New Workflows in NoSQL Schema Management*

Michael Fruth
University of Passau
Passau, Germany
michael.fruth@uni-passau.de

Kai Dauberschmidt
University of Passau
Passau, Germany

Stefanie Scherzinger
University of Passau
Passau, Germany
stefanie.scherzinger@uni-passau.de

ABSTRACT

Many NoSQL document stores allow for flexibility w.r.t. schema management: For instance, MongoDB allows to switch between a schema-free and a schema-fixed mode of operation. For declaring such schemas, the JSON Schema language has become highly popular. We introduce the prototype software *Josch*, first demoed at ICDE 2021, which enhances the NoSQL schema management workflow by integrating novel tools for checking JSON Schema containment. We point out new research challenges in this context.

Reference Format:
Michael Fruth, Kai Dauberschmidt, and Stefanie Scherzinger. New Workflows in NoSQL Schema Management. In the 2nd Workshop on Search, Exploration, and Analysis in Heterogeneous Datastores (SEA Data 2021).

Artifact Availability:
The source code has been made available online at [7].

1 OVERVIEW

NoSQL document stores such as MongoDB allow to switch between a schema-free and a schema-fixed mode of operation, by registering a JSON Schema [4, 11] declaration. Apart from solutions for isolated tasks, such as extracting a schema declaration from persisted documents, or validating documents against this schema, there are tools that combine these steps into comprehensive end-to-end schema management workflows (e.g. Hackolade [9] or Darwin [12, 16]).

Towards this family of software products, we contribute a new prototype called Josch [6, 7], where we enhance schema management workflows by integrating novel tools for checking JSON Schema containment. In interaction with Josch, we identify new research challenges for both practitioners and theoreticians working on search, exploration, and analysis in heterogeneous datastores.

2 WORKFLOWS

Our application scenario showcases a DevOps team who started application development and production operations with a MongoDB backend in schema-free mode. For data quality assurance, the team at one point decides to register a JSON Schema declaration with its MongoDB backend, so all writes are validated against this schema.

Schema extraction & validation. The DevOps team first has to extract a schema declaration from the persisted data [2, 9, 13–15]. Often, schema extraction algorithms rely on sampling to cope with large data volumes. Consequently, the extracted schema may not faithfully describe the entire data instance. In order to avoid validation errors at runtime, the entire data instance needs to be validated against the extracted schema. This impacts database performance.

Schema refactoring & containment checking. When the schema is edited, e.g. adjusting it to account for outlier documents, or restructuring it for better readability, the team risks that the schema semantics is unintentionally changed. In JSON Schema containment checking, two JSON Schema declarations are compared based on their semantics. Thus, we can automatically decide whether the schema semantics has been changed.

For illustration, let us consider two excerpts of JSON Schema documents that describe the month of a publication, $S1$: `{"type": ["number","string"]}` and $S2$: `{"type": ["number"]}`. Schema $S2$ is *contained* in $S1$, and therefore more restrictive, as it requires the month to be numeric, whereas $S1$ also allows a string.

3 RESEARCH CHALLENGES

We refer to our extended version [6] of this paper for a more detailed discussion of related work. The full workflow just outlined is supported by our software prototype Josch [6, 7], where Josch is geared to (but not limited to) MongoDB, and employs the third-party tools jsonsubschema [8] and is-json-schema-subset [10] for JSON Schema containment checking.

State-of-the-art JSON Schema containment checkers do not provide any explanation as to why two schemas differ. As a form of explainability, we may resort to generating a *witness document* [1], i.e., a JSON document that is valid w.r.t. one schema but not the other. At the moment, this is still a young research field.

Another limitation of current JSON Schema containment checkers are negation and recursive references [5]. While negation is rarely used in real-world schemas, it can lead to complex schemas [3].

The extracted schemas tend to be simplistic, yet highly verbose. A semi-automated refactoring that automatically extracts and introduces references for repeating structures to alleviate these shortcomings could prove helpful. Yet both schema refactorization and the extraction of complex schemas are open research challenges.

4 OUTLOOK

Solutions to the challenges outlined would also find application beyond NoSQL schema management, e.g., in the static validation of machine learning pipelines, as in the IBM LALE project [8].

