Schema Evolution and Gravitation to Rigidity: A Tale of Calmness in the Lives of Structured Data

Panos Vassiliadis(✉)

Department of Computer Science and Engineering,
University of Ioannina, Ioannina, Hellas
pvassil@cs.uoi.gr

Change is the essential process of all existence – Star Trek's Spock

Evolving dependency magnets, i.e., software modules upon which a large number of other modules depend, is always a hard task. As Robert C. Martin has nicely summarized it (see http://www.oodesign.com/design-principles.html), fundamental problems of bad design that hinder evolution include *immobility*, i.e., difficulty in reuse, *rigidity*, i.e., the tendency for software to be difficult to change and *fragility*, i.e., the tendency of the software to break in many places every time it is changed. In such cases, developers are reluctant to evolve the software to avoid facing the impact of change. How are these fundamentals related to schema evolution? We know that changes in the schema of a database affect a large (and not necessarily traced) number of surrounding applications, without explicit identification of the impact. These affected applications can then suffer from syntactic and semantic inconsistencies – with syntactic inconsistency leading to application crashes and semantic inconsistency leading to the retrieval of data other than the ones originally intended. Thus, the puzzle of gracefully facilitating the evolution of data-intensive information systems is evident, and the desideratum of coming up with engineering methods that allow us to design information systems with a view to minimizing the impact of evolution, a noble goal for the research community.

Several research paths towards this goal are being pursued. A first path involves works concerning an algebra of schema evolution operations, that can allow the description of the history of schema changes in a semantically rich sequence of operations [4,7]. Another path involves the management of the impact of changes [1,9]. A fairly novel path involves the identification of profiles and patterns in the usage of relational database access technologies in open source projects [5,6,10]. Moreover -and in particular, what interests us in the context of this talk- *as all engineering should be based on well-understood mechanics and laws, the research community has also tried to uncover mechanics and patterns that govern schema evolution.* Knowing the underlying mechanisms of schema evolution is fundamental in engineering solutions that gracefully handle it: without this knowledge we can easily stray in solutions that have no relationship to real-world problems. Although the body of work is rather small compared to the importance of the matter, one cannot ignore that access to schema histories was practically impossible before the proliferation of Free and Open Source Software (FOSS). Thus, apart from an original study in the early '90s [12], it was

Y. Ouhammou et al. (Eds.): MEDI 2017, LNCS 10563, pp. 18–23, 2017.
DOI: 10.1007/978-3-319-66854-3_2

only in the late '00s that the problem gained a certain momentum that continues till today [2–4,8,11,19], basically due to the availability of FOSS projects that contain DDL files in their history.

Since our team in the University of Ioannina started working on this topic, in 2013, we have encountered several interesting patterns of schema evolution; these findings will constitute the main body of this keynote talk.

In [13,14], we have studied how the schema of a database evolves in terms of its size, growth and activity. Our findings indicate that *schemata grow over time in order to satisfy new requirements, albeit not in a continuous or linear fashion, but rather, with bursts of concentrated effort of growth and/or maintenance interrupting longer periods of calmness* (Fig. 1).

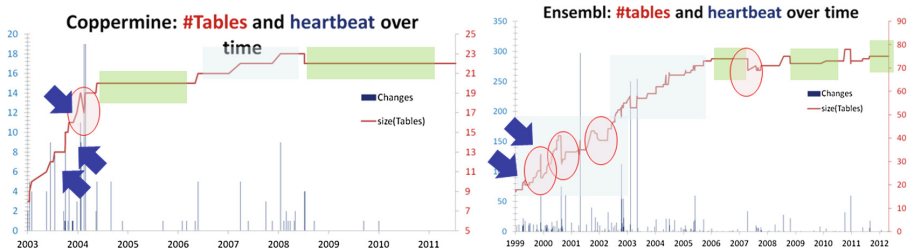

Fig. 1. Summary of [14] with schema growth over time (red continuous line) along with the heartbeat of changes (spikes) for two datasets. Overlayed darker green rectangles highlight the calmness periods, and lighter blue rectangles highlight smooth expansions. Arrows point at periods of abrupt expansion and circles highlight drops in size. (Color figure online)

A different, innovative research path that we have ignited in [17,18] was to study profiles and patterns of *tables*, rather than *schemata*. We have tried to correlate static properties of tables, like the point of birth, or their schema size at birth, with characteristics of activity, like the sum of changes the tables have undergone, or their survival (we like to refer to tables removed as "dead" and table that made it to the last known version of the history that we study as "survivors"). We came up with four patterns (Fig. 2).

1. The *Γ pattern* pattern (albeit with several exceptions) suggests that tables with large schemata tend to have long durations and avoid removal.
2. The *Comet pattern* suggests that the tables with most updates are frequently the ones with medium schema size.
3. The *Inverse Γ pattern*, the one with the fewest exceptions, states that tables with medium or small durations produce amounts of updates lower than expected, whereas tables with long duration expose all sorts of update behavior.
4. The *Empty Triangle pattern* indicates that the majority of removed tables have mostly short lives, which in turn, implies a low probability of deletion for old timers.

