Panos Vassiliadis, George Kalampokis. Taxa and super taxa of schema evolution and their relationship to activity, heartbeat and duration. Information Systems, Volume 110, 2022, 102109, ISSN 0306-4379, doi:10.1016/j.is.2022.102109.

Taxa and Super Taxa of Schema Evolution and their relationship to Activity, Heartbeat and Duration

Panos Vassiliadis[a], George Kalampokis[b,1]

[a] *University of Ioannina, Ioannina, Hellas*
[b] *GWF MessSysteme AG, Salonika, Hellas*

Abstract

In this paper, we present the findings of a large study of the evolution of the schema of 195 Free Open Source Software projects. We identify families of evolutionary behaviors, or taxa, in FOSS projects. A large percentage of the projects demonstrate very few, if any, actions of schema evolution. Two other taxa involve the evolution via focused actions, with either a single focused maintenance action, or a large percentage of evolution activity grouped in no more than a couple interventions. Schema evolution also involves moderate, and active evolution, with very different volumes of updates to the schema. We also investigate how the different taxa relate to measurable properties of schema evolution, specifically, duration of schema and project updates, activity volume, and heartbeat. We show that although different taxa have practically very similar duration, the evolutionary characteristics differ in analogy to the "active" character of each taxon. Moreover, by observing certain similarities in the measurable properties of the taxa, we take the opportunity to introduce super taxa, which complement the previous taxonomy with the groupings of the aforementioned taxa in terms of overall profile similarity, resulting in a more concise and intuitive taxonomy, providing a cleaner separation of evolution measures. Finally, we show that schema evolution is frequently, a time-concentrated activity.

1. Introduction

Much like traditional software, database schemata change over time. *Schema Evolution* means that tables and attributes can be added, deleted or renamed, and their data types and keys can be altered. Changing the schema of a database can incur a significant impact, as the surrounding code can be syntactically and semantically inconsistent with the new structure of the schema, leading to application failures or incomplete data delivered to end users. Despite this significance, and in sharp contrast to the study of software evolution and maintenance

Email addresses: pvassil@cs.uoi.gr (Panos Vassiliadis), gtkalampokis@gmail.com (George Kalampokis)
[1] Work done with Univ. Ioannina

for years from the software engineering community, we know very little on how schemata evolve: are there any patterns, similarities, recurring behaviors, or even laws in the way schemata evolve? Understanding the mechanics of schema evolution is a piece of knowledge, currently absent from our body of knowledge as a data engineering community, that apart from enriching humanity's knowledge and moving from word-of-mouth impression to concrete evidence, can facilitate both the management of information systems on the practical side, and its scientific basis on the research side, by providing insights to different audiences: (a) software curators and developers who can benefit from the ability to (even coarsely) predict the tendency of a schema to evolve (which can be used for decision making, recourse allocation, software selection), and, (b) the academic community, who can now have hard evidence on the evolutionary phenomena in order to educate the students and foster subsequent research on the topic.

The goal of this paper is to expand our understanding of how relational schemata in the domain of Free Open Source Software (FOSS) projects evolve at the logical level, by introducing taxa and super taxa of projects with similar schema evolution profiles.

1.1. Why is this important?

Fundamentally, by understanding whether clusters of projects with discernible behaviors exist, and by relating them to measurable properties of the projects in terms of schema evolution, we can achieve a very important gain: as soon as we can assign a project to a certain taxon, we can predict the breadth of the change of its quantifiable evolution-related properties. *Therefore taxa can serve as the "alphabet" of patterns of schema evolution.*

Naturally, the identification of patterns is an open-ended design problem: depending on the questions asked, and the measures used, different patterns can arise. *This research effort serves as a first step in this journey, by establishing important measures, terminology, research questions and answers as well as a large, diverse dataset of schema histories for subsequent research to work upon.*

1.2. Means

To address the goals of the paper, we have performed the largest study of schema evolution ever performed in the literature (to the best of our knowledge) via a systematic collection of data for schemata in FOSS projects. Specifically, we collected 327 schema histories in Free Open Source Software projects, based on a principled collection method and with quality criteria to avoid dummy projects. Out of them, we identified 195 candidate whose history seemed to exhibit evolutionary characteristics, and for each such project, we automatically extracted schema histories from their git repository hosted in Github[2]. For each

[2]All data, results, summary statistics and extra explanations are publicly available at `https://github.com/DAINTINESS-Group/Schema_Evolution_Datasets` aka `https://bit.ly/3nMggEx` (at Github). The site `https://www.cs.uoi.gr/~pvassil/projects/schemaBiographies/` aka `https://bit.ly/3ptnBJo`, summarizes our research.

such history (which is a list of versions of the schema DDL file), we extracted the differences between subsequent versions and measurements in terms of timing, schema size, numbers of tables and attributes changed, coming up with specific measures of change to characterize the *heartbeat* of change of a schema and its characteristics. We have also encoded several of the measured quantities (e.g., the total update activity, the shape of the line of the schema size) and carefully studied the observed phenomena

1.3. The Taxa of Schema Evolution

To the best of our knowledge, *this is the largest study ever performed, surpassing previous studies by at least an order of magnitude*. Apart from the sheer volume of the schemata studied, the principled method we followed for collecting the schemata, extracting their versions and performing the analysis, allows us to argue that *our results are largely generalizable for Free Open Source Software*.

Apart from the above, our study includes a systematic study of schema evolution heartbeat, also for the first time in the literature (to the best of our knowledge). To achieve this, it was necessary to clearly specify *the experimental method, the nomenclature, the visualization and the analysis methods*. Subsequent studies can benefit from our framing of the problem and our explanations for the decisions of units and experimental method choices.

Secondly, we verify previous research that schema evolution is indeed evident; however, *the examination of a large corpus of projects produces unshakable evidence for the first time, that its absence is way more omnipresent than its presence*.

Third, this is the first time ever *that taxa of schema evolution are presented*. Following an iterative, qualitative process, we have studied the evolution of the collected schema histories and grouped them in *taxa of evolution, i.e., families of schemata, which share similar evolution characteristics*. The taxa of schema evolution are: (1) completely *frozen* schema histories with zero change at the logical level; (2) *almost frozen* histories of very small change, typically with few intra-table attribute modifications; (3) almost frozen histories but with a single spike of change and almost no other change (*Focused Shot and Frozen*); (4) histories of *moderate evolution*, without spectacular changes, but rather small deltas spread throughout the life of a project; (5) projects with evolution similar to the moderate one but also with a pair of spikes on their activity (*Focused Shot and Low*); and (6) histories of *active* projects, typically with significant amount of change both as intra-table change and in terms of table generation and eviction. The identification of taxa provides us with *a fundamental tool for characterizing and forecasting the propensity of the DBA's of a certain database towards change*.

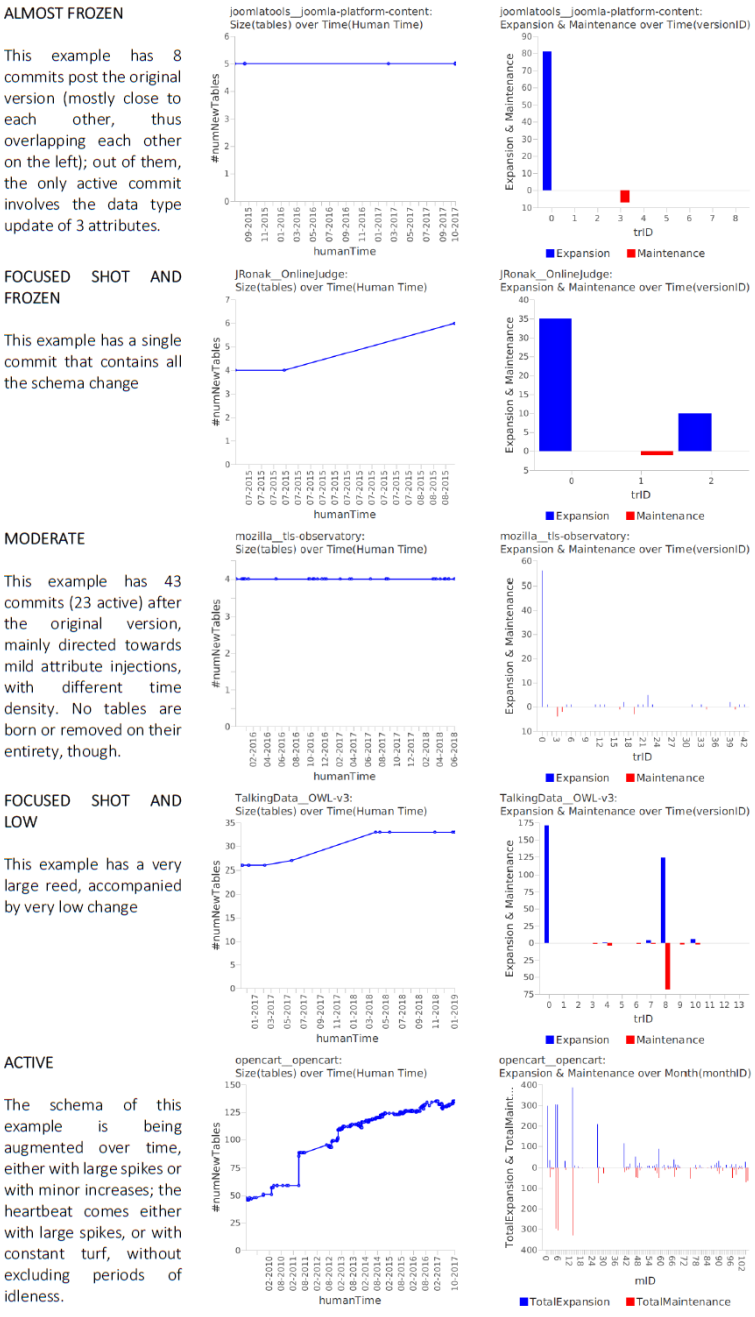

Figure 1: The Taxa of Schema Evolution: (Left) Schema size over time (#tables over human time; (Right) Expansion and Maintenance activity (#attributes affected) per commit for the first four examples, and per month, for the last one.

1.4. Taxa properties

A first presentation of all the above was given in [1] (see Section 2, for a discussion of the novelty of this paper). In the current paper, we extend our knowledge on how the taxa of schema evolution determine quantifiable properties of schema evolution. Specifically, we relate taxa to (a) the measurements of schema and project update period, i.e., for how long the schema and the entire project were being evolved, (b) the measurements of different types of activity like the expansion of the schema with new tables and attributes and its maintenance with the deletion and data-type update – all of which are gathered under the label "activity", (c) the tempo-related attributes of the evolution, with the characterization of small changes as "turf" and large changes as "reeds". The lessons learned from our study are as follows:

- We demonstrate that *both the period of schema update and the period of changes for the entire software project are practically orthogonal to the taxon of a project*. All taxa have projects of durations within 1 to 3 years and the differences between the taxa are not due to the durations of the projects. A small drift towards higher schema update periods is related to more active taxa.
- Although the taxa were originally derived via manual clustering and a-posteriori proved to have similarities of activity and heartbeat, our study reveals that *heartbeat characteristics alone can determine the taxon of a project*. This shows a direct correlation between activity and heartbeat and allows to separate taxa successfully with very few types of measurement.
- Although in our previous efforts we have argued for activity to be measured in terms of attributes (and continue to do so), we can also see that the above results also hold for the measurement of table births and deaths in the life of a schema (i.e., with tables being the unit of measurement).
- At the same time, *activity attributes, although not making taxa fully separable appear to be in direct relation to the level of each taxon* (consistently for both the overall activity and its breakdown in different subcategories).

1.5. SuperTaxa

At the same time, apart from verifying the separability of the proposed taxa, and understanding the range of their evolution-related measures, we also detected an opportunity. Specifically, we were able to provide a set of *super-taxa*, as groupings of the existing taxa (the object-oriented simile, would treat them as super-classes defined over the existing classes) that *allow (a) a more concise description of the classes, with (b) better separation of the respective measures*. We investigated the properties of the super taxa, too, and all the aforementioned properties are also confirmed.

1.6. Timing of Schema Evolution

Finally, we also discuss the time distribution of schema evolution in the life of the system that contains it. We show that for the most part of the corpus of schemata, *evolution is a narrowly concentrated effort in terms of time*.We show that in most cases, the schema comes with a large part of its structure at its birth and few updates later. The transition between birth and attainment of a large part of the total changes made to the schema is typically made in a short period of the life of the schema. Expectedly, taxa with higher rates of change demonstrate less of these effects compared to the more "frozen" ones.

1.7. Roadmap

In Section 2, we present related work. In Section 3, we discuss our experimental method, nomenclature, and threats to validity, in order to support the generalizability of our results. In Section 4, we discuss our findings concerning the taxa of schema evolution. In Section 5, we discuss the statistical support on the validity of the proposed taxa. In Section 6, we describe the measurable properties of the different taxa. In Section 7, we introduce super taxa and discuss their properties and in Section 8, we discuss schema maintenance concentration in time. We conclude in Section 9 with a discussion of final thoughts.

2. Related Work

2.1. Why studying schema evolution is important

There is a small set of works that discuss the gravity of the problems related to schema evolution in practice. In [2], Stonebraker at al., mention their personal -still: anecdotal- evidence after interviewing several DBA's who appear to intentionally try to avoid schema evolution in the first place, and issue a call to arms for collecting data in order to study schema evolution with a principled manner. In [3], Limoncelli explains the difficulties of schema evolution and data migration for live systems and the difficulty to bring code and schema in sync. In [4], the authors engaged in monitoring the actions of a database architect during the process of evolving the database embedded at the core of a software system. The authors collected their observations, measured time spent in different tasks, and, quite importantly summarize the problems observed in six core categories that include (i) the difficulty of "by hand" checks for interdependencies of database constructs, (ii) the difficulty of assessing the impact of schema change to the entire system, (iii) the difficulty of the management of multiple co-instances of the same evolving schema in different projects, as well as the difficulty of (iv) testing it. Moreover, concerning tool support, the problems observed included also (v) the lack of synchronization between the actual state of the database and the state that the IDE used had extracted, and (vi) the need to use a large number of non-integrated tools.

2.2. Algebras for schema evolution and adaptation techniques

The handling of adaptation to schema changes includes several works [5], [6], [7], [8], [9] – see [10] for an overview of query adaptation techniques. The introduction of algebras of schema evolution operations (SMO's), in order to be able to describe sequence of changes (either in forward- or reverse-engineering) includes works like [11], [12], [13].

2.3. Works on studying patterns of schema evolution in general

Apart from a study in the early '90s [14], it was only the proliferation of Free and Open Source Software (FOSS) that gave a momentum to the study of how schemata evolve. The original studies, [14] and 15 years later, [15], were focused on a single case study (a hospital database and mediawiki, respectively) and the quantification of changes in different categories, and the main finding is the significant dominance of expansion over deletion, as well as the different intensity of change at different tables. [16] follows along the same lines. [17] and [18] report that a large part of tables and modules of application code are not-synchronized at all times. The largest study so far has been [19] with ten open-source schemata studied. Again, the percentages of changes are reported, with *add table*, *add column* and *change column datatype* being the most populous. The lack of integrity constraints in several places (independently verified and explained later, in [20]), is also reported, along with the non-synchronization of application code and schema, as well as the presence of focused periods of change in the early life of the schemata. [21] shows that schemata grow over time with bursts of concentrated effort of growth and/or maintenance interrupting longer periods of calmness. [22] and [23] study patterns of *tables*, rather than *schemata*, best summarized by the *Electrolysis pattern*, named after the intense antithesis in the lives of dead and survivor tables: whereas dead tables are attracted to lives of short or medium duration and absence of schema update activity, survivors are mostly located at medium or high durations and the more active they are, the stronger they are attracted towards high durations. Two studies, specifically [20] and [24], discuss patterns and behaviors around the evolution of foreign keys, and the existence of different attitudes towards them (from full support to their total eclipse). Later studies have also moved towards the study of schema evolution in the realm of JSON, NoSQL databases [25], [26] – see [27] for an overview.

2.4. Comparison to related work

Compared to the previous work, the results of sections 3, 4, and 5 (already presented in [1] for their most part) (a) describe the first attempt to *study the heartbeat* and provide *profiles of schema evolution* for the studied projects, and (b) come over *a publicly available, new, dataset, one or two orders of magnitude higher in terms of studied projects*, via a *principled collection method*; resulting in a strong generalizability for the produced profiles for FOSS projects. In this paper, we have extended the discussion on threats to validity to demonstrate even more clearly the validity of the collected data set.

The rest of the paper, in Sections 6, 7 and 8 offers completely novel observations, appearing for the first time in the bibliography that build on top of the taxa of [1], in two ways. First, *we systematically study (a) how taxa are separated with respect to their behavior, and, (b) how different families of properties of the evolution of schemata relate to their classification to taxa, thus allowing us to be able to relate taxa and behaviors in a concrete way. Second, we extend the classification of [1] by proposing super taxa*, thus, allowing a more concise and intuitive taxonomy with increased degrees of separation. Third, *we provide evidence on the concentrated timing of schema evolution*, also for the first time.

3. Experimental Setting

In this section, we discuss our data collection process, the artifacts and metrics that have been extracted from the fully automated processing of the schema histories, and the threats to validity of this study.

