for tasks such as DB integration. This work proposes a process for the schema extracting from aggregate-oriented NoSQL DB (with possible geographic data), and the production of RDF views from the created JSON Schemas. To our knowledge, there are no related works that propose such a comprehensive approach and that deal with geographic NoSQL data. We analyze ten related works published in the last years, which present different proposals for NoSQL schema extraction, but few of them focus on the schema integration, demonstrating that there is still much to be done in this area. From this analysis, we present our proposal and some preliminary results.*

Keywords*: NoSQL. Data Reverse Engineering. Schema Extraction. Geographic data.*

1. Introduction

The *Big Data* market explosion has led large companies to demand databases (DB) that can handle large data volumes effectively. In this context, traditional Relational Databases (RDB) present several limitations, e.g., they prioritize strong consistency. The NoSQL DBs have emerged to deal with these RDB limitations [Cattell 2011]. They, generally, provide eventual data consistency, a robust availability and elasticity capabilities. A common feature of NoSQL DBs is that they are *schemaless*, i.e., they allow the storage of data without prior knowledge of their structure [Sadalage and Fowler 2013]. However, the lack of information regarding the schema makes difficult to perform several data processing tasks, such as data integration, data retrieval, validation, and analysis [Kapsammer et al. 2012]. Moreover, have information about the data scheme is very useful for application development. For example, several applications, such as *Foursquare*, retrieve data in JSON format but do not define a schema, being difficult for users to query such data because they are unaware of the documents structures. In short, a schema is useful because allows us to understand the data structure of a dataset.

Based on this motivation, the objective of this work is propose a process for the Reverse Engineering of NoSQL DBs that supports the aggregate-oriented data model, i.e., NoSQL DBs based on key-value, document-oriented and columnar data models. Our process presents several highlights, among them, the handling of complex data types (typical of NoSQL DBs) and of spatial data. The expected contributions are: (*i*) a process for generating aggregate-oriented NoSQL DB schemas; (*ii*) a canonical model based on JSON Schema recommendation and a process of mapping aggregate-oriented NoSQL schemas to the recommendation, considering an extension of the JSON Schema to represent geographic properties; (*iii*) a method for merge distinct schemas (represented in JSON Schema) from NoSQL DBs with heterogeneous aggregate-oriented data models and with possible geographical properties (*e.g.* point, line, polygon, multipoint, multiline, multipolygon, and geometry); (*iv*) generation and persistence of semantic visions in the RDF (Resource Definition Language)[1] format from the mapping of unified JSON Schemas, aiming to collaborate with applications that wish to make queries to multiple NoSQL DBs with possible geographic data.

The rest of this paper is organized as follows: Session 2 presents the problem statement and relevance; Section 3 presents the process proposed in this Thesis; Section 4 presents the preliminary results obtained with the extraction of NoSQL DB schemas, and Section 5 describes the future activities to be carried out to complete this Thesis.

2. Problem statement and relevance

When the database paradigm changes, new processes and tools need to be developed. So it was in the transition to RDB in the 1980s and 1990s; then, in the 2000s, with the growth in the use of XML data; and now with the NoSQL DBs.

In the NoSQL DBs context, the lack of schemas or the use of flexible schemas to define the data structure offers greater ease in dealing with scalability and increasing availability. However, this feature becomes a challenge for the data processing tasks because of the lack of homogeneity that it presents, which does not occur with RDBs. Another feature of schemaless DBs is to facilitate both the registration of complex and non-uniform

[1]https://www.w3.org/RDF/

data as the evolution of the data [Ruiz et al. 2015]. In general, common NoSQL data can be represented in JavaScript Object Notation (JSON)[2] format, while spatial data can be represented in GeoJSON[3] format. Both formats are recent industry standards. Although GeoJSON is a suitable standard for storing geographic data, in NoSQL this data type can be stored in several other formats, such as text, GML, KML, WKT, among others. Identifying geographic data stored in other data formats, different of GeoJSON, represents another challenge to be addressed in this research.