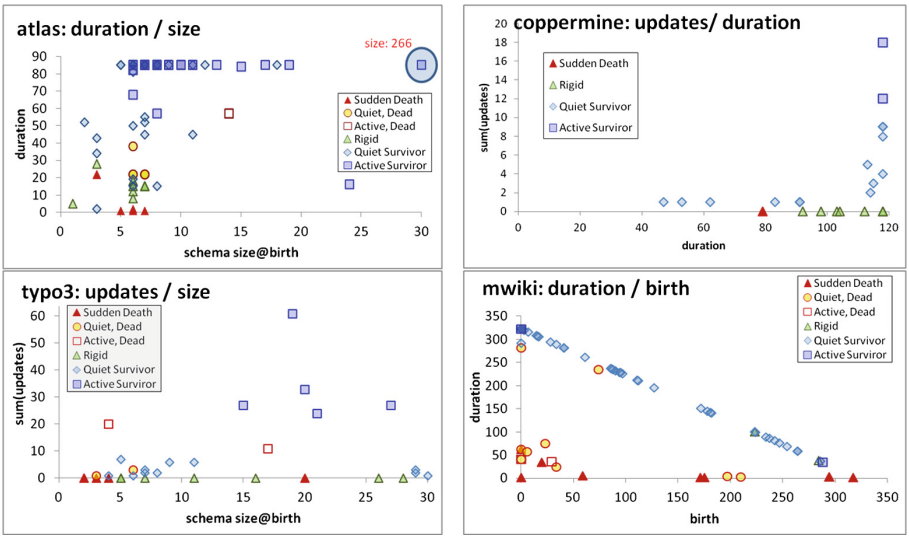

Fig. 2. The 4 patterns of [17,18]: Gamma (top left), inverse Gamma (top right), comet (bottom left) and empty triangle (bottom right).

In [16] we have studied how survivors differ from dead tables with respect to the combination of duration and activity profile. The resulting pattern was named *Electrolysis pattern* due to the intense antithesis in the lives of dead and survivor tables (Fig. 3):

1. Dead tables demonstrate short or medium lifetimes (much shorter than survivors), practically never at high durations. Moreover, with few exceptions, the less active dead tables are, the higher the chance to reach shorter durations.
2. Oppositely to dead tables, survivors are mostly located at medium or high durations. The more active survivors are, the stronger they are attracted towards high durations, with a significant such inclination for the few active ones that cluster in very high durations (which makes the antithesis with the dead ones quite intense).

If time permits, this talk will also cover recent results in topics like foreign key evolution [15] and evolution of Web Services [20].

A key message of this talk is that change is at much lower levels than one would expect. We encounter this observation again and again, in several of the aforementioned research explorations, and, we are quite confident to say that *the absence of evolution is clearly more evident that evolution itself*. This has to do both with survival and activity. In terms of survival, deletions of tables and attributes are much more infrequent than additions. Calmness in the growth of the schema is more frequent than schema growth; moreover, tables are rarely resized. In terms of activity, quiet tables with low rates of change are the majority; rigid tables outnumber the (really few) active ones, both in the survivor and,

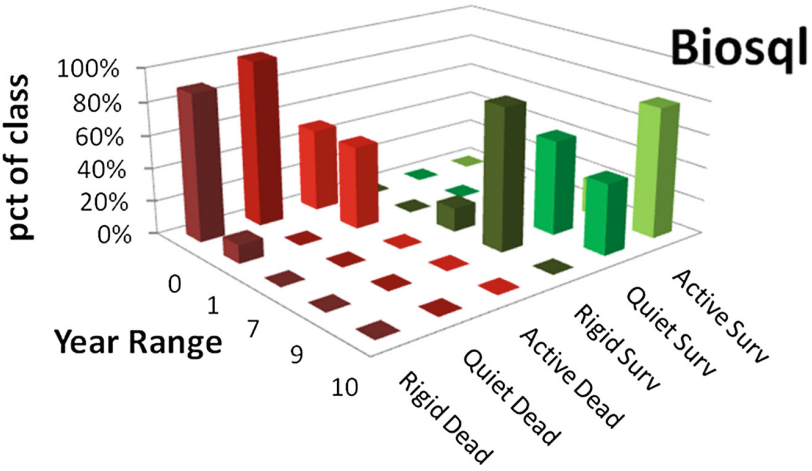

Fig. 3. The Electrolysis pattern [16]: the left axis denotes durations in years; the right axis denotes the level of activity (or, *LifeAndDeath* class) of tables (rigid have undergone zero change, active have an update rate of 10% or more, and quiet is the -majority of- all the rest; the vertical axis denotes the percentage of tables with respect to their activity class

in particular, in the dead class. Therefore, the presence of growth and occasional maintenance actions should not eventually overshadow the absence of restructuring and update that is a significant characteristic of the life of a schema in the long term.