3.1. Data Collection

The goal of the data collection process was to collect data for a large number of projects with quality guarantees. Originally, we tried to work with GHTorrent [28], a well-known GitHub mining tool. Still, we were unable to usefully produce a data set with it, and, we resorted to one of its querying platforms, Google Cloud BigQuery[3]. Among its many datasets, BigQuery provides the GitHub Activity Data dataset in relational format, along with SQL facilities to query it. The GitHub Activity dataset is a 3TB+ dataset that contains a full snapshot of the contents and the commits of more than 2.8 million open source GitHub repositories. We queried the contents table for all file descriptions ending to a '.sql' suffix, in 2019-04-24 and 25, and obtained a collection of SQL file descriptions (to which we refer to as SQL-Collection, hereafter) for 133,029 repositories.

Since this number of files and repositories is extremely high to handle, we had to narrow it down via a principled selection method. To this end, we combined the SQL-collection that we obtained with another public dataset, available via BigQuery as the Libraries.io dataset[4]. Libraries.io is an opensource community monitoring and gathering metadata for over 2.7M unique open source packages from 3 source code repositories, namely GitHub, Gitlab and BitBucket. We have worked with the collection exported at 2018-12-22. The Libraries.io collection offers project metadata, including whether the project was an original project or a fork, its number of stars, watchers, etc.[5] We joined the two data sets over (a) their repository names and (b) the URL of their projects, taking care to

[3] `https://cloud.google.com/bigquery/`

[4] `https://libraries.io/about`

[5] Among others, a GitHub project has (a) stars (i.e., someone considered it interesting and pressed the 'Star' button), (b) forks (i.e., a user copies the project in his own 'space' to work independently on it), (c) collaborators (users contributing to a project owned by someone else).

include only Libraries.io projects which were (i) original repositories, (ii) with more than 0 stars and (iii) more than 1 contributor.

To alleviate the possibility of void projects, or repetitions of the same change in multiple files, the results where post-processed with several criteria:

- We excluded all results whose file descriptions included the terms 'test' or 'demo' or 'example' in the path.
- For all the cases where multiple vendors were supported, we chose MySQL as the DBMS to investigate (as the most popular DBMS in our collection).
- For all the cases where multiple SQL files were reported, we went through manual inspection, to identify candidates that could be reduced to a single DDL file with the table creation statements. Cases omitted included (i) several DDL scripts in a file-per-table mode, (ii) incremental maintenance of the schema, (iii) the Cartesian product of multiple vendors X different versions of the same schema for different languages (e.g., projects having different schemata for the combination of {English, French, ...} by {mysql, postgres, mssql, ...}).

The result of this post-processing was a data set of **365 FoSS schema histories**, which we refer to as the Lib-io dataset. For all these 365 projects we went on to clone them locally and extract their schema histories between 7 and 26 May 2019.

To remove erroneous or void files, a final post processing took place over the retrieved repositories. First, we removed 14 projects whose history extraction resulted in 0 versions (i.e., their file descriptions in Github Activity did not match their actual, downloaded .git). We also removed the commits with empty files, as well as the histories whose .sql files did not contain "CREATE TABLE" statements. This involved 24 projects. Out of the remaining **327 repositories**, we isolated **132 rigid projects with just one version** of the schema file, i.e., projects whose schema never changed. The number is striking: 132 out of 327 is a vast 40% of projects without any schema evolution(!). Eventually, **we ended up with 195 non-rigid repositories** that were used for our subsequent analysis, and to which we refer as the Schema_Evo_2019 data set, publicly available at `https://bit.ly/3nMggEx` at Github, along with a very precise description of all the steps of the data collection.

3.2. Nomenclature and Measurements

To address the diversity of nomenclature and measurements, in this section,we establish a reference nomenclature.

A *Schema History* is a list of *commits* (a.k.a. *versions*) of the same DDL file of a database schema, ordered over time. A *transition* from an older version i to its subsequent version $i+1$ occurs at the timepoint where version $i+1$ is committed, and potentially incurs changes in the schema. The *initial, originating version* of the history is called, as shorthand, $V0$. *Active commits* are the commits whose sum of updates (see next) exceeds zero. *Non-Active commits* involve

changes in comments, directives to the DBMS, INSERT statements, indexing, and other changes that do not affect the logical capacity of the schema in terms of tables, attributes, data types or primary keys. The *Schema Update Period (SUP)* is the time span (in human time) between the first and the last commit of the schema file. This is a very different time interval than its superset, *Project Update Period (PUP)* that marks the start and end of project history.

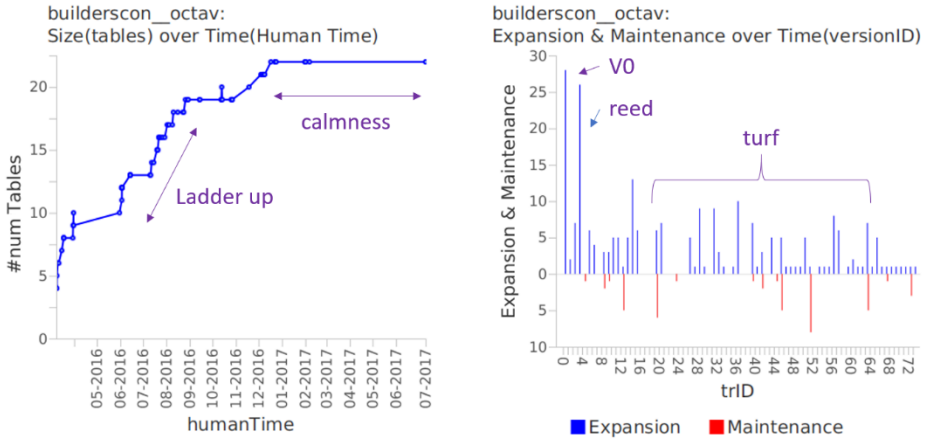

Figure 2: A reference example of schema and activity evolution via the builderscon_octav project: (Left) Schema size over time (#tables over human time): each dot on the line is a commit (observe that many of them do not affect schema size, as they are intra-table changes, or changes to documentation, data insertions etc., that do not affect the schema). The points are placed with respect to human time thus, they are not uniformly distributed. Observe that (a) the main part of the schema growth is a focused period tagged "ladder up" and (b) towards the end of the monitored period, commits are more infrequent and change is smaller. (Right) Expansion and Maintenance activity (#attributes affected) over transition ID (attn: not human time, but sequential id's of commits to the schema file). Blue bars above the x-axis measure expansion (attributes added to the schema), whereas read bars under the x-axis measure maintenance (attribute deletions, data type or PK changes). Observe how the change rates of the left and right parts are not isomorphic.

For each transition of the schema history, our tool, Hecate, automatically computes several categories of measurements. First, it computes *timing* information, like the distance of the i +1 commit from V0 in days, and the running month and year. Second, it registers the *schema size* (no. of tables, attributes) of both the older and the subsequent version of the transition. Third, Hecate identifies and quantifies *updates* (all measured in attributes): attributes born with a new table, attributes injected into an existing table, attributes deleted with a removed table, attributes ejected from a surviving table, attributes having a changed data type, or a participation in a changed primary key. We measure as *Expansion* the sum of attributes born and injected, and as *Maintenance*, the sum of all the other categories. *Total Activity*, or simply *Activity*, of the schema, is the sum of Expansion and Maintenance (in what follows, remember that fundamental unit of measurement of change in our setting is the attribute, for all categories).

We define the *heartbeat* $H = \{c_i(e_i, m_i)\}$ of the schema as the ordered list of pairs (expansion, maintenance), one per commit, of the schema history. Due to the visual impression of the shape of the heartbeat (see Fig. 2), we refer to standing out commits with total activity strictly higher than 14 attributes as *"reeds"*, and commits with lower activity as *"turf"*. The reed limit was produced by taking all single-commit projects, sorting them by activity (producing a power-law like distribution)and splitting them at the 85% limit).

3.3. Threats to validity

From the very beginning, our goal was to be able to clearly specify the scope and generalization of our study.

Scope. *We are interested in the monitoring of the evolution of the logical-level relational schema for significant Free Open Source Software projects, hosted in GitHub.* We want to stress that, in the context of our deliberations, we are not covering or generalizing to proprietary schemata outside the FoSS domain. We do not cover conceptual or physical schemata. We are also restricted in relational schemata and not XML, JSON, or another format.

Experimental Reliability. We tested our extraction scripts with OpenCart (the largest of our studied projects) for which we had a previous past extraction of its history, in 2016. The comparison produced an almost identical result, as only one commit out of 412 was missing from the GitHub history we extracted. We manually tested the histories of the retrieved files against the number of commits reported at GitHub for the respective file, for a random sample of 50 cases. In all cases there was an exact match. We also confirmed that the missing projects had also been removed from GitHub at the time of the cloning via a sample of 7 of them. Concerning our own software, we did extensive checks to our metrics computation tools. Overall, although bugs or omissions are still possible, we are quite confident with our software tool suite.

External Validity. The external validity refers to the possibility of generalizing the findings of a study to a broader context. We claim that our elicited repositories and their extracted history give a fairly representative view of schema evolution in FoSS projects.

First, the SQL-Collection data set includes the locations of schemata that are part of Free Open Source Projects (and not proprietary ones), available via GitHub. *Practically, the domain of search was all the .sql files of GitHub* reported at Github Activity and Libio. *In our opinion, this is also a very good representative of open source software overall, as GitHub is the main public repository for FoSS software.* We applied the restriction that the respective files end with a '.sql' suffix. It is possible that other suffixes, are used by developers. To the extent that this would be a non-recommended practice, we believe that the projects ending up in our study are valid candidates to be monitored as significant projects.

Second, the Lib-io data set is a restricted version of the SQL-collection data set with the schemata whose repository path was monitored by the public Libraries.io data set. *We applied the filter of more than one contributor, more*

than 0 stars and non-forking. We believe this to be a fairly broad scope for original projects with a degree of significance (without implying, of course, that other projects are not significant).

Third, *the subsequent filtering, performed an extra quality check*. We believe that filtering out tests, examples and demos is not decreasing the value or validity of our approach. Although databases of these types have their value, monitoring their evolution, would not say much for the essence of a database supporting the regular operation of a software project. For the case of multi-vendor support for schemata, we are also confident that our choice to select only one vendor is the appropriate one, especially since we are studying logical-level changes. The only ambiguous situation, was the necessity to omit multi-file DDL declarations. This improves the precision of our study (as we are sure we get the correct history of the DDL statements) but reduces the recall.

Fourth, *the domains of the collected projects are diverse enough to support our external validity claim*. Specifically, the project domains include Content Management Systems, IoT Management on the cloud, Task Management Systems for operating systems, similarly for web services, Messaging Platforms, Systems for the management of Scientific Data, Web on-line stores, On-line Charging Systems (OCS), CMS extensions, BitTorrent trackers, etc.

Fifth, we performed a post-hoc assessment, and *our results verify that our data set does not involve projects that are "toy" projects, class assignments, or other projects that do not demonstrate effectively the way Free Open-Source Software projects evolve their schemata*. Specifically, our post-hoc assessment of the validity of the dataset collection has been performed with respect to the distribution of the time span of the updates made to each *project* (attn: not schema, but project). Short project durations would signify either (a) that the project is too recent, xor, (b) that the project was abandoned soon after its start. Long durations on the other hand, signify projects that were maintained over a longer period of time. [6]

As already mentioned, in May 2019, we cloned the projects from Github and "git logged" (i.e., extracted) the history of their commits. So, for each commit, the date, list of modified files, and commit message was retrieved. By keeping record of the dates of first and last commit of the project and the first and last commit of the schema, we were able to obtain the Project Update Period (PUP) and the Schema Update Period (SUP) that are mentioned in the paper and here.

Our findings are summarized in Figure 3. We group the projects by (a) taxon, and (b) the time span of project updates, PUP (i.e., the time between the first commit and the last commit our git clone tracked) in (i) less than 1 year,

[6] Caveat: naturally, this duration-based hypothesis does not replace the in-depth study of each individual project. However, with 195 projects at hand (a) this is not feasible with any reasonable effort, and, (b) it is a more conservative criterion than the actual study would possibly reveal (i.e., one can see how a more in-depth could possibly validate some of the short-PUP projects, e.g., as young ones, recently ignited, where it seems rather difficult to imagine how other projects could be invalidated).

	<=12M	[13-24M]	>24M	TOTAL	<=12M	[13-24M]	>24M	TOTAL
FROZEN	7	4	23	34	21%	12%	68%	100%
AL.FROZEN	17	10	38	65	26%	15%	58%	100%
FS+Frozen	10	4	11	25	40%	16%	44%	100%
MODERATE	4	4	21	29	14%	14%	72%	100%
FS+Low	5	1	14	20	25%	5%	70%	100%
ACTIVE	1	1	20	22	5%	5%	91%	100%
	44	24	127	**195**	23%	12%	65%	100%

Figure 3: Breakdown of durations, Project Update Periods (PUP), per taxon; right hand side percentages refer to each row.

(ii) between 1 and 2 years, and (iii) longer than 2 years. With the exception of Focused Shot and Frozen, the rest of the taxa demonstrate an overwhelming majority of projects having PUP higher than 2 years. If one extends the validity time span to include PUP between 1 and 2 years, 151 out of 195 projects are valid candidates for study (i.e., 3 out of 4).

We find the above finding to be quite strong a statement. Take into consideration that half the taxa demonstrate very small -or even zero- amount of change at their schema. At the same time, and in sharp contrast to the above, their vast majority, esp., in the taxa of higher activity, the project time spans are significant.

Open Issues. An issue of future investigation is also the commit habits of different projects: other teams commit small increments, other ones group changes in larger commits. Although this has an impact to the internal structure of changes, it does not impact the aggregate profile of a project. An extra issue of concern has to do with the non-linearity of git histories [29]. We investigate the entire schema history, whereas one might consider focusing on a single branch of the history.

4. The Taxa of Schema Evolution

Our study starts with our answer to the following research questions:

RQ1. Is schema evolution extensively present? Is schema evolution a process that frequently encountered, and if yes, to what measurable extent does it occur in terms of frequency and volume?

RQ2. Are there consistent patterns in the lives of schemata – i.e., can we extract families, ("taxa" as in biology) of schemata, with respect to the way they evolve over time?

4.1. Derivation and intuition of taxa of similar evolution profile

4.1.1. Qualitative derivation of taxa

We have organized the studied schemata in families of evolutionary behavior, which we call *"taxa"*. *The derivation of the taxa was a (i) manual, (ii)*

qualitative and (iii) iterative process. We automatically generated charts, in particular, the heartbeat and the schema size chart[7]. Manual inspection made apparent that there was a clear discrimination between *(a) completely frozen histories*, without any change to their logical schema, (b) a very large number of *almost frozen histories*, with very few commits, very small volume of change and mostly without change in the schema size of the project, (c) *moderately evolving projects*, mostly in terms of "regular" small changes and less in terms of schema growth, and, (d) *active* projects with significant change in the schema size, and typically, frequent active commits. Soon, our iterative, manual, visual inspection of the project charts and metrics revealed that we could further isolate two more taxa of *focused change*: a taxon of very few commits (i.e., mostly in the almost frozen category) but with a focused amount of change in a single commit and maybe a couple of commits of very small volume (i.e., the change was fundamentally focused in a single commit), and a taxon of -again- few commits, but also with a couple of reeds, and moderate to high change.

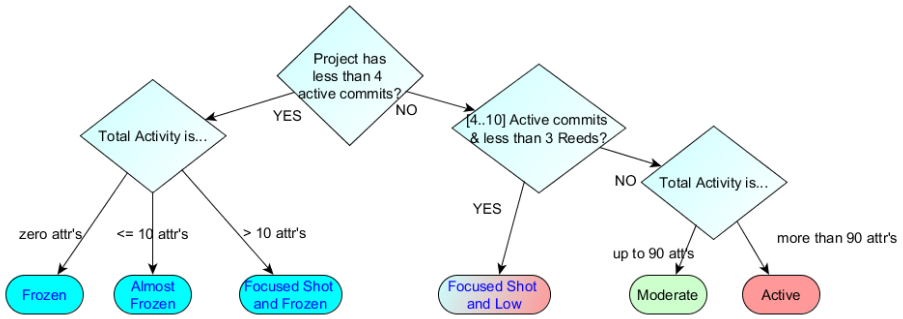

Figure 4: Taxa of Schema Evolution for FOSS Projects

Once this manual, qualitative process was mature, with only a few grey-zone projects having an ambiguous label, we were able to extract a simple *classification tree of taxa* (see Fig. 4), with respect to active commits and activity. Table 1 demonstrates the initial qualitative intuition and the subsequent rule-based, quantitative definitions for the taxa. A Kruskal-Wallis analysis (see Section 5), both for the entire set of the taxa, and for all pairwise comparisons, verifies the difference of the taxa in terms of active commits and total activity. In what follows, it is important to remember that *we have not selected just any random project, but rather, we intentionally restricted our scope to original, stared projects, where people were actually contributing effort to develop and maintain* (see Sec. 3). *Overall, 65% of projects spanned more than 24 months and 77% more than a year.*

[7]Hecate at `https://github.com/DAINTINESS-Group/Hecate` and Heraclitus Fire at `https://github.com/pvassil/HeraclitusFire`, set up our toolset.

Table 1: Intuition and Definition for the Taxa of Schema Evolution.