Despite these facilities, its increasingly perceived that a growing interest in the management of explicit NoSQL schemas is emerging [Klettke et al. 2015, Ruiz et al. 2015, Karpov 2017, Ruckstieß 2017]. However, there is not a standard for representing schemas for JSON data. The JSON Schema[4] is a vocabulary that is under discussion and is heading to become the default schema definition for JSON documents.

Most of the works analyzed (Table 1) aim to create tools for manipulate and manage schemas [Izquierdo and Cabot 2013, Kapsammer et al. 2012, Klettke et al. 2015, Mesiti and Valtolina 2014, Ruiz et al. 2015, Wang et al. 2015, Baazizi et al. 2017]. Two papers emphasize the use of JSON data in RDBs [Discala and Abadi 2016, Liu et al. 2016]. Only four papers presented a complete Reverse Engineering process. They extract schemas from several NoSQL datasets and produce an integrated conceptual schema [Kapsammer et al. 2012, Izquierdo and Cabot 2013, Kiran and Vijayakumar 2014, Mesiti and Valtolina 2014].

The works of [Kapsammer et al. 2012] and [Izquierdo and Cabot 2013] use a similar approach to identify schemas implicit in JSON documents coming from social networking APIs and Web applications. The schemas obtained are integrated to generate a domain schema for the application. The two works emphasize the extraction of schemas from JSON documents obtained through calls to application APIs and products a schema represented in the ECORE model[5]. In this Thesis, we intend to extract the schemas of NoSQL DBs with heterogeneous data models, unifying them in an RDF schema.

BigLoader [Mesiti and Valtolina 2014] is a theoretical design that aims to support the user in data loading activities in NoSQL DBs. The work proposed by [Kiran and Vijayakumar 2014] extracts schemas from tables stored in HBase that are used to create local ontologies in OWL (Web Ontology Language)[6]. These are used to form a global ontology, which can be used to perform searches from a SPARQL endpoint. We consider a limitation of this work that the schema extraction process only is applied to HBase. Our approach is more suitable, running the Reverse Engineering of any NoSQL DB that supports an aggregate data model.

Some works use an intermediate data model different from the model used to represent the final schema. This intermediate model is generally used in the initial stages of the process to represent the structure of a single NoSQL data instance. A hierarchical structure, used to represent the structure of nested JSON document objects, is proposed by [Klettke et al. 2015, Wang et al. 2015, Discala and Abadi 2016]. Works

[2]http://json.org/
[3]http://geojson.org/
[4]http://json-schema.org/
[5]https://www.eclipse.org/modeling/emf/
[6]https://www.w3.org/OWL/

Table 1. Comparing the works of Reverse Engineering from NoSQL DBs

Paper	**Data Entry (Source)**	**Extraction or Integration Strategy**	**Intermediate Model**	**Data Types**	**Ver-sions**	**Inte-gration**	**Stan-dards**	**Output**
Kapsammer et al. 2012	**JSON** (social networks APIs)	MDE	**JSON Schema**	**Yes**	**Yes**	**Yes ***	**Yes**	ECORE *Schema*
Izquierdo et al. 2013	**JSON** (web services APIs)	MDE	JSON meta-model	**Yes**	**Yes**	**Yes ***	**Yes**	ECORE domain schema
Kiran et al. 2014	*HBase*	Hierarchical Summarization	*HBase*	No	**Yes**	**Yes**	**Yes**	**OWL Ontology**
Mesiti et al. 2014	**JSON** and others	Hierarchical Summarization	–x–	N/I	No	**Yes**	N/I	Key-value Schema
Klettke et al. 2015	**JSON** (MongoDB dataset)	Hierarchical Summarization	Structure Identification Graph (SG)	**Yes**	**Yes**	No	**Yes**	JSON Schema
Ruiz et al. 2015	**JSON** (MongoDB, CouchDB e HBase)	MDE	JSON meta-model	**Yes**	**Yes ****	No	**Yes**	NoSQL schema metamodel
Wang et al. 2015	**JSON** (Datasets)	Hierarchical Summarization	–x–	No	**Yes**	No	No	*eSiBu-Tree*
Discala et al. 2016	**JSON** (Dataset JSON or CSV)	Machine Learning	Directed graph	**Yes**	No	No	No	Relational Schema
Liu et al. 2016	**JSON** (Oracle JSON column)	Hierarchical Summarization	–x–	**Yes**	No	No	No	JSON DataGuide
Baazizi et al. 2017	**JSON** (Dataset)	Hierarchical Summarization	–x–	**Yes**	No	No	No	Proprietary Key-value Schema
This Thesis	**JSON (aggregate-oriented NoSQL)**	**Hierarchical Summarization and MDE**	**JSON Schema**	**Yes**	**Yes**	**Yes**	**Yes**	**RDF**