[*] An extended version of this work has been presented as a demo at ICDE 2021 [6].

Published in the Proceedings of the 2nd Workshop on Search, Exploration, and Analysis in Heterogeneous Datastores, co-located with VLDB 2021 (August 16-20, 2021, Copenhagen, Denmark) on CEUR-WS.org.

ACKNOWLEDGMENTS

We thank Mohamed-Amine Baazizi, Dario Colazzo, Giorgio Ghelli, and Carlo Sartiani for sharing their insights on JSON Schema, Uta Störl for her comments on our full version of this paper, and the authors of [8] for assistance in using their tool. We thank Pascal Desmarets for providing us with an academic Hackolade license, as well as his feedback from the practitioners' point-of-view.

This project was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), grant #385808805.

REFERENCES

[1] Lyes Attouche, Mohamed-Amine Baazizi, Dario Colazzo, Francesco Falleni, Giorgio Ghelli, Cristiano Landi, Carlo Sartiani, and Stefanie Scherzinger. 2021. A Tool for JSON Schema Witness Generation. In *Proc. EDBT*. 694–697.

[2] Mohamed-Amine Baazizi, Dario Colazzo, Giorgio Ghelli, and Carlo Sartiani. 2019. Parametric schema inference for massive JSON datasets. *VLDB J.* 28, 4 (2019), 497–521.

[3] Mohamed-Amine Baazizi, Dario Colazzo, Giorgio Ghelli, Carlo Sartiani, and Stefanie Scherzinger. in-press. An Empirical Study on the "Usage of Not" in Real-World JSON Schema Documents. In *Proc. ER.*

[4] Pierre Bourhis, Juan L. Reutter, Fernando Suárez, and Domagoj Vrgoc. 2017. JSON: Data model, Query languages and Schema specification. In *Proc. PODS*. 123–135.

[5] Michael Fruth, Mohamed-Amine Baazizi, Dario Colazzo, Giorgio Ghelli, Carlo Sartiani, and Stefanie Scherzinger. 2020. Challenges in Checking JSON Schema Containment over Evolving Real-World Schemas. In *Proc. EmpER*. 220–230.

[6] Michael Fruth, Kai Dauberschmidt, and Stefanie Scherzinger. 2021. Josch: Managing Schemas for NoSQL Document Stores. In *Proc. ICDE*. 2693–2696.

[7] Michael Fruth, Kai Dauberschmidt, and Stefanie Scherzinger. 2021. *sdbs-unip/josch: Josch Version 1.0.0.* https://doi.org/10.5281/zenodo.5155117

[8] Andrew Habib, Avraham Shinnar, Martin Hirzel, and Michael Pradel. 2021. Finding Data Compatibility Bugs with JSON Subschema Checking. In *ISSTA*. 620–632.

[9] Hackolade. online. *Hackolade.* https://hackolade.com

[10] haggholm. online. *is-json-schema-subset.* https://github.com/haggholm/is-json-schema-subset version 1.1.24.

[11] JSON Schema. online. *JSON Schema.* https://json-schema.org

[12] Meike Klettke, Hannes Awolin, Uta Störl, Daniel Müller, and Stefanie Scherzinger. 2017. Uncovering the Evolution History of Data Lakes. In *Proc. Big Data*. 2462–2471.

[13] Meike Klettke, Uta Störl, and Stefanie Scherzinger. 2015. Schema Extraction and Structural Outlier Detection for JSON-based NoSQL Data Stores. In *Proc. BTW*. 425–444.

[14] Diego Sevilla Ruiz, Severino Feliciano Morales, and Jesús García Molina. 2015. Inferring Versioned Schemas from NoSQL Databases and its Applications. In *Proc. ER*. 467–480.

[15] William Spoth, Oliver Kennedy, Ying Lu, Beda Christoph Hammerschmidt, and Zhen Hua Liu. 2021. Reducing Ambiguity in Json Schema Discovery. In *Proc. SIGMOD*. 1732–1744.

[16] Uta Störl, Daniel Müller, Alexander Tekleab, Stephane Tolale, Julian Stenzel, Meike Klettke, and Stefanie Scherzinger. 2018. Curating Variational Data in Application Development. In *Proc. ICDE*. 1605–1608.