Taxon	Motivating Intuition and *Resulting Classification Definition* (*italics*)
History-less	Only 1 commit of the .sql file (we did not study them, due to lack of transitions)
Frozen	With history, but with *total activity of 0 changes & 0 active commits*
Almost Frozen	Very few commits and low change volume *Def: At most 3 active commits, change less or equal to 10 updated attributes*
Focused Shot & Frozen	Very few commits, focused change (not necess. small) in a single commit *Def.: At most 3 active commits, change more than 10 updated attributes (typically also involves a single reed)*
Moderate	Moderate rate of heartbeat (active commits), moderate volume of activity *Def.: None of the rest, total change less than 90 updated attributes*
Focused Shot & Low	A couple of reeds and a few active commits, focused (mod. - high) change *Def.: Between 4 and 10 active commits, no more than 2 reeds*
Active	Frequent rate of heartbeat (active commits), high volume of activity *Def.: None of the rest, total change more than 90 updated attributes*

4.1.2. A first quantitative discussion of taxa

In Fig. 5 we present a summary of the statistical profiles of the different taxa with respect to their (a) cardinalities, and, (b) measures that characterize their evolutionary activity. Observe that all taxa come with a significant cardinality, and in any case, each of the taxa alone is several times larger than the corpus of each of the previous studies in the related work (that never surpassed a dozen of studied projects).

Concerning RQ1 requiring a quantified objective assessment of schema evolution, one of the problems that we encountered, and, consequently, *one of our main contributions is the clear specification of important measures of evolution*. Fig. 5 summarizes the most important of them. Volume of change is measured in affected attributes (as the universal unit of change) that can capture both table births and deaths, but also intra-table changes. The number of commits of the DDL file and active commits, that also include changes to it, is a demonstrator of the frequency of change. Reeds and turf commits characterize the density of change. Tables inserted and deleted, as well as tables at start and end characterize the resizing of the schema at a coarser level of detail.

Concerning the contents of Fig. 5 we present a detailed analysis in Section 6, and constrain ourselves here to giving a couple of high level remarks on two -up to now anecdotal- concerns involving the existence/absence of schema evolution as well as its focused nature over time.

- *How is the amount and frequency of change demonstrated?* The frequency of change is really low. Out of the 195 projects studied, 124 (64%) have 0 - 3 active commits. The delta change in terms of tables is significantly small in almost all categories except for the active one (practically between zero and two tables in most categories). Deletions are extremely rare, in particular. Overall, with the exception of the active category, the change in the number of tables is quite small.

- *Is change mostly concentrated in few commits of focused change?* Focused commits of change do exist, and they are also apparent in both moderate and active projects, but also in low-activity projects (where although we do not count reeds, there are small-volume active commits that concentrate the small change of a project). However, focused change is not a recurring practice. Most taxa come with less than 2 reeds, and only active projects practically surpassing this number.

	Frozen				Almost Frozen				Fshot n Frozen				Moderate				Fshot n Low				Active			
Count	34				65				25				29				20				22			
	min	**med**	max	*avg*	min	**med**	max	*avg*	min	**med**	max	*avg*	min	**med**	max	*avg*	min	**med**	max	*avg*	min	**med**	max	*avg*
Sch. Upd. Period (months)	1	**1**	69	*8.24*	1	**6**	99	*11.98*	1	**2**	46	*9.28*	1	**20**	100	*23.62*	1	**17.5**	57	*21.05*	1	**31**	100	*35.95*
TotalActivity	0	**0**	0	*0*	1	**3**	10	*3.62*	11	**23**	383	*45.64*	11	**23**	88	*30.0*	27	**71**	315	*105.15*	112	**254**	3485	*546.14*
#Commits	2	**2**	11	*3.18*	2	**3**	13	*3.83*	2	**4**	17	*4.56*	5	**10**	43	*13.52*	7	**10.5**	19	*11.55*	9	**36.5**	516	*77.36*
#Active Commits	0	**0**	0	*0*	1	**1**	3	*1.40*	1	**2**	3	*1.76*	4	**7**	22	*8.52*	4	**6.5**	10	*6.30*	7	**22**	232	*43.95*
#Reeds	0	**0**	0	*0*	0	**0**	0	*0*	0	**1**	3	*0.84*	0	**0**	2	*0.17*	1	**1**	2	*1.40*	1	**5.5**	31	*7.32*
Turf commits	0	**0**	0	*0*	1	**1**	3	*1.40*	0	**1**	3	*0.92*	4	**7**	22	*8.34*	2	**5**	9	*4.90*	0	**18.5**	207	*36.60*
Table Insertions	0	**0**	0	*0*	0	**0**	2	*0.26*	0	**2**	18	*2.48*	0	**2**	6	*2.14*	0	**4.5**	16	*6.70*	0	**24**	301	*5.23*
Table Deletions	0	**0**	0	*0*	0	**0**	1	*0.09*	0	**1**	45	*3.88*	0	**0**	4	*0.66*	0	**2.5**	15	*4.45*	0	**9**	214	*25.64*
#Tables@Start	1	**2**	227	*14.26*	1	**3**	68	*5.94*	1	**4**	47	*6.60*	1	**5**	65	*8.31*	2	**8**	26	*8.90*	2	**20**	61	*24.18*
#Tables@End	1	**2**	227	*14.26*	1	**3**	70	*6.11*	1	**5**	18	*5.80*	1	**6**	68	*9.79*	2	**10**	33	*11.15*	1	**22.5**	135	*33.77*

Figure 5: Measurements per Taxon (min, **median**, max, *avg*).

4.2. The Frozen Land of Total Rigidity

"In a survey of 20 database administrators (DBAs) at three large companies in the Boston area, we found that ... DBAs try very hard not to change the schema when business conditions change, preferring to "make things work" without schema changes."[2] We have named this tendency as *gravitation to rigidity* in studies of smaller scale [21], [22], [23]. The most characteristic and undisputed finding of this study is the confirmation *with solid numbers over a very large dataset* of the above. Overall, *within the scope of FOSS projects, it is the absence of evolution of the logical level of the schema that demonstrates itself in large numbers, as opposed to its presence*. Out of the 327 repositories that we cloned, 132 (40%) had a single commit for their schema whatsoever, 34 (10%) had more than 1 commits, but zero changes at the logical-level schema, and 65 (20%) were almost frozen (with less than 4 active commits and 10 modified attributes). Overall, *70% of the projects, demonstrated total absence or very small presence of change.*

Interestingly, this absence of evolution is not a feature of abandoned projects, but rather an attitude of the developers: In terms of project duration (attn.: not schema update period), 68% of Frozen projects span more than 24 months, and

79% span more than 12 months. The respective numbers for Almost Frozen are 58% and 73%, respectively. The commits concerning the DDL file amounted to 6% and 5% of the total commits, respectively. Concerning change, the Almost Frozen category, which is the only one with some change, has a median of 3 commits, one of them active, a median of zero tables inserted and deleted (75% of projects having a flat schema line) and a median total change of 3 attributes.

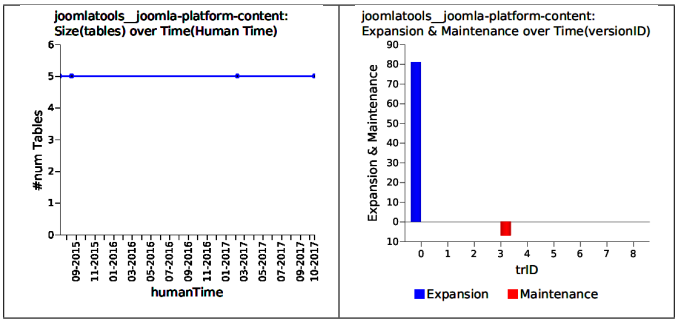

Figure 6: A typical example of an Almost Frozen schema: schema size over human time (left) and expansion vs maintenance over transitionId (right). There are 8 commits post the original version (mostly close to each other, thus overlapping each other on the left) and out of them, the only active commit involves the data type update of 3 attributes.

4.3. Hit and Freeze Evolution

There is a particular family of projects whose evolution demonstrates specific transitions with significantly higher amount of changes than the Almost Frozen taxon. The *Focused-Shot-and-Frozen* taxon is based around one or two such transitions, with often a single reed being practically the only change performed to the schema. The jRonak / Onlinejudge project in Fig. 7 depicts another form of schema evolution, again with a very restricted set of just a couple of active commits, and in this case, a small expansion of the set of tables of the schema.

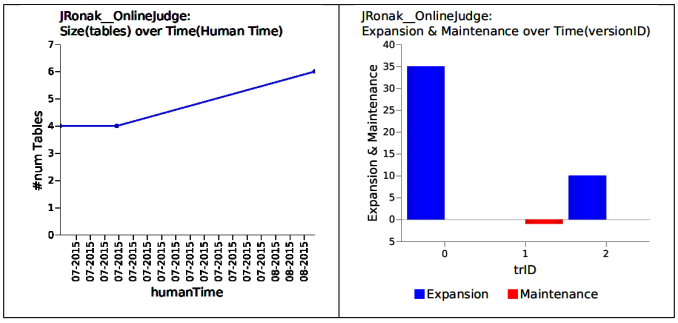

Figure 7: Example of a focused expansion of two tables

This taxon's schemata do not change significantly and the amount of change is small in terms of attributes. In 36% of the projects, evolution involves at-

tribute injections into existing tables (i.e., a flat schema line) and a small maintenance activity, all combined in a single active commit. In other words, a significant percentage of such projects simply focuses evolution in almost a single commit. 52% of the projects involve a single step-up in the schema line; on average the projects of the taxon insert 2 tables and remove 1 in their life.

Concerning project duration, out of the 25 projects, 9 lasted less than 1 year and another 6 less than 2 years. However, there are 11 projects (44%) which outlasted 2 years of project duration (PUP). In contrast, SUP has a median of 2 months and an average of 9 months. The commits concerning the DDL file amounted to 4% of the total commits.

4.4. The Timid Life of Moderate Evolution

Moderate evolution is the characteristic of 29 projects. There is a consistency in change, as the median Schema Update Period is 20 months with a median of 10 commits, 7 of them active, typically all of them turf (distinguishing the taxon from the subsequent ones), a median of two tables inserted and zero deleted and a median total change of 23 attributes. 65% of projects have a rise in the schema, 10% have a flat line and the rest of the projects have turbulent or dropping schema lines. In terms of project duration, 72% of the taxon's projects span more than 24 months, and 86% more than 12 months. The commits concerning the DDL file amounted to 5% of the total project commits.

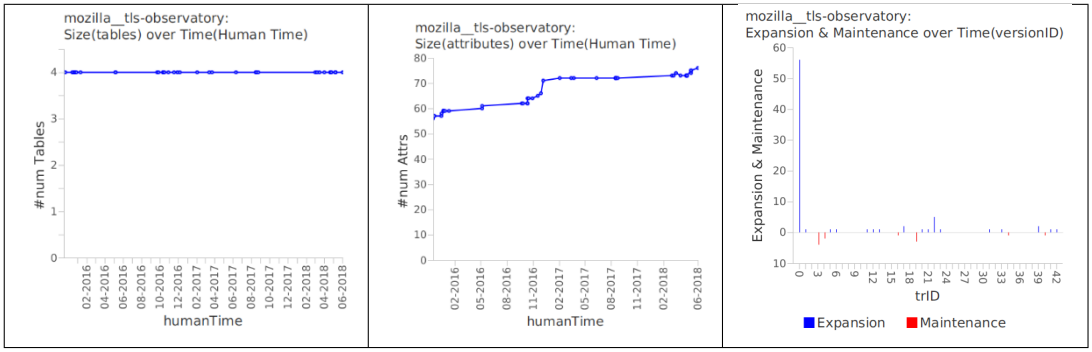

Figure 8: A typical example of a schema evolving with a moderate tempo: schema size over human time as tables (left), attributes (center) and heartbeat over transitionId for the mozilla/tls-observatory project. There are 43 commits (23 active) after the original version, mainly directed towards mild attribute injections, with different time density (as the middle figure demonstrates).

4.5. Focused Change and Turf

Apart from a moderate and low-volume evolution of the schema, there is a distinct subcategory of not-insignificant heartbeat with focused change (via a couple of reeds) and a few extra active commits. We refer to this taxon as *Focused Shot and Low*. The taxon is characterized by one or two reeds and no more than 10 active commits. Change in this category comes to a large

extent due to the " reeds" of focused activity, rather than the regular " turf-like" activity of small volume. Except for activity, the numbers are very similar to the moderate taxon: the median Schema Update Period is 17.5 months with a median of 10.5 commits, 6.5 of them active, with one reed, a median of 4.5 tables inserted and 2.5 deleted. This taxon is the second largest in activity volume, right after active projects, with a median total change rise to 71 attributes (significantly different from the previous taxa). In terms of project duration, 70% of the projects of this taxon span more than 24 months, and 75% span more than 12 months. The commits concerning the DDL file amounted to 6% of the total project commits.

SUP in this category comes in two flavors: short and long schema update periods. In Fig. 9, the jasdel/harvester project has a very short SUP, with a couple of reeds and a two-step increase in the schema line. The TalkingData/OWL-v3 project is quite expressive of the family: the "reed" reaches 124 attributes of growth and 68 attributes of maintenance, practically including 90% of the project's post-V0 activity.

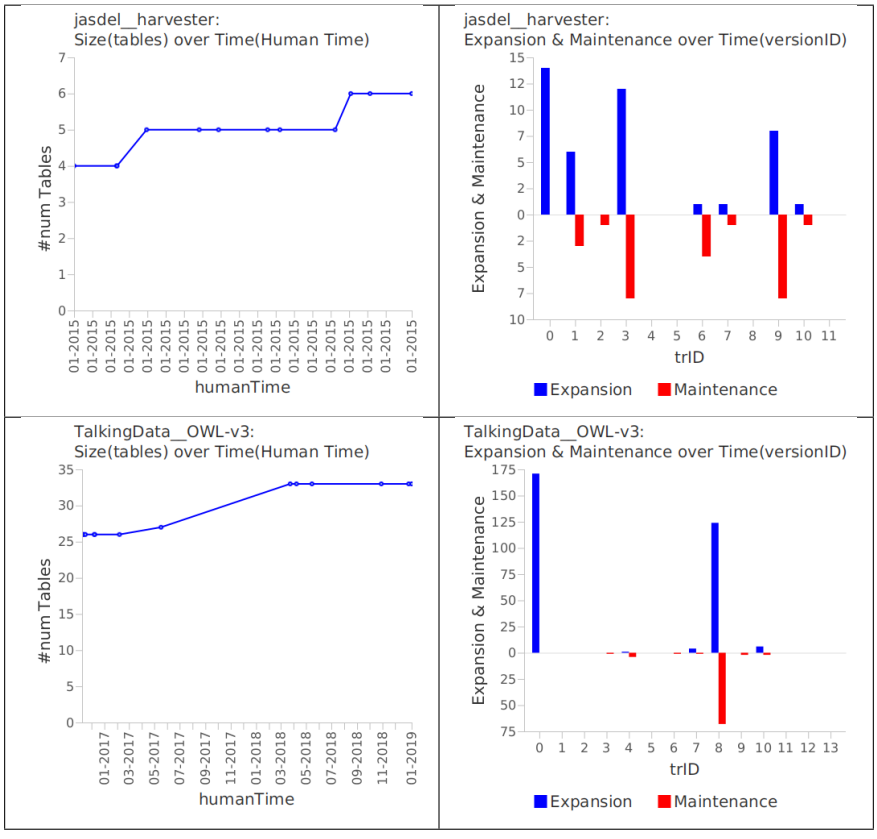

Figure 9: Examples of focused maintenance: a two-step schema increase accompanied with a few turf commits (top) and a very large reed, accompanied by very low change (bottom).

4.6. Schemata with high volumes of updates

Apart from the aforementioned, low-to-med activity taxa, there are also schemata that demonstrate *significant volume of updates*. The schemata of this taxon, although few, are the ones with the largest amount of tables involved, the largest SUP durations (31 months median), a heartbeat with a median of 36.5 commits, 22 active, 5.5 reeds and the rest turf, 24 tables inserted and 9 deleted and a median total activity of 254 attributes. In terms of project duration, 91% of the projects of this taxon span more than 24 months, and 95% span more than 12 months. The commits concerning the DDL file amounted to 6% of the total project commits. In other words, the behavior of the taxon demonstrates significantly higher volumes of change than the other taxa (Fig. 1, 2, 10).

It is important to note that in active projects, the heartbeat is not homogeneous. This has to do both with the frequency and the size of the change events. In terms of frequency, there are periods of systematic activity, with versions of small-to-medium size changes, periods of idleness, spikes of massive maintenance, growth and restructuring. The size of the schema is typically growing (50% of the cases with several steps, 9% with a single step), and, out of the 22 cases there are also 2 cases of flat schemata, 3 cases of massive drop of its size and 4 cases of turbulent evolution.