* Integration of schemas obtained from Web service APIs.
* Only work that produces versioned schemas.

based on MDE [Izquierdo and Cabot 2013, Ruiz et al. 2015] use ECORE metamodels. Works that cite the use of JSON schemas generally adopt *key-data_type* constructs. Only [Kapsammer et al. 2012] use the JSON Schema recommendation as an intermediate model. We propose use JSON Schema to store the schemas found in a NoSQL dataset.

Notice that some works are limited to analyze the structure of JSON documents, not doing any special treatment about the data types of each atomic element in the JSON document. We can also notice that just few approaches uses standards recommended by organizations such as W3C[7] to represent the final schema produced in the Reverse Engineering process. In this Thesis, we consider that standards such as JSON, GeoJSON, JSON Schema and RDF can contribute to solve the Reverse Engineering problem and also simplify the work by not requiring the developer to define new technologies or tools.

Regarding the final representation of the schema, there is also no consensus. The works that use the MDE approach present an application domain schema as an ECORE metamodel [Kapsammer et al. 2012, Izquierdo and Cabot 2013, Ruiz et al. 2015]. Other works like *eSiBu-Tree* [Wang et al. 2015] use their own structure. While [Baazizi et al. 2017] proposes the own schema language, based on JSON Schema, [Kiran and Vijayakumar 2014] adopts OWL as the final model of schema representation. Only [Klettke et al. 2015] use the JSON Schema recommendation. We propose to use RDF to represent an data schema and a catalog containing the mappings between the local JSON Schema and the schema view in RDF.

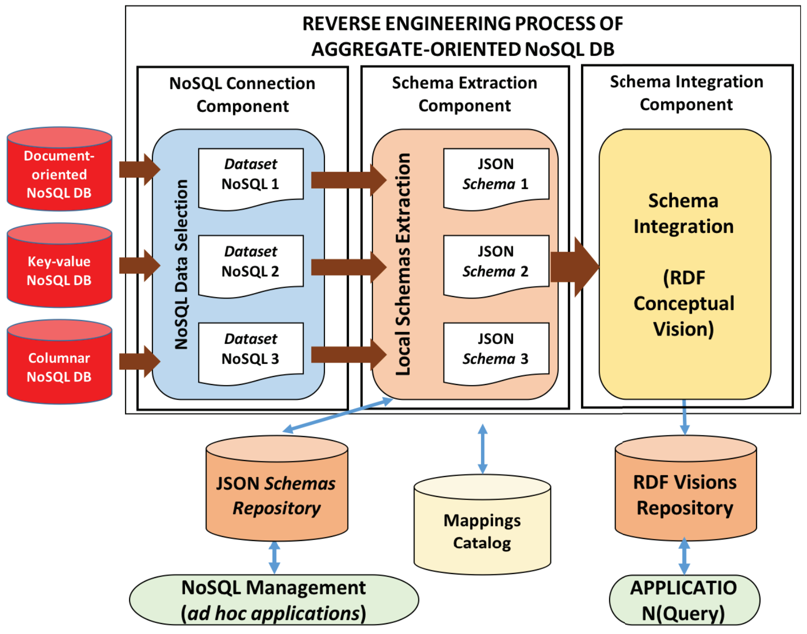
[7]https://www.w3.org/

Finally, we do not found any work emphasizing the spatial data Reverse Engineering in NoSQL DBs, this being another point of originality of this Thesis.