We attribute this phenomenon to what we have named as *gravitation to rigidity*. Any change in the schema requires the maintenance of the surrounding applications. Especially in the case of deletions, renames or type updates, the result might be syntactic inconsistency (and therefore failure) of the application, which is a grave consequence, especially if different developers are involved. The plethora of concurring signs that point towards gravitation to rigidity is possibly the single most important discovery of this line of research and a clear sign of vulnerability in the way the relational model, queries and applications are entangled.

We kindly refer the interested researchers to our website http://www.cs.uoi.gr/~pvassil/projects/schemaBiographies/index.html that contains pointers to available data sets, the tools that we have built and material around our published work.

Acknowledgements. The results of our team in the University of Ioannina on the topic of studying and understanding the mechanics of schema evolution would have never been possible without the collaboration and dedication of many colleagues and students. The list of people who have contributed to this effort includes my colleague and long-time collaborator Apostolos Zarras, as well as several of my students who have worked on the topic, specifically Ioannis Skoulis, Fanis Giahos, Michael Kolozoff,

Athanasios Pappas, and Maria Zerva. Special mention also goes to my long-term collaborator George Papastefanatos and my PhD student Petros Manousis, with whom we have collaborated for long on the topic of managing schema evolution, from its engineering perspective.

References

1. Cleve, A., Brogneaux, A.-F., Hainaut, J.-L.: A conceptual approach to database applications evolution. In: Parsons, J., Saeki, M., Shoval, P., Woo, C., Wand, Y. (eds.) ER 2010. LNCS, vol. 6412, pp. 132–145. Springer, Heidelberg (2010). doi:10.1007/978-3-642-16373-9_10
2. Cleve, A., Gobert, M., Meurice, L., Maes, J., Weber, J.H.: Understanding database schema evolution: a case study. Sci. Comput. Program. **97**, 113–121 (2015)
3. Curino, C., Moon, H.J., Tanca, L., Zaniolo, C.: Schema evolution in Wikipedia: toward a web information system benchmark. In: Proceedings of ICEIS 2008. Citeseer (2008)
4. Curino, C., Moon, H.J., Deutsch, A., Zaniolo, C.: Automating the database schema evolution process. VLDB J. **22**(1), 73–98 (2013)
5. Decan, A., Goeminne, M., Mens, T.: On the interaction of relational database access technologies in open source Java projects. CoRR abs/1701.00416
6. Decan, A., Goeminne, M., Mens, T.: On the interaction of relational database access technologies in open source java projects. In: Post-proceedings of the 8th Seminar on Advanced Techniques and Tools for Software Evolution, Mons, Belgium, 6–8 July 2015, pp. 26–35 (2015)
7. Herrmann, K., Voigt, H., Behrend, A., Lehner, W.: CoDEL – a relationally complete language for database evolution. In: Morzy, T., Valduriez, P., Bellatreche, L. (eds.) ADBIS 2015. LNCS, vol. 9282, pp. 63–76. Springer, Cham (2015). doi:10.1007/978-3-319-23135-8_5
8. Lin, D.Y., Neamtiu, I.: Collateral evolution of applications and databases. In: Proceedings of the Joint International and Annual ERCIM Workshops on Principles of Software Evolution (IWPSE) and Software Evolution (Evol) Workshops, IWPSE-Evol 2009, pp. 31–40 (2009)
9. Manousis, P., Vassiliadis, P., Papastefanatos, G.: Automating the adaptation of evolving data-intensive ecosystems. In: Ng, W., Storey, V.C., Trujillo, J.C. (eds.) ER 2013. LNCS, vol. 8217, pp. 182–196. Springer, Heidelberg (2013). doi:10.1007/978-3-642-41924-9_17
10. Meurice, L., Nagy, C., Cleve, A.: Static analysis of dynamic database usage in Java systems. In: Nurcan, S., Soffer, P., Bajec, M., Eder, J. (eds.) CAiSE 2016. LNCS, vol. 9694, pp. 491–506. Springer, Cham (2016). doi:10.1007/978-3-319-39696-5_30
11. Qiu, D., Li, B., Su, Z.: An empirical analysis of the co-evolution of schema and code in database applications. In: Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2013, pp. 125–135 (2013)
12. Sjøberg, D.: Quantifying schema evolution. Inf. Softw. Technol. **35**(1), 35–44 (1993)
13. Skoulis, I., Vassiliadis, P., Zarras, A.: Open-source databases: within, outside, or beyond Lehman's laws of software evolution? In: Jarke, M., Mylopoulos, J., Quix, C., Rolland, C., Manolopoulos, Y., Mouratidis, H., Horkoff, J. (eds.) CAiSE 2014. LNCS, vol. 8484, pp. 379–393. Springer, Cham (2014). doi:10.1007/978-3-319-07881-6_26
14. Skoulis, I., Vassiliadis, P., Zarras, A.V.: Growing up with stability: how open-source relational databases evolve. Inf. Syst. **53**, 363–385 (2015)