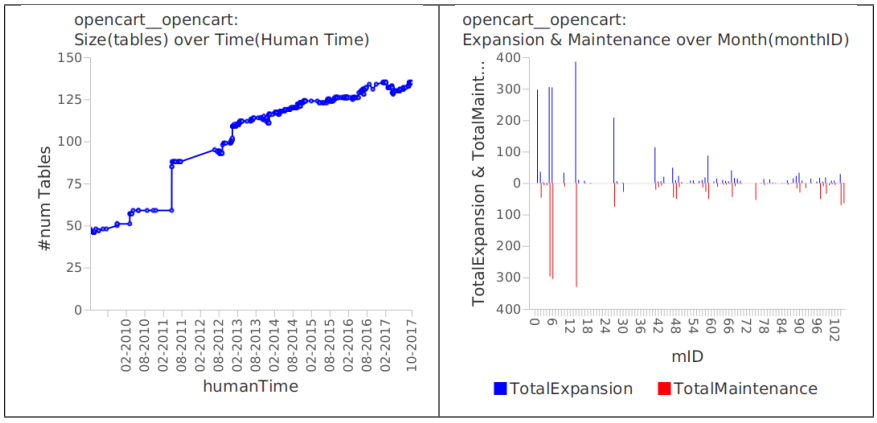

Figure 10: Example of high, systematic activity: the schema is being augmented over time via a systematic heartbeat that comes either with large spikes, or with constant turf and minor increases, without excluding periods of idleness. Observe that in this figure, the schema size is depicted over human time, whereas the heartbeat over the aggregated monthly changes. See Fig. 1 and 2, too, for more active schemata.

5. Taxa well-formedness

Are the taxa that we derived reasonable? Is it possible that two taxa actually hide the same behavior? To address these sanity-check questions, in this Section, we argue that our proposed taxa abide by three well-formedness criteria:

- Non-triviality: no taxon should include just a handful of projects, unless they are of extra-ordinary nature.
- Completeness: i.e., covering all possible cases of activity behavior
- Disjointness: the characteristics of the different taxa are different, and each project can belong to exactly one taxon
- Internal Cohesion: within a taxon, the behavior of its projects is similar

Non-triviality is covered by the fact that all taxa include at least 20 projects. It is easy to see that *Completeness* is covered both by the classification scheme of Fig. 4 at the logical level, and, as sanity check, by the projects at the instance level. Similarly, *Disjointness* is covered by mutual exclusion of the constraints of the classification scheme of Fig. 4, which dictate mutually disjoint taxa. Fig. 11 visually demonstrates why the rule-based classification makes sense, as the different taxa have a fairly small overlap. Proving *Cohesion*, however, is not trivial, and for this, we need to employ statistical tests.

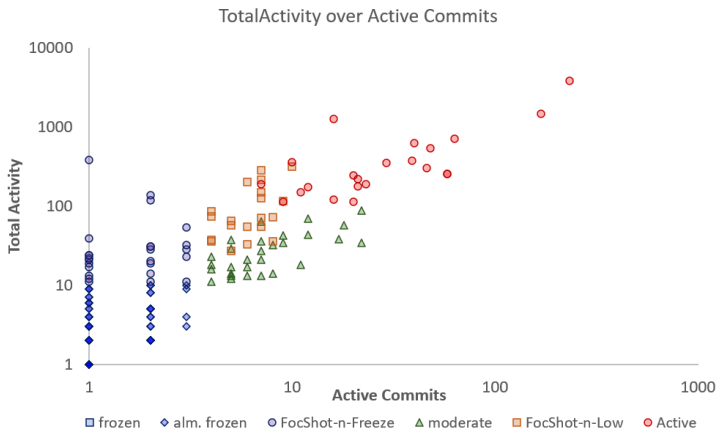

Figure 11: Project profiles in terms of active commits and activity. Frozen are not shown due to the logarithmic nature of the axes; almost frozen (blue diamonds) are lower left; focused shot & freeze (blue circles) are upper left; moderate (green triangles) occupy the center; focused shot & low (orange squares) complements moderate in the upper center with higher volumes of activity and moderate heartbeat; active projects (red circles) are at the upper right. Naturally, the borders are not completely separated; however, the original qualitative discrimination fits quite well with the rule-based discrimination of Fig. 4.

Statistical differences between taxa. To assess whether taxa actually form different "populations" of projects, we compared the derived taxa both (a) as a set of taxa over the entire data set, and, (b) pairwise, on whether their statistical characteristics qualify them to be different. To this end, we assessed statistical significance of the taxa differences over (i) their number of active commits and (ii) their total activity.

We employed the Kruskal-Wallis test, in R, to test the differences of the defined taxa. The null hypothesis of the test is that the different taxa have the same median and thus the reported p-value is a measure on the rejectability of the null hypothesis. We excluded the totally frozen taxon, which is practically a special case of the Almost Frozen, from this analysis. The overall assessment of the Kruskal-Wallis test for the entire data set for the activity measurements produces a Kruskal-Wallis chi-squared = 178.22, df = 5, p-value < 2.2e-16 and for Active Commits Kruskal-Wallis chi-squared = 175.27, df = 5, p-value < 2.2e-16. In other words, it is extremely improbable that the taxa represent similar behaviors.

Note that our data are not normally distributed: Shapiro-Wilk normality test on total activity produces W = 0.24386 and a p-value < 2.2e-16, i.e., it is extremely unlikely that activity data are normally distributed. Internally, within each taxon, the respective test revealed non-normality for all taxa for both active commits and total activity, with the exception of active commits for the case of Focused Shot and Low.

Extra statistical evidence. To reinforce the above conclusions, we performed extra statistical tests. First, we compared the taxa pairwise via a Kruskal-Wallis test. In Fig. 12, we report the p-values of the respective test: the lower left triangle refers to the active commits and the upper right triangle to the total activity. Assuming an acceptance threshold of 5%, *the test reveals that the differences between taxa are significant, with the exception of two cases*.

... wrt total activity

... wrt active commits

	Alm. Frozen	FShot+Frozen	Moderate	FShot+Low	Active
Alm. Frozen		1.730e-13	8.455e-15	1.141e-11	2.013e-12
FShot+Frozen	0.03199		0.7945	2.138e-05	6.076e-08
Moderate	3.714e-16	2.282e-10		5.406e-06	1.294e-09
FShot+Low	3.884e-13	7.043e-09	0.2796		1.855e-05
Active	7.204e-14	3.1103-09	5.355e-07	9.745e-08	

Figure 12: p-values of the Kruskal-Wallis test for the pairwise comparison of the taxa of our study: the lower left triangle refers to the active commits and the upper right triangle to the total activity values.

More concretely, the only cases where we cannot reject the null hypothesis are (a) the similarity of moderate with focused shot and frozen for their activity, and (b) the similarity of moderate with focused shot and low for their active commits. For both cases, however, the two involved taxa demonstrate significant difference in the other measures. Specifically:

- For the case of Moderate with Focused Shot and Frozen: whereas the overall number of affected attributes appears to be similar, the number of active commits is significantly different - in other words, whereas the Moderate schemata demonstrate a turf-oriented, more frequent evolution activity, the Focused Shot and Frozen schemata group a "similar" volume of activity in no more than 3 active commits. Thus, the difference lies in the active commit part.

- For the case of Moderate with Focused Shot and Low: whereas the overall number of affected attributes appears to be quite different, and the Focused Shot and Low schemata demonstrate significantly higher amounts of activity, the number of active commits is similar: in other words, the schemata of moderate activity lack the reeds of high activity that the Focused Shot and Low schemata demonstrate, and who drive their activity to larger heights. This resonates quite well with our intuition of treating Focused Shot and Low as a special case of Moderate evolution in terms of heartbeat, but whose reeds reach high activity volume.

In other words, we see that there is not a single pair of taxa that demonstrates similar behavior in both the number of active commits and total activity. Based, thus, on all the above discussion, we argue that the taxa that our analysis derived are pairwise different.

Active Commits	Alm. Frozen	FS_Frozen	Moderate	FS_Low	Active
MIN	1	1	4	4	7
Q1	1	1	5	5	15
Q2	1	2	7	6,5	22
Q3	2	2	10	7	50,5
MAX	3	3	22	10	232

Activity	Alm. Frozen	FS_Frozen	Moderate	FS_Low	Active
MIN	1	11	11	27	112
Q1	1	15,5	15	41,5	177
Q2	3	23	23	71	254
Q3	5	31,5	37,5	143	558,5
MAX	10	383	88	315	3845

Figure 13: Quartiles of activity and active commits for the different taxa.

To further strengthen the statistical evidence, we also computed the quartiles of total activity and active commits for each taxon (Fig. 13). We also depicted these numbers in the double box plot shown in Fig. 14. The separation of the taxa demonstrates overlaps and "grey zones", however, the shape and placement of the boxes is quite reassuring. Specifically:

- The active taxon is very far from the rest. They are just 22 projects, but really far apart from the other taxa (see the legend of the Fig. 14 for details)
- The frozen taxa are quite close (also by definition) one to another. However, due to their larger populations, the separation can be justified: frozen with zero active commits and activity are by definition a taxon of its own. The most populous, Almost Frozen, is really close; however, we discriminate it from Frozen as having even such a small activity. Focused Shot and Frozen is also close both to Almost Frozen and to Moderate, however the shape of its heartbeat is quite different from both.

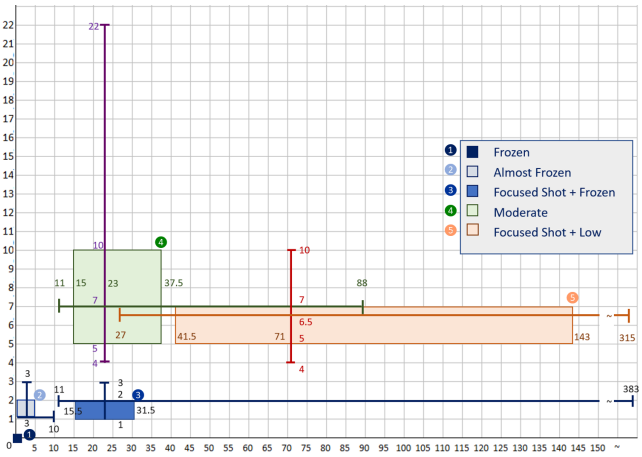

Figure 14: Double box plot for active commits and activity for the different taxa. The horizontal axis depicts the total activity (in number of affected attributes) and the vertical axis depicts the number of active commits. Each taxon has a rectangle with the Q1 and Q3 quartiles at its edges, for both dimensions. A cross formed by lines passing from the Q2 (median) for each dimension annotates the box of each taxon. The min and max values of each taxon for the respective dimension mark the limits of each line. For example, for the moderate taxon, activity has: min: 11, Q1: 15, Q2: 23, Q3: 37.5, max 88, and, active commits have: min: 4, Q1: 5, Q2: 7, Q3:10, Q4: 22. The active taxon is not shown, as its activity lies far away from the rest: Q1: 177, Q3: 5585 and its active commits with Q1: 15 and Q3: 50.5.

Cohesion revisited. Is it possible that a taxon is not cohesive? The cohesion of the taxa is demonstrated by the double box plot of active commits vs activity, for the proposed taxa. The following observations are due:

- All taxa are heavily biased towards lower values, for both active commits and activity.
- The 3 most frozen taxa are really clustered in very cohesive boxes. This has to do both with the box and the whiskers of the box plot, with one exception: the 6 projects of the upper quartile of Focused Shot and Frozen, that spans from 31.5 attributes to 383. This should also be assessed under the prism of their population: there are 34 Frozen, 65 Almost Frozen (largest category and smallest distribution of all), 25 Focused Shot and Frozen projects, which means that there is almost an inverse relationship between population and surface of the box in the plot (remember that Moderate projects are 29, Focused Shot and Low are 20, and Active are 22 projects).
- The only box that spans a significant amount of surface in Fig. 14 is the taxon with the smallest population, Focused Shot and Low. However, there is no other taxon really, around the area of its high values, (there are only 3 Focused Shot and Frozen projects above 55 attributes of activity, and the entire taxon of Active, however at different areas of active

commits, i.e., height in the chart).

- Finally, the Active taxon,with its 22 projects of high activity, is very far apart from the rest. Both the location of the projects in the 2D area of Fig. 14, and its statistical tests, emphasize its separation from the other taxa.

6. How do schema evolution taxa and evolution-related properties relate?

In this section, we discuss the correlation of the taxa with schema evolution measures. This knowledge can be exploited in two ways: (a) understanding what features can determine the taxon of a project, and (b) given the taxon of a project, predicting the way its evolution-related properties will unfold. The research (meta)question addressed is:

RQ3. What are the quantitative characteristics of schema evolution and how do they perform for different taxa?

6.1. Durations and Taxa

Before proceeding with the evolutionary properties, we make a small inquiry to clarify the landscape concerning the durations of the projects involved in the study. Specifically, we are interested on how the Project Update Period and Schema Update Period of an evolving schema can relate to its taxon.

Research Question: do taxa come with significant differences to their projects' Schema and Project Update Periods?

6.1.1. Project Update Period

The Project Update Period is the time between the first and the last commit of the life of the project that we have recorded.

In Figure 15, we can see the boxplot of PUP per taxon. Although the total span of the projects' PUP differs per taxon, due to the presence of outliers, it is very interesting, that *overall, the mean values and the boxes of the different taxa with respect to their PUP are very similar!* The IQR range is around 3 years and the mean ranges between 30 and 43 months for the first 5 of the six taxa. The only taxon that slightly deviates from this rule is the Active taxon, whose box is shifted towards higher values (still significantly overlapping with the others) and its mean being 65 months. *In other words: the behavior of the evolution of the entire project in terms of its time span, does not depend on the taxon; moreover, the schema evolution of a project, does not seem to depend on the project evolution, but rather, it has a character of its own.*[8]

[8] As a side-note: no taxon is based on dummy projects (for example, the IQR of the PUP for the Frozen taxon, comes with a 25% percentile of 23 months!)

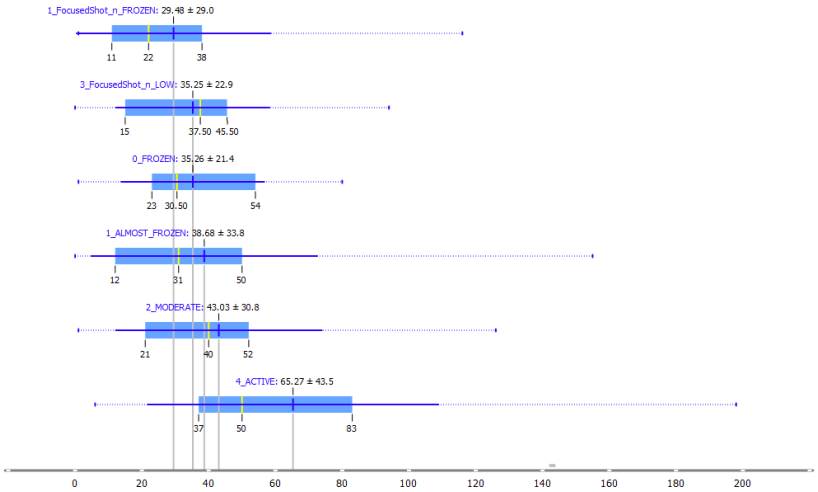

Figure 15: Boxplot of Project Update Period in months (PUP) over taxon.

In Figure 16 we depict the breakdown of projects per taxon and PUP, having grouped the PUP in 2-year periods. If we compare the three 2-year periods with each other, the distribution among different taxa looks fairly similar (with small differences). If one compares the different taxa with one another, there are several behaviors. The moderate activity taxa (Moderate and Focused Shot and Low) have a bell-like shape and are mostly biased to medium PUP durations. Frozen and Active are mostly biased towards the 4 Years+ category. The two of the three "frozen" taxa, Almost Frozen and Focused Shot and Frozen are mostly biased towards shorter PUP durations. See also how high the 4-Years+ category climbs for Frozen and Almost Frozen.

6.1.2. Schema Update Period (SUP)

The Schema Update Period (SUP) is the duration between the first commit that defines the database schema in the DDL file and the last commit that concerns the DDL file. Obviously, SUP is a subset of the lifetime of the project. Figure 17 presents the distribution of SUP values (in months) per taxon in the form of a boxplot. The plot indicates that the 3 frozen taxa are pretty close in terms of behavior, with an average of 8 - 12 months of SUP; the Focused Shot and Low and Moderate taxa come with a similar "box" in the box plot, between 0 and 3 years, and average SUP 21 and 23 months, respectively; finally, Active is a shifted towards higher durations with an average of 36 months of SUP (although not entirely different from the 2 aforementioned taxa, in terms of spread). Lessons learned:

- *It is interesting, that, overall, all taxa reach fairly high durations for the schema update period.*

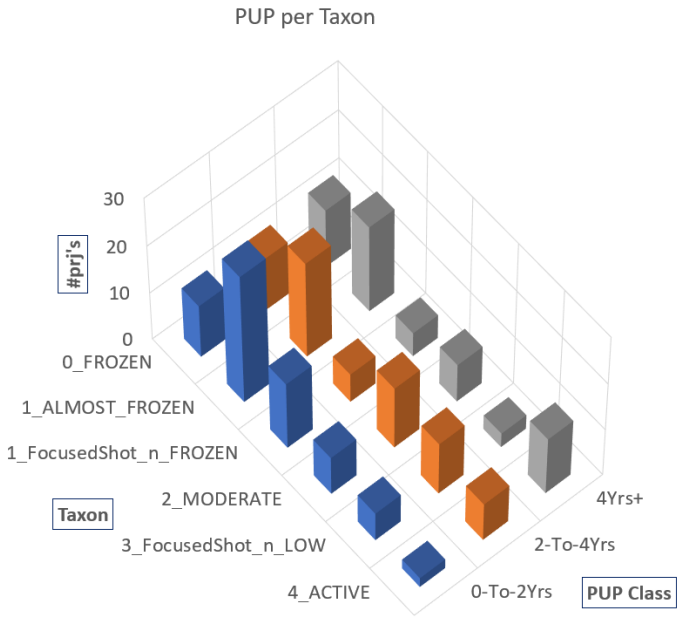

Figure 16: Project Update Period (PUP) over taxon.