3. Proposed NoSQL Schema Extraction Methodology

Figure 1 presents an overview of the methodology for Reverse Engineering of NoSQL DBs proposed in this Thesis. It can serve as a basis for the construction of data integration frameworks running between NoSQL DBs and the applications.

Figure 1. Process for Reverse Engineering of aggregate-oriented NoSQL DBs


REVERSE ENGINEERING PROCESS OF AGGREGATE-ORIENTED NoSQL DB
NoSQL Connection Component
Schema Extraction Component
Schema Integration Component
Document-oriented NoSQL DB
Key-value NoSQL DB
Columnar NoSQL DB
NoSQL Data Selection
Dataset NoSQL 1
Dataset NoSQL 2
Dataset NoSQL 3
Local Schemas Extraction
JSON Schema 1
JSON Schema 2
JSON Schema 3
Schema Integration (RDF Conceptual Vision)
JSON Schemas Repository
Mappings Catalog
RDF Visions Repository
NoSQL Management (ad hoc applications)
APPLICATIO N(Query)

The Reverse Engineering process starts with the connection to the NoSQL DBs through the *NoSQL Connection Component*. This component can select the entire DB or a sample (filtered) data. NoSQL data is loaded considering the data model of each NoSQL DB. If geographic data are present, they can be loaded in GeoJSON or converted to this format if they are in another similar spatial data format.

The connection component delivers pre-selected NoSQL dataset to the *Schema Extraction Component*. This component parses the input NoSQL data to produce local schemas in the JSON Schema format. For each dataset, a single local schema must be generated, which is later stored in a *JSON Schemas Repository*. The extraction process analyzes the structure of the NoSQL data, as well as its attributes, to infer information such as JSON data types and mandatory/optional elements. This information extracted from the data structure is stored in a *Mapping Catalog* and can be used to enrich the conceptual view to be produced in the next step. Since the JSON Schema recommendation does not support the specification of spatial data schemas, an extension of this recommendation should be proposed to allow this specification. Once the local schemas are extracted in the JSON Schema format, possible with geographic properties, the *Schema Integration Component* takes effect. It aims to integrate the local schemas obtained previously and generate an unified semantic view represented in RDF. This integration process is based on matching operations of schemas appropriate to JSON Schema. The Schema

Integration Component will be developed in future works. In this Thesis, it is only intended to map the JSON Schemas, produced in the Schema Extraction Component, to a RDF schema representation.

The RDF Schema produced must be stored in an *RDF View Repository* so applications developers with interest in the integrated data can access it. Also, the Mapping Catalog should be populated with information about the matches between the local JSON Schemas and the RDF representation.

4. Experimental Evaluation

We conducted initial experiments to verify the correctness and completeness of the JSON $\rightarrow$ JSON Schema mappings. To do so, five JSON documents with several data types and heterogeneous structures were created. An accuracy of 100% was obtained since all expected data types in the input documents were identified. These data types include mandatory attributes, Extended JSON types, number of items in an array, and union. Experimental results showed that the processing time spent on reading the documents and generating the raw schemas reached around 99% of the total processing time, since all the documents in a collection must be read. It shows that the bottleneck of our process is the document reading and not the processing steps themselves.

We also evaluate our approach considering the same JSON dataset used by the works of [Wang et al. 2015] and [Baazizi et al. 2017]. These datasets have many different raw schemas, the number of raw schemas extracted is very close to the total of documents in the dataset. According to [Wang et al. 2015], this is due to the nature of the datasets, such as *DBPedia*, whose documents usually have some heterogeneities. The results points out that the accuracy of our approach is promising, being equivalent or superior than related work. This was because we consider the name and type of each document attribute to define the raw schema, and not only the attribute names. In fact, the data type is also important to allow more accurate queries with filters on certain attributes.