- *All taxa have broad IQRs that range between 1 and 3 years, meaning that similar amount of change is produced with different time distribution within the same taxon.* Thus, the lesson learned is, that even the non-active taxa have quite long ranges of SUP and PUP, meaning that ultimately absence of change is not due to absence of activity in the project.

- *Finally, it is interesting that the differences between taxa are evident (more than PUP) but not big. Progressively, box plots and medians shift to the right, as the taxon becomes more active; however, the overlap of the IQR boxes is significant, and medians cluster in groups.* To a certain extent, progression towards higher SUP for "hotter" taxa is expected: whereas long duration can -and frequently does- have a small volume of change in the schema, high activities presuppose longer schema update durations, whenever change happens with regularity and in small turfs.

Figure 18 shows how taxa and SUP durations interrelate. The Active taxon is biased towards longer Schema Update Periods. Moderate and Focused Shot and Low projects are mostly biased towards medium SUP ranges, whereas "frozen land" is mostly oriented to small SUP periods, with the prominent exception of 19 out of 65 Almost Frozen projects, that have almost frozen SUPs between 1 and 3 years. Compared to Figure 16, where the evolutionary behavior is not particularly related to project duration, Figure 18 depicts a different picture, where there is a tendency towards lower SUP durations for lower taxon activity.

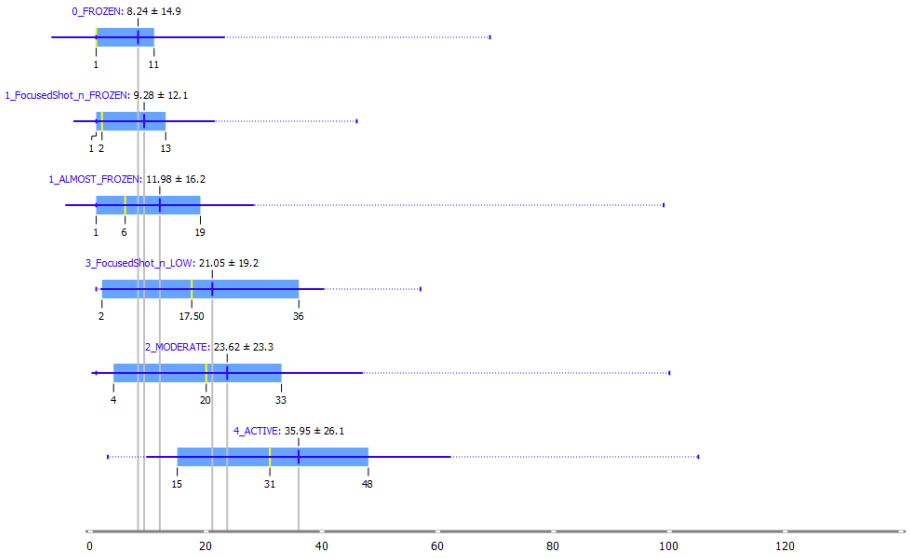

Figure 17: Boxplot of Schema Update Period in months (SUP) over taxon.

6.1.3. How do PUP and SUP relate?

We have also explored the joint distribution of PUP and SUP, and we present the results in Figure 19. The columns of the Figure represent completed years for the Schema Update Period, whereas the rows of the figure represent the completed years for the Project Update Period of the monitored projects. The upper right triangle could only be empty, of course. In black, bold font, we depict the marginal sums. In the more extreme column and row we depict cumulative sums that we will explain in the sequel.

The results show a single cell, [0,0] with a 20% of projects that could be eligible for an invalidation test (the difference with the 21% of Figure 19 is due to the limit: one is <=12 months and the other is < 12 months). The rest of the projects exceed 1 year of PUP.

Coming to the cumulative sums, the situation is inverse in terms of how the distribution of projects behave, and thus the point of having an inverse cumulative sum for PUP and a cumulative sum for SUP.

- The inverse cumulative sum for PUP starts counting at projects with the maximum PUP (16 years) and ends at the 20% of projects with 0 completed years of PUP.
- The cumulative sum for SUP starts with the 59% of projects having less than one completed year of SUP and ends at the maximum SUP of 8 completed years.

Observe the inverse cumulative sum for ***Project* Update Period**. 80% of the

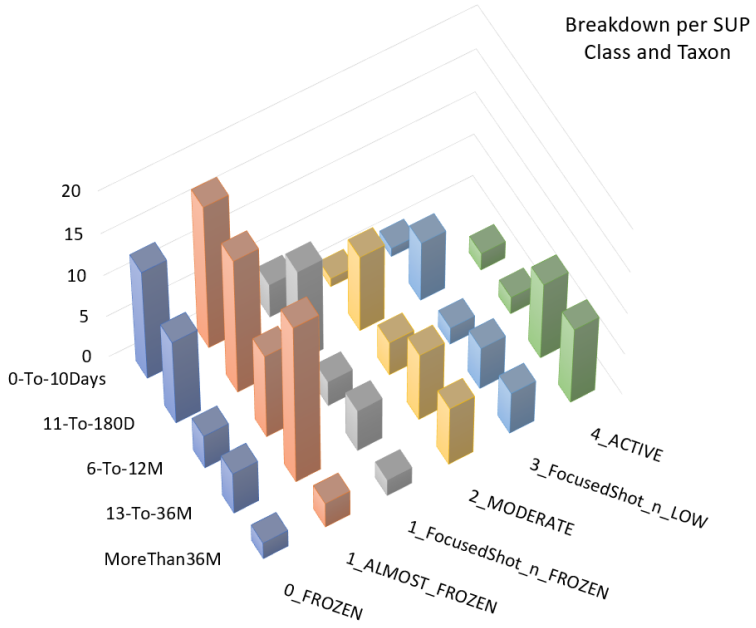

Figure 18: Schema Update Period (SUP) over taxon.

projects have project duration greater than 1 year, 66% of the projects have a duration greater than 2 years, *48% (almost half!) the projects have a duration greater than 3 years*, etc. *Overall, the projects are of significant durations*. As already mentioned, this is also a clear statement of the validity of the data set, as it includes projects that were being updated for a long period of time.

At the very same time, for ***Schema* Update Period**, 59% of the projects have less than 1 year of Schema Update Period, 74 less than 2 Years, etc. In other words, we have practically the inverse distribution of values compared to Project Update Period, grossly biased towards short values. *In other words, whereas the projects kept having updates for long periods of time, the schema updates were focused in much shorter periods!*

6.2. Active Commits, Turf and Reeds

Research Question: How do taxa relate to the characteristics of the heartbeat? Can we characterize taxa via heartbeat attributes? Can we know the value range of heartbeat attributes given the taxon of a project?

The amount of evolutionary activity, in combination with heartbeat properties was the main driver of the determination of the different taxa. However, as we shall demonstrate, here, reeds and turf can solely be used to determine the taxon of a project, even without the usage of its evolutionary activity. This property is analyzed in more detail in this subsection.

For 195 prjs SUP Years →

PUP Years ↓	0	1	2	3	4	5	7	8	Grand Total	Inv. Cumul. sum
0	20%	0%	0%	0%	0%	0%	0%	0%	20%	100%
1	9%	5%	0%	0%	0%	0%	0%	0%	14%	80%
2	11%	5%	3%	0%	0%	0%	0%	0%	18%	66%
3	8%	2%	4%	2%	0%	0%	0%	0%	14%	48%
4	6%	2%	2%	2%	2%	0%	0%	0%	13%	33%
5	3%	1%	1%	1%	1%	1%	0%	0%	6%	20%
6	2%	0%	1%	0%	1%	1%	1%	0%	5%	14%
7	0%	1%	0%	1%	1%	1%	0%	0%	3%	9%
8	0%	1%	1%	1%	0%	0%	0%	0%	2%	7%
9	0%	0%	0%	1%	0%	0%	0%	0%	1%	5%
10	0%	1%	0%	0%	1%	1%	0%	1%	3%	4%
11	1%	0%	0%	0%	0%	0%	0%	0%	1%	2%
12	0%	0%	0%	0%	0%	0%	0%	1%	1%	1%
16	0%	1%	0%	0%	0%	0%	0%	0%	1%	1%
Grand Total	59%	15%	11%	6%	4%	3%	1%	2%	100%	
Cumul. Sum	59%	74%	85%	91%	95%	98%	98%	100%		

Figure 19: Joint distribution of quantized PUP and SUP in completed years; the year labels indicate round-down lower limits – e.g., a label of 0 indicates a period of [0 months .. 12 months), a label of 1 indicates a period between [13 months. . . 24 months), etc. The percentages are all with respect to the 195 projects of the dataset.

Concerning **active commits**, we report their breakdown in the different taxa in Figures 20 and 21. Figure 20 presents the percentages of the active commit classes per taxa where the following discretization has taken place: (a) 0_NONE, with zero active commits, (b) 1_TOO_FEW with active commits in the range [1 - 3], (c) 2_FEW, with active commits in the range [4 - 10], (d) 3_MODERATE, with active commits in the range [11 - 15], and, (e) 4_SEVERAL, with active commits > 15.

Taxon	0_NONE	1_TOO_FEW	2_FEW	3_MODERATE	4_SEVERAL	*Grand Total*
0_FROZEN	100%	0%	0%	0%	0%	*100%*
1_ALMOST_FROZEN	0%	66%	34%	0%	0%	*100%*
1_FocusedShot_n_FROZEN	0%	44%	56%	0%	0%	*100%*
2_MODERATE	0%	0%	0%	76%	24%	*100%*
3_FocusedShot_n_LOW	0%	0%	0%	100%	0%	*100%*
4_ACTIVE	0%	0%	0%	14%	86%	*100%*
Grand Total	*17%*	*28%*	*18%*	*23%*	*13%*	***100%***

Figure 20: Taxa and Active Commits class values

Figure 21 also offers the boxplot of the active commits with respect to the different taxa. Both representations suggest that *the taxa are fairly different concerning the active commits, and as the taxon becomes more active, all the related measures increase: the mean, the median, the IQR and the range of the active commits*.

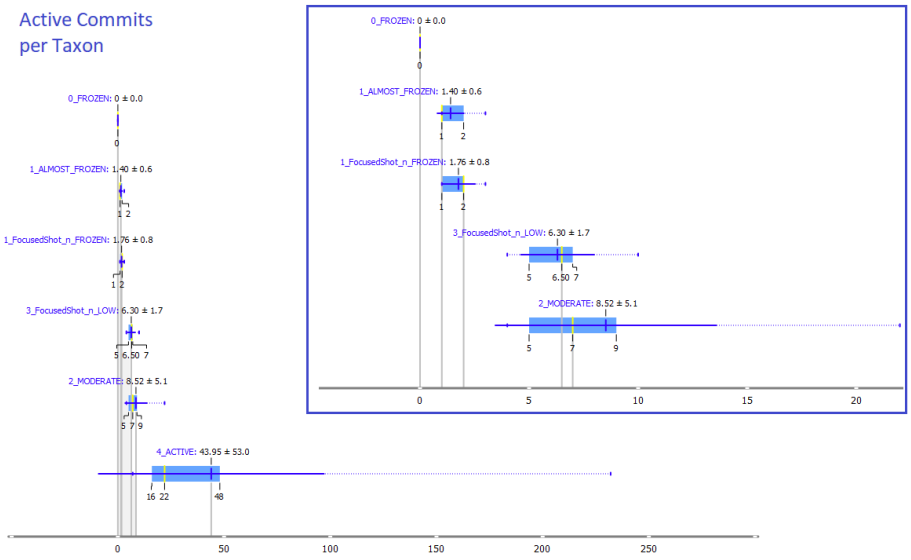

Figure 21: Boxplot of Active Commits per Taxon

At the same time, we can observe that there are two cases that bear similarities. The two of the three frozen taxa, share the same IQR, although the median of Almost Frozen is on the edge of the box of Focused Shot and Frozen, and, as we shall see, they differ a lot in terms of reeds. The Moderate and Focused Shot and Frozen have a rather large difference in their spread and their means, although their medians are close and their IQR boxes have a fair amount of overlap (again, as we shall see, later they differ a lot in terms of reeds).

Active commits and Reeds. Figure 22 codifies the relationship between active commits and reeds in particular. The basic idea is that (with very few outliers) taxa contain different combinations of activity and focused activity as (codified by reeds); at the same time, increased values in the combination of heartbeat features progressively lead to more active taxa.

- The 3 "Frozen" taxa operate in very few active commits and necessarily no more than one reed at most.
- The Moderate taxon is mostly in the range of moderate numbers of commits with zero reeds, i.e., demonstrates a strong turf-oriented character (which has nothing to do with its definition). The Focused Shot and Low, is the only one determined by heartbeat at its definition which imposes moderate commits and a couple of reeds.
- The Active taxon works with several or excessive number of reeds in mostly large numbers of active commits.

Active Commit Class Values \ *Reeds*	NONE	SINGLE	DOUBLE	SEVERAL [3 .. 10]	EXCESSIVE >10
NONE (0)	**Frozen (34/34)**				
TOO FEW (1)	**Almost frozen (43/46)**	**FS Frozen (8/8)**			
FEW [2 ..3]	**Almost frozen (22/26)**	**FS Frozen (8/8)**			
MODERATE [4 ..10]	**Moderate (22/22)**	**FS Low (12/12)**	**FS Low (8/8)**	Active (3/3)	
SEVERAL >10	Moderate (4/4)	Active (2/3)	Moderate (2/3)	**Active (13/13)**	Active (3/3)

Figure 22: Taxa determination on the grounds of Reed Class and Active Class (bold denotes large concentrations)

Lesson learned: Active Commits and Reeds alone can be used as the main determinants of the taxon of a project.

Turf and Reeds. Due to the very nature of turf (which is extremely close to Active Commits with very few exceptions), we can make the same claim with the combination of Reed and Turf. *Lesson learned: Turf and Reeds alone can be used as the main determinants of the taxon of a project.*

Turf Class Values \ *Reed Class*	NONE	SINGLE	DOUBLE	SEVERAL [3 .. 10]	EXCESSIVE >10
NONE (0)	Frozen (34/34)	FS Frozen (8/8)		FS Frozen (1/2)	
TOO FEW (1)	Almost Frozen 43/46	FS Frozen (6/6)			
FEW [2 ..3]	Almost Frozen 22/26	FS Low (3/5)	FS Low (3/3)		
MODERATE [4 ..10]	Moderate (22/22)	FS Low (9/9)	FS Low (5/6)	Active (4/4)	
SEVERAL >10	Moderate (4/4)	Active (2/3)	Moderate (2/2)	Active (11/11)	Active (3/3)

Figure 23: Taxa determination on the grounds of Reed Class and Turf Class

Figure 23 tells the story for the determination of taxa with the usage of only turf and reed class-encoded attributes (to be accurate: in their post-V0 version). Again, activity is not a factor in the classification. However, we can observe quite separated regions in the space of $reeds \times turfs$, with a small exception between Almost Frozen and Focused Shot and Frozen. To the extent that the information used in the $reeds \times turfs$ was not used for the formation

of the taxa, the figure is a major supporting factor on the separability of the proposed taxa, and a nice guide to predict properties of the evolution, given the taxon of a project.

6.3. Activity: how is activity related to taxa as well as to other measures?

Research Question: Is Total Activity different per taxon? Do taxa differ with respect to the range of values of schema evolution total activity their projects produce?

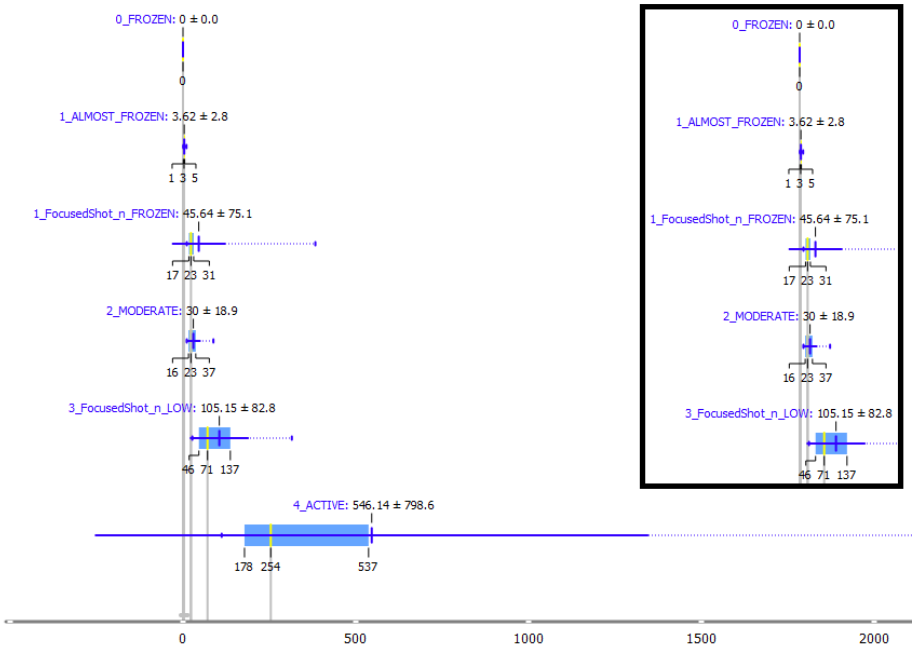

Figure 24: Boxplot of Total Activity (and zoom in 5 of the less active taxa)

Figure 24 depicts the total activity boxplot with respect to the different taxa. As Activity is used extensively (although not solely) for discriminating the taxa, it appears that total activity is progressively increasing while moving from towards more active taxa. The only projects with a very similar boxplot are Focused Shot and Frozen and Moderate; however, by definition, their heartbeat is significantly different.

As a side effect of the results presented here, and actually summarizing the overall "profile" of the taxa from various categories, we can claim that the taxa provide an *Inactivity Hierarchy*, with the order of levels being the one depicted in Figure 24, and its perfect inverse, a *Liveliness Hierarchy*. The higher level of the liveliness hierarchy is the Active taxon and the lower is the Frozen one.

Lesson learned: although exceptions do exist, the more lively a taxon is in the liveliness hierarchy, the higher the amount of activity it entails.

Activity Class		
None	Total Activity = 0	(34 projects)
Almost Frozen	Total Activity in [01..10]	(65 projects)
Small	Total Activity in [11..30]	(35 projects)
Moderate	Total Activity in [31..90]	(29 projects)
High	Total Activity higher than 90	(32 projects)

Table 2: Activity class definition

Activity class and Heartbeat Information. The *activity class* is an encoded representation of the total activity of a project. The definition of the ordinal domain of total activity is presented in Table 2.

Activity class and heartbeat can be very well correlated. Figure 25 summarizes (with a few approximations) how we can estimate activity class from the turfs and reeds of the heartbeat (we use turf ratio as it is slightly better a classifier than reed-class; had we used turf class instead, the boxes of the decision tree would be identical, with a little bit lower precision).

	Active Commits	Reed Class	Turf Ratio
Frozen	NONE		
Almost Frozen	TOO FEW	NONE	
	FEW		
Small	TOO FEW	SINGLE	
	FEW		
Moderate	MODERATE	SINGLE	<=0.833
	MODERATE	DOUBLE	<=0.6
	SEVERAL	NONE	
High	MODERATE	SINGLE	>0.833
	MODERATE	DOUBLE	>0.6
	MODERATE	SEVERAL	
	SEVERAL	> NONE	

Figure 25: A decision tree on how Activity Class is dissected by Heartbeat Information

Differences between activity classes: We observe that *Almost Frozen* differs from *Small* with respect to the existence of a reed. *Small* differs from Moderate with respect to the volume of active commits. *Moderate* differs from *High* with respect to the turf ratio. Overall, although all activity classes share some similarity with their neighbors, we see that, effectively, activity class is also related to the internal characteristics of how the change volume was achieved, i.e., to the extent of active commits, reeds and turf.

Lesson learned: three heartbeat attributes with the appropriate rules are adequate to characterize the volume of activity of a project.

Activity class and Commit Information. We can predict the activity class of a project with very high accuracy given only information related to the commits and the rates concerning commits (Fig. 26). Observe how the volume of the evolution is related to the frequency rates of the schema and the entire projects. Active commits give a good separation of the taxa; for the only pair of taxa that could be overlapping –*Small* and *Moderate* that is– the maintenance rate clears the potential conflict.

	Active Commits	Exp. Rate Per Commit	Mntnc. Rate Per Commit	Commits	Schema To Prj Commits
Frozen	0				
Almost Frozen	<=3	<=2			
Small	<=3	>2	<=8.25		
	4..11		<=1		
Moderate	>11		<=1		
	>3	<= 6.66	>1	<=12	
	>3	<=2.05	>1	>12	<=0.06
High	otherwise				

Figure 26: How activity class relates to commit information.

	Active Commits	Attr. Ins With New Tables	Attr. Injected	Attr. Del. With Del. Tables
Frozen	0			
Almost Frozen	<=3	in [0.. 7]		
Small	<=3	in [8..16]		
	>3	in [0..16]	<=9	<=5
High		> 16		> 21
		> 46		<=21
Moderate	otherwise			

Figure 27: How activity class relates to activity breakdown.

Activity class and Activity breakdown. We can relate the activity of a project to the breakdown of its components (Fig. 27). Few commits drive activity volume to small or almost frozen heights. If attribute insertion with new tables does not exceed 16 attributes, it is very rare to escape *Small* class (and consequently, its upper limit of 30 attribute changes). Hight activity on the other end of the spectrum comes with either large numbers of deleted attributes in table removals (larger than 21 attributes), or very high numbers of newly born attributes (larger than 46). All other projects come with moderate activity.

Lesson learned: Given the breakdown of change, we can predict where the activity class of the project lies.

6.4. Study of Table Insertions and Deletions

In this section, we discuss the study of the change of the number of tables in the schema. We define as *TotalTablesDelta* the total number of such table

insertions and deletions in the entire lifetime of the project:

$TotalTablesDelta = TotalTablesInserted + TotalTablesDeleted$

Observe that the measure reports on the total change of the involved tables in the schema and not the change in the schema size (for example, if 3 tables are inserted and 5 are deleted, the measure reports 8 and not -2).

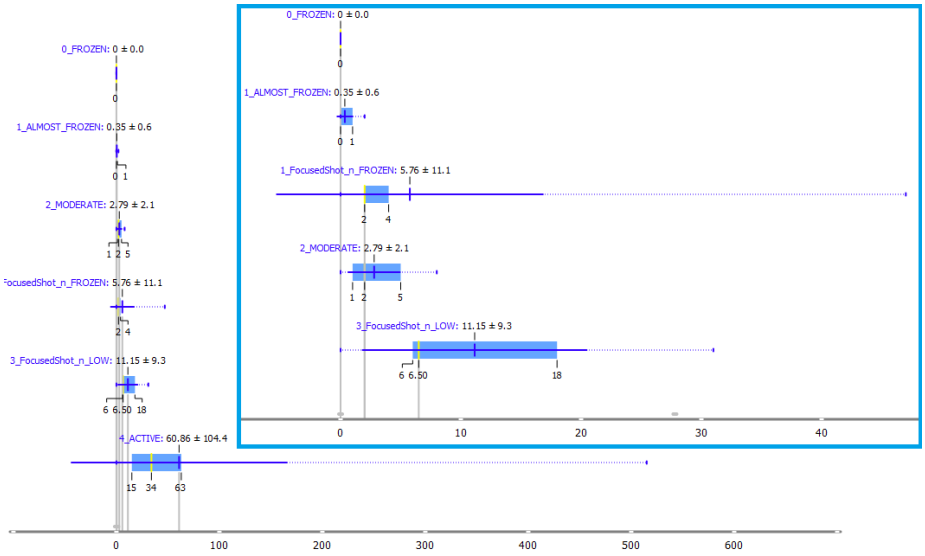

Figure 28: Boxplot of *TotalTablesDelta* per taxon (a) for all taxa, (b) zoom-in for all taxa but the active one.

Research Question: how is the amount of change in a schema's tables related to the taxon of a project?

Figure 28 depicts the box plot of *TotalTablesDelta* for the different taxa. We can make the following observations from Figure 28:

- With the exception of FocusedShotAndFrozen and Moderate, the boxes do not overlap! Practically, *taxa demonstrate different behavior with respect to the number of tables removed or added to the schema*.
- *There is a clear progressive shift towards higher amount of change as the taxon becomes more active*. Naturally, the Frozen taxon demonstrates no change. The Almost Frozen taxon restricts itself to practically no more than a single table change. The FocusedShotAndFrozen taxon demonstrates a variable behavior with 2 tables change as a median, but 5.76 as an average (i.e., despite its narrow number of interventions, change can indeed reach higher volumes). The Moderate taxon, with its mild evolution, expectedly, lies between 1 and 5 changes in the table roster, with a median of 2. On the other hand, FocusedShotAndLow exhibits much larger volumes of change than all the rest, with 6 to 18 tables changed. Finally,

the Active taxon is even more volatile, with its box ranging between 15 and 63 tables.

Overall, we can answer the research question by saying that with the exception of the Focused Shot And Frozen and Moderate taxa, each taxon demonstrates a different behavior, which progressively increases with the increase in the liveliness rank of the taxon.

7. Super Taxa: an opportunity for a cleaner separation of projects

In this section, we introduce the notion of super taxa, which are generalizations of the taxa presented in [1] and were generated by the merge of similar taxa into larger groups. We discuss how these super taxa are related to the heartbeat, the activity, the table-level activity measurements, as well as the durations.

Research opportunity [RQ4] Can we do better than grouping projects in taxa of schema evolution?

Super-taxon	**Taxa**	**Super-Taxon Characteristics**
COLD	Frozen	Low total activity in average.
	Almost Frozen	At most 4 active commits.
	Focused Shot and Frozen	
MILD	Moderate	More than 4 active commits.
	Focused Shot and Low	Majority of projects has no more than 3 reeds.
HOT	Active	More than 4 active commits.
		Majority of projects has more than 3 reeds.

Table 3: Table of Super Taxa with their characteristics

7.1. The possibility of deriving super taxa

The different taxa of schema evolution provide a first stepping stone towards the classification of projects into families of similar evolutionary behavior. The different taxa have different behavior with respect to a large spectrum of properties, concerning the rate (heartbeat), the volume (activity), and, to a lesser extent, the duration of change. These differences are also reflected at even more detailed features than the aforementioned ones, like the subcategories of activity and the breakdown in terms of table and attribute injection, ejection,

and update. Although the taxa provide a classification that is rationally and numerically justified, it is also clear that certain taxa are more similar than others, and despite their differences, also share similarities in terms of the statistical profile of the aforementioned properties. Thus, we have the potential of providing *super taxa*, i.e., larger groupings / families of taxa with coarser characteristics, but at the same time, (i) fewer in number, (ii) better separated in terms of statistical profile, and, therefore, (iii) more intuitive and easy to use. Interestingly, these super taxa can easily be produced by grouping the existing taxa into larger groups. Table 3 reports on how the original taxa of schema evolution are merged in order to produce the super taxa.

COLD. This super taxon was generated by the merge of Frozen, Almost Frozen and Focused Shot and Frozen classes. The individual classes were determined to be merged, because of (a) primarily, their small number of active commits to the DDL file, as well as (b) their small, in average, total activity. Regarding the child taxa, the Frozen taxon consists of projects with zero total activity, the Almost Frozen taxon consists of projects with small total activity and a small number of active commits, and the Focused Shot and Frozen taxon consists of projects with a small number of active commits but higher total activity, which was caused by a couple of focused-shot big commits. Again, these taxa are very similar to each other and their common line is their "cold" behavior (zero to a few active commits). The reader who refers to Figures 22 and 23 can observe that the three taxa are similar, although not identical in their heartbeat, occupying adjacent positions in the respective tables. Their behavior in terms of activity and table delta is also similar although not identical. Overall, the quantification of the behavior of these taxa suggests that although not identical, they are close to each other and quite far from the rest of the other taxa.

MILD. This super taxon was produced by the combination of Moderate and Focused Shot and Low, selected to be merged due to their medium to high activity. As far as the child taxa are concerned, the Moderate taxon consists of projects with medium total activity and a medium number of active commits, and the Focused Shot and Low consists of projects with medium to high total activity and a medium number of active commits. The reader who refers to Figures 20, 21, 22 and 23 can observe that the IQRs of their box plots overlap with respect to active commits (also: turf, but not with respect to reeds), while the IQRs of the boxplots for activity and table delta are discrete, but quite neighboring at the same time (Figures 28 and 24). Overall, the taxa merged are similar with respect to heartbeat, and complementary neighbors in terms of activity and focused activity.

HOT. This super taxon HOT is the same as the original Active taxon, which differs a lot from the other taxa and consists of projects with extremely high activity during their lifecycle. There is practically no feature where this taxon does not show its difference from all the rest. For this reason, we decided to let this taxon the same as the original Active taxon.

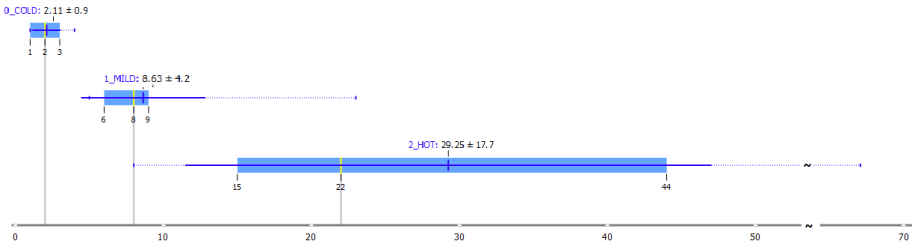

Figure 29: Box plot of Active Commits for Super Taxa

Measure	COLD	MILD	HOT
Reeds	0.73	1.63	6.25
Turfs	1.38	7.00	23.00
Active Commits	2.11	8.63	29.25
ActiveCommitRatio	0.66	0.72	0.73
ReedRatioAComm	0.39	0.22	0.29
TurfRatioAComm	0.61	0.78	0.71
ReedRatioTComm	0.23	0.15	0.22
TurfRatioTComm	0.43	0.57	0.52

Table 4: Mean values of heartbeat-related measures for the different super taxa

7.2. Super taxa and Heartbeat

Super taxa are distinct with respect to the number of active commits (Figure 29), which, as expected, increases while moving from the COLD class to the HOT class. Especially, COLD seems to be totally distinct from the other classes, in terms of the number of active commits, which makes sense, because COLD class has zero to few active commits, by definition. Active commits is defined as the sum of reeds and turfs, which both share very similar box plots with active commits.

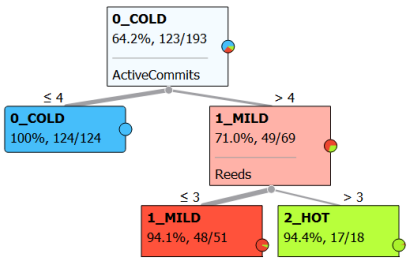

Figure 30: Decision Tree – Heartbeat for Super Taxa

Figure 30 depicts a decision tree, produced by taking as input all the attributes that are related to the Heartbeat. We can clearly conclude that *the super taxa are highly related to the heartbeat of the projects.* More specifically,

projects with less or equal than 4 active commits automatically belong to the COLD super taxon. Additionally, projects with more than 4 active commits, belong to MILD super taxon, if they have less or equal than 3 reeds, whereas if they have more than 3 reeds they belong to the HOTsuper taxon. Exceptions at the aforementioned rules are 4 misclassified projects out of 193, meaning 2%. The separation of the super taxa is very successful, and, inversely, given a super taxon, the determination of its heartbeat properties straightforward. Table 4 summarizes the behavior of the mean values of the aforementioned heartbeat-related measures for the different super taxa.

7.3. Super taxa and Activity

In this section, we discuss the relation between the super taxa and all the attributes and metrics, that are related to the activity of the projects.

Regarding the total activity, as shown in Figure 31, the super taxa are well separated from each other. *The concentration of the values at each taxon shows that (a) although outliers do exist, (b) the IQRs of the Total Activity of the super taxa do not overlap (in fact, they are quite well separated), and thus, we can conclude that (c) moving from the COLD to the HOT class the total activity increases.* However, there are a few outliers at the COLD class with high total activity. These outliers belong to the FocusedShot and Frozen taxon, which contains projects with a small number of active commits (less than 4) and total activity greater than 10.

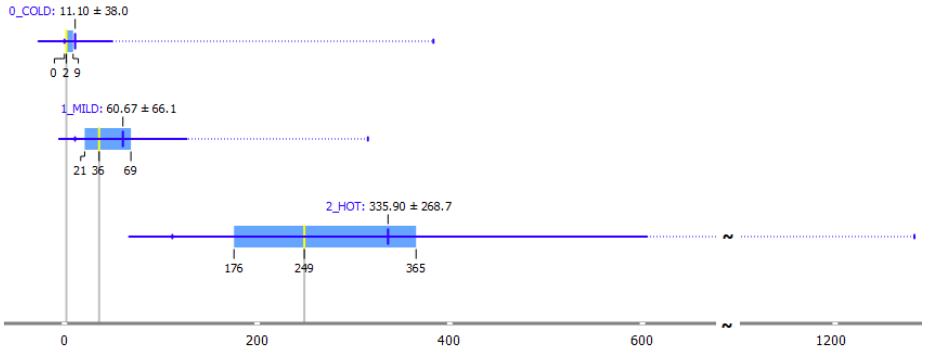

Figure 31: Box plot of Total Activity

The relationship of total maintenance and, especially, expansion to the super taxa, is similar to the relationship of the taxa to total activity. The same holds for attributes injected to and ejected from existing tables. All the respective figures are very similar to Figure 31.