5. Methodology and plan

This paper presented a methodology for Reverse Engineering of aggregate-oriented NoSQL Databases, with emphasis on geographic data. Our methodology considers the element data types (JSON and Extended JSON data type), including geographic data, for creating schemas in the JSON Schema format. We also consider creating conceptual views of schemas in RDF format. Preliminary experiments have demonstrated that our methodology is equivalent to or superior to state-of-the-art approaches.

This Thesis is in the middle of its development, and the next steps are: a) update the works related to the state-of-the-art; b) development of local schema extraction prototypes for key-value and columnar NoSQL DBs; c) propose roles to mapping JSON Schemas to RDF format and to develop complementary experiments to validate the whole process; d) writing articles; e) defense of the Doctoral Thesis, scheduled for Feb. 2020.

Two papers with partial results were presented and awarded at the Regional School of Database (ERBD 2017 and 2018). An article was published at IRI 2018 (Qualis B1), addressing schema extraction in document-oriented NoSQL DB . Other publications planned are: *i*) a survey on NoSQL DB schema extraction to be submitted to the

ACM Computing Surveys (Qualis A1); *ii*) a paper describing the integration component to be submitted to the IEEE International Conference on Software Analysis, Evolution, and Reengineering (SANER) (Qualis A2); *iii*) a paper describing the Reverse Engineering process of Geographic Data and the use of GeoJSON and JSON Schema to be submitted to ACM Transactions in GIS (Qualis B1).

References

Baazizi, M.-A., Lahmar, H. B., Colazzo, D., Ghelli, G., and Sartiani, C. (2017). Schema Inference for Massive JSON Datasets. In *Proc. 20th EDBT*.

Cattell, R. (2011). Scalable SQL and NoSQL data stores. *ACM SIGMOD Record*, 39(4):12.

Discala, M. and Abadi, D. J. (2016). Automatic Generation of Normalized Relational Schemas from Nested Key-Value Data. In *Proceedings SIGMOD '16*, pages 295–310, New York, NY, USA. ACM.

Izquierdo, J. L. C. and Cabot, J. (2013). Discovering implicit schemas in JSON data. In *LNCS*, volume 7977 LNCS of *ICWE'13*, pages 68–83, Berlin, Heidelberg. Springer-Verlag.

Kapsammer, E., Kusel, A., Mitsch, S., Proll, B., Retschitzegger, W., Schwinger, W., Schonbock, J., Wimmer, M., Wischenbart, M., and Lechner, S. (2012). User profile integration made easy - Model-driven extraction and transformation of social network schemas. In *Proc. 21st WWW*, pages 939–948.

Karpov, V. (2017). Mongoose NPM package.

Kiran, V. K. and Vijayakumar, R. (2014). Ontology based data integration of NoSQL datastores. In *9th ICIIS*, pages 1–6.

Klettke, M., Storl, U., and Scherzinger, S. (2015). Schema Extraction and Structural Outlier Detection for JSON-based NoSQL Data Stores. In *BTW*, volume 241 of *LNI*, pages 425–444. GI.

Liu, Z. H., Hammerschmidt, B., Mcmahon, D., Lu, Y., and Chang, H. J. (2016). Closing the Functional and Performance Gap Between SQL and NoSQL. In *Proceedings of the SIGMOD'16*, pages 227–238, New York, NY, USA. ACM.

Mesiti, M. and Valtolina, S. (2014). Towards a user-friendly loading system for the analysis of big data in the internet of things. In *Proceedings IEEE 38th COMPSACW*, pages 312–317.

Ruckstieß, T. (2017). Mongodb-schema NPM package.

Ruiz, D. S., Morales, S. F., and Molina, J. G. (2015). Inferring versioned schemas from NoSQL databases and its applications. *Lecture Notes in Computer Science*, 9381:467–480.

Sadalage, P. J. and Fowler, M. (2013). *NoSQL distilled : a brief guide to the emerging world of polyglot persistence*. Addison-Wesley.

Wang, L., Hassanzadeh, O., Zhang, S., Shi, J., Jiao, L., Zou, J., and Wang, C. (2015). Schema Management for Document Stores. *VLDB Endowment*, 8(9):922–933.