Finally, to get a deeper understanding of how the activity is related to the super taxa, observe the decision tree, presented in Figure 32. At first glance, observe that this decision tree is not as simple and straightforward as the one we presented in Figure 30, which was produced from the heartbeat metrics. In this case, we are combining heartbeat-related attributes (like reeds and turfs post

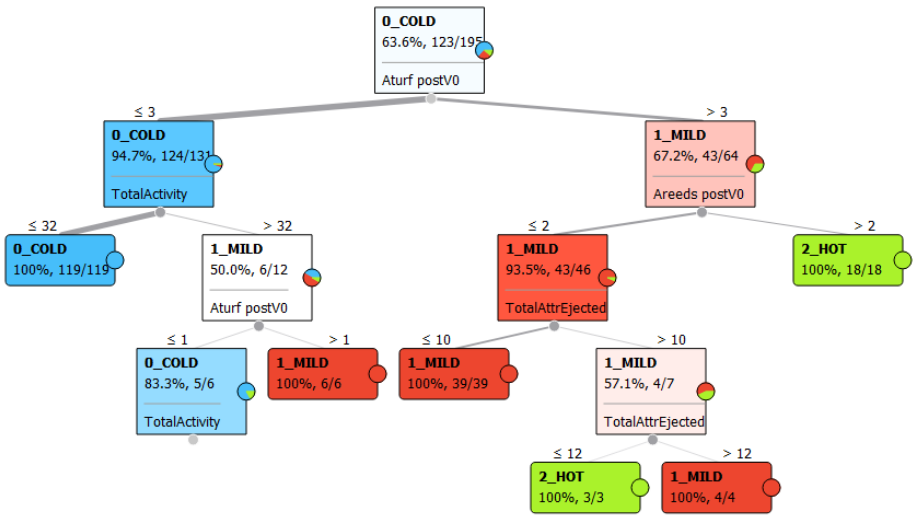

Figure 32: Decision Tree - Activity for Super Taxa

V0) with activity attributes, to separate taxa from one another. However, the benefit reaped is that *there is just 1 misclassified project out of 195.* The main idea is that COLD projects have less than 3 turf commits post the originating version, and either activity below 32 (for a very large majority of them), or just one turf. HOT, on the other hand, have more than 2 reeds and more than 3 turfs post V0, or in case they have less than reeds, they have more than 10 ejected attributes. Everything else belongs to MILD.

Figure 33 vividly depicts how cleaner is the separation of super taxa with respect to taxa. Again, we feel oblidged to note that there is a trade off: we pay the price of coarser classification and prediction for the benefit of gains in clarity, conciseness of the domain and larger degrees of separation between super taxa.

7.4. Super taxa and Table-Level Activity Measurements

In this subsection, we discuss how the super taxa and the table-level activity relate. Observe Figures 34 and 35 presenting table insertions and deletions, respectively. The concentration of the values in the super taxa shows that the super taxa are distinct. *The more active the super taxon is, the bigger the number of tables inserted and deleted is*, which verifies that the total activity of a taxon is approximated by the number of table operations fairly successfully.

8. How Focused in Time is Schema Evolution Change?

One particular question that came up in the context of our deliberations, was, "how concentrated in time is schema evolution?", as part of the broader question of "when, in the project's life, do the events of schema evolution take

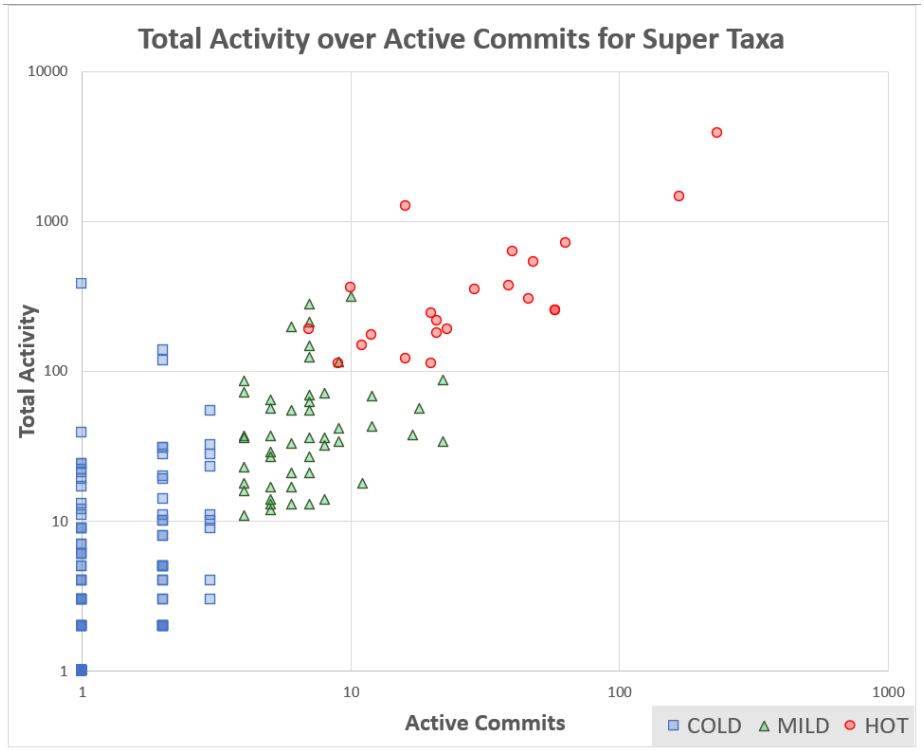

Figure 33: Active Commits and Activity for Super Taxa

place?". To this end, we have performed a timing analysis of how schema evolution unfolds in the history of the project's life.

Exactly because we are very interested in timing considerations, we have excluded all projects with a life time less or equal to 12 months, in order to minimize the probability of observing phenomena that could be attributed to a relatively Project Update Period. Out of the 195 projects of the data set, 151 qualified as the corpus of the investigation. A similar discussion is made in [30].

8.1. Nomenclature and Metrics Used

To compare the lives of these projects, we have normalized both the timespan and the activity. Thus, we measure (a) the time progress as the percentage of time passed in the project's lifespan (attention, not the Schema Update Period, but the Project Update Period), and (b) the cumulative percentage of activity (which is normalized in [0..1]) for both the schema activity and the project activity. To compute the project update period and the project duration, we have downloaded the entire history of the projects from Github and measured the number of files changed as a coarse indicator of activity. We use the month as a reasonable unit of time; thus, we have aggregated activity per month.

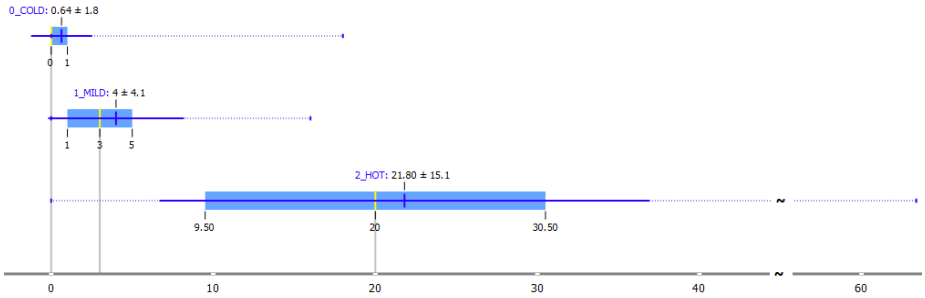

Figure 34: Box plot of Total Table Insertions

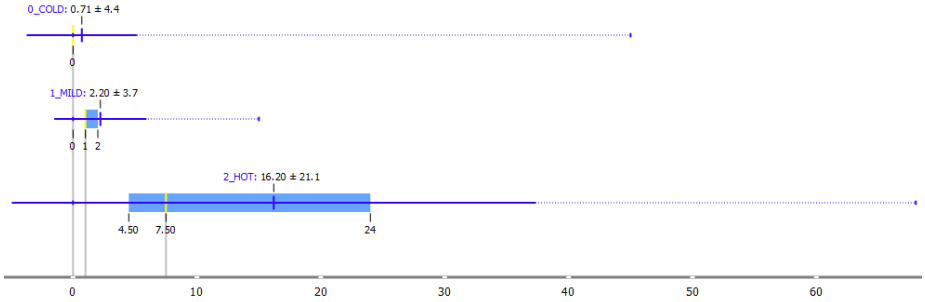

Figure 35: Box plot of Total Table Deletions

The following terms are introduced:

- The *Schema Birth Time Point* refers to the point in the context of the project lifespan, when the schema is born. *Birth volume* signifies the percentage of total activity of schema evolution at the time of birth. Birth volume is discretized in class labels as follows: *low* in [0%..25%] range, *fair* in (25%..75%] range, *high* in the (75%..100%) range, and *full* if 100%.

- *Top-Band of Schema Activity* is the top range of 90%-100% of schema activity; when a schema reaches the top-band, its evolution is close to completion. The *Time of attainment of the Top-Band of Schema Activity* is the time point of the attainment of this 90% of the total activity of schema evolution. The *Growth period* is the period between birth and top-band.

- We say that the project has a *vault* if the time span between birth and reaching the top-band is lower than 10% of project's life. The *tail period* is the interval between top-band attainment and the end of the project's history. The reason for the choice of the terms is pictorial, as (a) the vault signifies reaching a very high percentage of total activity in a very short time of the project's life, resembling a long vertical line, and (b) the tail is very often a flat line, or close to it.

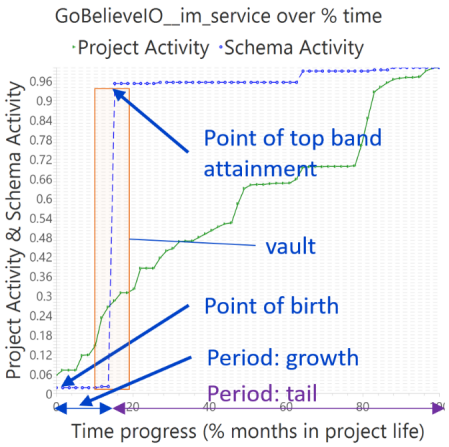

Figure 36: Example of schema and source code co-evolution. The horizontal axis demonstrates the progress in time, as a percentage of a project's life. The vertical axis depicts the cumulative progress as a percentage of the total amount of evolution activity, for (a) the schema (dotted, blue line) and (b) the source code (solid, green line).

- *Percentage of active growth* comes in two flavors, as the fraction of the months in the growth period with schema activity beyond zero over (a) the total number of months of PUP, and, (b) the number of months of the growth period only. The higher the percentage is, the more steps schema evolution took within the growth period, and the denser the evolution rate is. We discretize Active Percentage over PUP with class labels zero, if 0%, *few* in (0%..20%] range, *fair* in (20%..75%] range, and *large* in the (75%..100%].

To forestall any possible criticism, we acknowledge that both the vault thresholds are arbitrary; in fact, they were chosen to be as extreme as possible. However, we will demonstrate, more frequent than not, the projects of our corpus come with a vault. The discretization of numerical values was based on two principles, (a) isolating important extremes like zero and 100%, and, (b) trying to attain equal-size buckets in the histogram.

8.2. Findings for the concentration of schema maintenance actions in time

Frequency of Vaults. *How frequently do vaults appear? Vaults are really frequent!* The data show that 88 out of 151 histories, a surprising 58% that is, demonstrate the presence of a vault, signifying a very high probability of a super-focused nature of birth and schema stabilization. The other 63 projects do not demonstrate such an extreme transition from birth to 90% of schema evolution.

Vaults and (Super)Taxa. *Do the (super) taxa play a role in the probability of a vault? They most certainly do: the colder a super taxon is, the greater the probability it evolves via vault behavior*. In Figure 37 we depict the relationship

of taxa and super taxa to the existence of vaults, in both absolute and relative numbers.

Count of Project	hasVault		
Super Taxon	FALSE	TRUE	Grand Total
0_COLD	21	69	90
1_MILD	25	15	40
2_HOT	17	4	21
Grand Total	**63**	**88**	**151**

Count of Project	HasVault		
Taxon	FALSE	TRUE	Grand Total
0_FROZEN		27	27
1_ALMOST_FROZEN	14	34	48
1_FocusedShot_n_FROZEN	7	8	15
2_MODERATE	14	11	25
3_FocusedShot_n_LOW	11	4	15
4_ACTIVE	17	4	21
Grand Total	**63**	**88**	**151**

Count of Project	hasVault		
SuperTaxon	FALSE	TRUE	Grand Total
0_COLD	23%	77%	100%
1_MILD	63%	38%	100%
2_HOT	81%	19%	100%
Grand Total	**42%**	**58%**	**100%**

Count of Project	HasVault		
Taxon	FALSE	TRUE	Grand Total
0_FROZEN	0%	100%	100%
1_ALMOST_FROZEN	29%	71%	100%
1_FocusedShot_n_FROZEN	47%	53%	100%
2_MODERATE	56%	44%	100%
3_FocusedShot_n_LOW	73%	27%	100%
4_ACTIVE	81%	19%	100%
Grand Total	**42%**	**58%**	**100%**

Figure 37: Super Taxa (left) and Taxa (right) with respect to their inclusion of projects with vaults

The behavior is monotonic for both taxa and super taxa: the lower the volume of activity is, the higher the chances of demonstrating a vault.

Growth Period. As already mentioned, we denote as growth period the period between birth and the attainment of 90% of total evolutionary activity. Typically, one would expect a rather large amount of activity in the growth period, due to the extremely high threshold of the top-band. Thus, the question is: *how active is the growth period?* The answer (originally counterintuitive, but expected after observing the high frequency of vaults) is that *zero change is really widespread, especially, the colder the super taxon is*.

Figure 38 correlates the super taxon, the existence of vaults and the activity during the growth period, expressed as the fraction of active months in the growth period over the project's PUP.

- COLD has 85 out of 90 projects with zero activity in the growth period (expectedly, as it is mostly dominated by 89 out of 90 projects with a vault). In other words, COLD projects climb into the top-band quickly after they are born, and typically in a single step.
- MILD has 12 out of 40 projects (30%) with zero activity in the growth period, but otherwise is it mostly fair in its growth activity and without a vault.
- HOT is largely placed in the area of high growth activity without a vault (i.e., slow consistent growth with regular heartbeat). However, this involves only 14 projects, which, despite the fact that they constitute 2/3 of the corpus of the super taxon, amount to less than10% of the corpus being studied.

Overall, the existence of vaults is strongly correlated with zero activity in the growth period, and thus, an abrupt transition from birth to 90% of schema

Count(prj)	hasVault		
Super Taxon/ active %PUP	FALSE	TRUE	Grand Total
0_COLD	**21**	**69**	**90**
0_zero	16	69	**85**
1_fair	5		**5**
1_MILD	**25**	**15**	**40**
0_zero	1	11	**12**
1_fair	18	4	**22**
2_high	6		**6**
2_HOT	**17**	**4**	**21**
0_zero		1	**1**
1_fair	3	3	**6**
2_high	14		**14**
Grand Total	**63**	**88**	**151**

Count(prj)	hasVault		
Super Taxon/ active %PUP	FALSE	TRUE	Grand Total
0_COLD	**23%**	**77%**	**100%**
0_zero	19%	81%	**100%**
1_fair	100%		**100%**
1_MILD	**63%**	**38%**	**100%**
0_zero	8%	92%	**100%**
1_fair	82%	18%	**100%**
2_high	100%		**100%**
2_HOT	**81%**	**19%**	**100%**
0_zero		100%	**100%**
1_fair	50%	50%	**100%**
2_high	100%		**100%**
Grand Total	42%	58%	**100%**

Figure 38: Relationship of Super Taxon, Active Pct PUP and Vault

evolution activity (Figure 38). 81 of the 151 projects have a zero activity at growth in the presence of vaults. Moreover another 17 projects, show zero growth activity in the absence of vaults (practically signifying a pattern of the form: birth; stillness for a long period that surpasses the 10% time-limit; quick climbing to the top-band afterwards). In other words, 98 of 151 projects, nearly 2/3 of the studied corpus, show no activity between birth and top band. Birth volume is closely related too, as the COLD taxon has 72 of the 83 projects with high (more than 75%) or full volume of activity at birth (see Figure 39). These 83 out of 151 projects, i.e., 55% of the corpus showing no activity between birth and top-band, is a really high percentage, and adds to our case towards absence rather than presence of evolution in the lives of schemata.

Birth Volume, Vaults and Super Taxa. *Is there a correlation between super taxa, vaults and birth volume?* One would expect that the higher the percentage of change the schema birth introduces (equivalently: the less change happens after birth) the bigger the possibility of vaults and cold behavior. Indeed, *the results support the claim that the higher the amount of birth volume*

Count (prj)	Birth Volume				
SuperTaxon / active%PUP	1_low	2_fair	3_high	4_full	Grand Total
0_COLD	1	17	33	39	90
0_zero	1	12	33	39	85
1_fair		5			5
1_MILD	7	24	9		40
0_zero	1	5	6		12
1_fair	5	14	3		22
2_high	1	5			6
2_HOT	7	12	2		21
0_zero			1		1
1_fair		5	1		6
2_high	7	7			14
Grand Total	15	53	44	39	151

Figure 39: Relationship of Super Taxon, Active Pct PUP and Birth Volume

(i.e., the less activity after birth, the schema has) the more likely to have vaults, and the more likely to be in a colder super taxon.

Count(Prj)	Vault		
Super Taxon/ Birth Vol	FALSE	TRUE	Grand Total
0_COLD	21	69	90
1_low	1		1
2_fair	11	6	17
3_high	9	24	33
4_full		39	39
1_MILD	25	15	40
1_low	6	1	7
2_fair	16	8	24
3_high	3	6	9
2_HOT	17	4	21
1_low	7		7
2_fair	10	2	12
3_high		2	2
Grand Total	63	88	151

Figure 40: Relationship of Super Taxon, Birth Volume and Vault

Figure 40 demonstrates how super taxa are involved: the COLD super taxon contains 39 projects (out of 90 COLD and totally 151 projects) with full volume in the month of their birth. Another 33 are born with a high volume at birth (i.e., higher than 75% of their total activity). The percentage drops to 9 out of 40 for the MILD super taxon and 2 out of the 21 for the HOT one. In all cases, the existence of a vault, naturally correlates with higher birth volumes.

Overall, the lesson learned from the deliberations of this Section is that the colder the super taxon is, the higher the probability for its projects to demonstrate high percentage of their change at birth (i.e., absence later) as well as concentrated-in-time evolution (in the form of a vault and zero growth period).

9. Concluding Remarks

In this study, we attack the problem of understanding the characteristics of schema evolution by performing the largest empirical study ever performed in the domain of Free Open Source Software projects. Our first contributions involve:

- A clear definition of the nomenclature and the important measures of the problem.
- The collection of a significantly large dataset, with a population of 195 studied projects (almost 20 times larger than the largest study in the literature).
- The study of the heartbeat of schema evolution as well as the identification of taxa of schema evolution (both for the first time ever), with taxa being distinct classes of archetypal behavior of a schema over its lifetime.
- The answering of several research questions around the nature of schema evolution, for the first time in the related literature (see next).

Coming back to our original research questions, we can summarize our findings as follows.

9.1. Schema Evolution and its Taxa

RQ1. Is schema evolution present extensively? For a very large percentage of projects, schema evolution is practically absent. As already mentioned, out of the largest possible collection of 327 projects that we came up, 40% had no evolution whatsoever, 10% had different versions but no schema changes at the logical level and 20% were almost frozen.

We believe that our empirical evidence is important exactly because it refutes the traditional belief that schema evolution is extensive (also reported in the literature, e.g. [19]) and replaces it with a new perspective: *schema evolution is mostly absent from the typical Free Open Source Project, and emphatically present only in a small percentage of projects with an active profile of continuous schema maintenance*. The massive and widespread nature of this absence drives us to conjecture that there is a strong possibility that the idiosyncrasy of schema evolution is that this "absence" is not due to the lack of its necessity, but rather due to its difficulty (see also [2])!

RQ2. Are there archetypal patterns of schema lives? For the first time in the related literature, we *define and study the heartbeat* and *present patterns of schema evolution*. Frozen projects with no change whatsoever and Almost Frozen with few active commits and small change constitute 17% and 33% of the studied population – i.e., half the projects of the study. Focused Shot and Frozen projects with almost no activity other than a single spike of change arise to 13% and Focused Shot and Low projects, with a couple of high-volume reeds of evolution and less than 10 active commits overall another 10%. Projects of constant rate of schema maintenance involve (a) Moderate projects, with

less than 90 attributes changed in their lifetime, and a fraction of 15% of the population, and, (b) Active ones, with frequent change and high volumes of it, amounting to an 11% of the population.

9.2. Taxa Properties

RQ3. How do the taxa of schema evolution relate to the demonstrable evolutionary properties, such as volume, frequency and important characteristics? The taxa of schema evolution have been investigated in depth with respect to four families of characteristics: heartbeat, evolutionary activity, schema evolution duration and project duration. In the sequel, we summarize our findings.

Taxa and Duration of schema and project change. Concerning the project update period (PUP), the mean values and the boxes of the different taxa with respect to their PUP are very similar! At the same time, *all* taxa reach fairly high durations for the schema update period and have broad IQRs that range between 1 and 3 years, meaning that similar amount of change is produced with different time distribution within the same taxon. The Schema update period slowly increases with the taxon's "heat" of activity, but, although the differences between taxa are observable, they are not extreme. *Result: change differences are not due to the duration, but due to the inherent characteristics of projects.*

Taxa and heartbeat. Active Commits, Turfs and Reeds alone can be used as the main determinants of the taxon of a project. The taxa are clearly different concerning their active commits, and as the taxon becomes more active, all the related measures increase: the mean, the median, the IQR and the range of the active commits. Active commits are the best separator of the taxa, although both reeds and turf demonstrate similar behavior.

Taxa and activity characteristics. There is a clear progressive shift towards higher amount of change activity as the taxon becomes more active. The pattern concerns both Total Activity and its breakdown (Maintenance and Expansion). The volume of table births and deaths, which is a coarse approximation of Total Activity, follows the above pattern, too.

9.3. Super Taxa of Schema Evolution

At the same time, we have observed an opportunity of providing more clarity in these descriptions via the introduction of super taxa, which are an extension of the taxonomy via the groupings of the aforementioned taxa, providing a cleaner separation of measures.

RQ4. Can we do better in terms of a taxonomy of evolutionary behaviors than the aforementioned taxa?

The introduction of super-taxa was based on the observation that while the overall separation of the projects in taxa was successful overall, we can still provide a more concise taxonomy –and in fact, expressed as a hierarchy of super taxa on top of them– with increased separation and better intuitiveness. Here, we summarize our findings.

Schema Heartbeat. Reeds, Turfs, and Active Commits distinguish the super taxa well. This conclusion is based on two pillars, specifically, (a) the observation

of the respective box plots, where the values for each taxon are distinct, as well as, (b) from the decision tree for the super taxa, n the basis of heartbeat-related information (Active Commits and Reeds).

Activity. We have found that the "hotter" the super taxon is, the bigger the general activity of its projects is. The box plots of all activity-related measures (here: only Total Activity was shown) indicate that *all activity-related measures are good taxon discriminators and confirm the assumption that the activity of the projects increases as we move from the cold to the hot super taxon*. Additionally, in the decision tree produced by combining heartbeat and activity-related measures, the areas of each super taxon are distinct, with an extremely successful separation.

Table-Level Activity Measurements. The measures $totalTableInsertions$ and $totalTableDeletions$ discriminate the taxa well. Again, *the "hotter" the super taxon, the higher the amount of table-level activity we can expect to observe.*

A final note, regarding *duration*, which was not explicitly mentioned in the super taxa discussion: *all the observations around durations made for the taxa continue to hold!* In this case, the definition of the super-taxa as clean groupings of the detailed taxa also leaves the behavior of both Schema and Project Update Periods intact.

9.4. Timing of Evolution

Concerning the time behavior of schema evolution, we observe that *concentrated maintenance effort (in the form of a vault) as well as a high percentage of change at birth (and therefore, its absence, afterwards) are very frequent*. Moreover, such a behavior is *highly correlated with the super taxon of a project*: the colder a super taxon is, the more likely it is to observe the aforementioned phenomena.

9.5. Why does this matter?

Studying schema evolution is important because, apart from enriching humanity's knowledge and moving from word-of-mouth impression to concrete evidence, it provides insights to different audiences.

Is what we have observed "healthy"? Are there norms that dictate the right way to evolve the schema of a system, given the evolution of requirements, environment and surrounding source code? We believe our work here highlights an important gap in our body of knowledge. For the moment, as a community, we have theoretically explored, implemented in our DBMSs, and subsequently, taught our students, variations of interdependencies between the fundamental components of relational databases (tables and attributes that is) in the form of fundamental constraints of uniqueness and consistency. At a more theoretical level, we have also explored variants of functional dependencies. However, we have no norms to evaluate whether a schema has evolved "correctly", or methodologies on how to do it. As educators, we have the responsibility to teach students on what they will face in the trenches and how to deal with it. Principled methods for evolving schemata, hopefully with tool assistance would

greatly facilitate this task. The proposed taxa in this paper *provide researchers and teachers with a "reference" set of archetypes that can serve both as a starting point for this research, as well as a concise summary of what to expect from the evolution of a schema – at least in the FOSS world*.

Moving from the metamodel to the schema level, studying how the different taxa relate to measurable properties of schema evolution has implications on a two-way relationship: being able to define and separate taxa on the grounds of such properties, and, on the other hand, being able to predict evolution-related properties, given the taxon of a project. Thus, another community interested in our results are *software curators and developers assessing software*. The presented profiles can be used as *predictors on the potential tendency of the developers of a FOSS project with respect to the evolution of the database*. This can affect (a) design decisions from the part of project curators, and, (b) selection decisions from the part of people (re)using existing projects.

9.6. Open paths for research

As already mentioned in [2], the absence of industrial schema histories (and thus our reliance on FOSS projects only) makes the problem of acquiring publicly available schema histories from the industry quite an improbable scenario. This is especially important since we have to make a specific note for the fact that our deliberations have involved the area of Free Open Source Software and cannot be generalized to the entire domain of schema evolution for all database-backed information systems. The investigation of proprietary cases is still a gazing gap in the body of knowledge of the scientific community. Of course, closing this gap would require the systematic monitoring and reporting of the history of schema evolution of such projects – a task that is all but obvious on how one would get access to the necessary resources, in order to bring it to a successful end.

In the absence of such data, however, we can continue research to test the existence of patterns at the table level, to extract the treatment of constraints (esp., foreign keys) in FOSS projects and to qualitatively study gravitation to rigidity at more depth.

A final thought concerns the meta- level again: do we observe the absence of evolution of the schema because there is no need to perform it, or due to other reasons? For example, is the difficulty of evolving schemata when applications are built on top of them, inherent to the relational model? Software engineering, when it comes to the issue, would attribute the absence of evolution to rigidity [31, 32, 33]. However, we do not have any explicit evidence (e.g., via questionnaires or interviews) that schema and application evolution was intentionally avoided at large. Being able to demonstrate, beyond reasonable doubt, whether absence of evolution is due to absence of a need to evolve, or due to any form of rigidity is another topic open to research.

[1] P. Vassiliadis, Profiles of schema evolution in free open source software projects, in: 37th IEEE International Conference on Data Engineering, ICDE 2021, Chania, Greece, April 19-22, 2021, IEEE, 2021, pp. 1–12.

[2] M. Stonebraker, R. C. Fernandez, D. Deng, M. L. Brodie, Database decay and what to do about it, Commun. ACM 60 (1) (2017) 11. doi:10.1145/3014349.
URL https://doi.org/10.1145/3014349

[3] T. A. Limoncelli, SQL is no excuse to avoid devops, Commun. ACM 62 (1) (2019) 46–49. doi:10.1145/3287299.
URL https://doi.org/10.1145/3287299

[4] J. Delplanque, A. Etien, N. Anquetil, O. Auverlot, Relational database schema evolution: An industrial case study, in: 2018 IEEE International Conference on Software Maintenance and Evolution, ICSME 2018, Madrid, Spain, September 23-29, 2018, IEEE Computer Society, 2018, pp. 635–644.

[5] A. Maule, W. Emmerich, D. S. Rosenblum, Impact analysis of database schema changes, in: W. Schäfer, M. B. Dwyer, V. Gruhn (Eds.), 30th International Conference on Software Engineering (ICSE 2008), Leipzig, Germany, May 10-18, 2008, ACM, 2008, pp. 451–460.

[6] S. K. Gardikiotis, N. Malevris, A two-folded impact analysis of schema changes on database applications, Int. J. Autom. Comput. 6 (2) (2009) 109–123.

[7] G. Papastefanatos, P. Vassiliadis, A. Simitsis, Y. Vassiliou, Hecataeus: Regulating schema evolution, in: ICDE, 2010, pp. 1181–1184.

[8] M. Hartung, J. F. Terwilliger, E. Rahm, Recent advances in schema and ontology evolution, in: Z. Bellahsene, A. Bonifati, E. Rahm (Eds.), Schema Matching and Mapping, Data-Centric Systems and Applications, Springer, 2011, pp. 149–190.

[9] P. Manousis, P. Vassiliadis, A. V. Zarras, G. Papastefanatos, Schema evolution for databases and data warehouses, in: 5th European Summer School on Business Intelligence , eBISS 2015, Vol. 253 of Lecture Notes in Business Information Processing, Springer, 2015, pp. 1–31.

[10] L. Caruccio, G. Polese, G. Tortora, Synchronization of queries and views upon schema evolutions: A survey, ACM Trans. Database Syst. 41 (2) (2016) 9:1–9:41.

[11] C. Curino, H. J. Moon, A. Deutsch, C. Zaniolo, Automating the database schema evolution process, VLDB J. 22 (1) (2013) 73–98.

[12] K. Herrmann, H. Voigt, J. Rausch, A. Behrend, W. Lehner, Robust and simple database evolution, Inf. Syst. Frontiers 20 (1) (2018) 45–61.

[13] R. E. Schuler, C. Kesselman, A high-level user-oriented framework for database evolution, in: 31st International Conference on Scientific and Statistical Database Management, SSDBM 2019, Santa Cruz, CA, USA, July 23-25, 2019, ACM, 2019, pp. 157–168.

[14] D. Sjøberg, Quantifying schema evolution, Information and Software Technology 35 (1) (1993) 35–44.

[15] C. Curino, H. J. Moon, L. Tanca, C. Zaniolo, Schema evolution in wikipedia: toward a web information system benchmark, in: Proceedings of ICEIS 2008, 2008.

[16] A. Cleve, M. Gobert, L. Meurice, J. Maes, J. H. Weber, Understanding database schema evolution: A case study, Sci. Comput. Program. 97 (2015) 113–121.

[17] D.-Y. Lin, I. Neamtiu, Collateral evolution of applications and databases, in: Joint Intl. Annual ERCIM Workshops on Principles of Software Evolution (IWPSE) and Software Evolution (Evol), 2009, pp. 31–40.

[18] S. Wu, I. Neamtiu, Schema evolution analysis for embedded databases, in: 2011 IEEE 27th International Conference on Data Engineering Workshops, ICDEW '11, 2011, pp. 151–156.

[19] D. Qiu, B. Li, Z. Su, An empirical analysis of the co-evolution of schema and code in database applications, in: 2013 9th Joint Meeting on Foundations of Software Engineering, (ESEC/FSE), 2013, pp. 125–135.

[20] P. Vassiliadis, M. Kolozoff, M. Zerva, A. V. Zarras, Schema evolution and foreign keys: a study on usage, heartbeat of change and relationship of foreign keys to table activity, Computing 101 (10) (2019) 1431–1456.

[21] I. Skoulis, P. Vassiliadis, A. V. Zarras, Growing up with stability: How open-source relational databases evolve, Information Systems 53 (2015) 363–385.

[22] P. Vassiliadis, A. V. Zarras, I. Skoulis, Gravitating to rigidity: Patterns of schema evolution - and its absence - in the lives of tables, Information Systems 63 (2017) 24–46.

[23] P. Vassiliadis, A. V. Zarras, Schema evolution survival guide for tables: Avoid rigid childhood and you're en route to a quiet life, Journal of Data Semantics 6 (4) (2017) 221–241.

[24] K. Dimolikas, A. V. Zarras, P. Vassiliadis, A study on the effect of a table's involvement in foreign keys to its schema evolution, in: G. Dobbie, U. Frank, G. Kappel, S. W. Liddle, H. C. Mayr (Eds.), Conceptual Modeling - 39th International Conference, ER 2020, Vienna, Austria, November 3-6, 2020, Proceedings, Vol. 12400 of Lecture Notes in Computer Science, Springer, 2020, pp. 456–470.

[25] M. Klettke, H. Awolin, U. Störl, D. Müller, S. Scherzinger, Uncovering the evolution history of data lakes, in: IEEE International Conference on Big Data, BigData 2017, Boston,A, USA, December 11-14, 2017, IEEE Computer Society, 2017, pp. 2462–2471.

[26] S. Scherzinger, S. Sidortschuck, An empirical study on the design and evolution of nosql database schemas, in: G. Dobbie, U. Frank, G. Kappel, S. W. Liddle, H. C. Mayr (Eds.), Conceptual Modeling - 39th International Conference, ER 2020, Vienna, Austria, November 3-6, 2020, Proceedings, Vol. 12400 of Lecture Notes in Computer Science, Springer, 2020, pp. 441–455.

[27] U. Störl, M. Klettke, S. Scherzinger, Nosql schema evolution and data migration: State-of-the-art and opportunities, in: A. Bonifati, Y. Zhou, M. A. V. Salles, A. Böhm, D. Olteanu, G. H. L. Fletcher, A. Khan, B. Yang (Eds.), Proceedings of the 23rd International Conference on Extending Database Technology, EDBT 2020, Copenhagen, Denmark, March 30 - April 02, 2020, OpenProceedings.org, 2020, pp. 655–658.

[28] G. Gousios, The ghtorent dataset and tool suite, in: Proceedings of the 10th Working Conference on Mining Software Repositories, MSR '13, San Francisco, CA, USA, May 18-19, 2013, IEEE Computer Society, 2013, pp. 233–236.

[29] C. Bird, P. C. Rigby, E. T. Barr, D. J. Hamilton, D. M. Germán, P. T. Devanbu, The promises and perils of mining git, in: 6th International Working Conference on Mining Software Repositories, (MSR), IEEE Computer Society, 2009, pp. 1–10.

[30] P. Vassiliadis, F. Shehaj, G. Kalampokis, A. Zarras, Joint Source and Schema Evolution: Insights from a Study of 195 FOSS Projects, in: Proceedings of the 26th International Conference on Extending Database Technology EDBT 2023, Ioannina, Greece, March 28-31, 2023, OpenProceedings.org, 2023.

[31] M. M. Lehman, Laws of Software Evolution Revisited, in: Proceedings of 5th European Workshop on Software Process Technology, (EWSPT), 1996, pp. 108–124.

[32] R. Martin, Agile Software Development, Principles, Patterns, and Practices, Pearson, 2002.

[33] M. M. Lehman, J. C. Fernandez-Ramil, Software Evolution and Feedback: Theory and Practice, Wiley, 2006, Ch. Rules and Tools for Software Evolution Planning and Management.