Everything you always wanted to know about your process, but did not know how to ask

Gonzalez Lopez de Murillas, E.; Reijers, H.A.; van der Aalst, W.M.P.

Published in:
Business Process Management Workshops

DOI:
10.1007/978-3-319-58457-7_22

Published: 01/01/2017

Document Version
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Everything You Always Wanted to Know About Your Process, but Did Not Know How to Ask

Eduardo González López de Murillas[1(✉)], Hajo A. Reijers[1,2], and Wil M.P. van der Aalst[1]

[1] Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, The Netherlands
{e.gonzalez,h.a.reijers,w.m.p.v.d.aalst}@tue.nl

[2] Department of Computer Science, VU University Amsterdam, Amsterdam, The Netherlands

Abstract. The size of execution data available for process mining analysis grows several orders of magnitude every couple of years. Extracting and selecting the relevant data to be analyzed on each case represents an open challenge in the field. This paper presents a systematic literature review on different approaches to query process data and establish their provenance. In addition, a new query language is proposed, which overcomes the limitations identified during the review. The proposal is based on a combination of data and process perspectives. It provides simple constructs to intuitively formulate questions. An implementation of the language is provided, together with examples of queries to be applied on different aspects of the process analysis.

Keywords: Process mining · Databases · Event logs · Query languages

1 Introduction

One of the main goals of process mining techniques is to obtain insights into the behavior of systems, companies, business processes, or any kind of workflow under study. Obviously, it is important to perform the analysis on the right data. Being able to extract and query some specific subset of the data becomes crucial when dealing with complex and heterogeneous datasets. In addition, the use of querying tools allows finding specific cases or exceptional behavior. Whatever the goal, analysts often find themselves in the situation in which they need to develop ad-hoc software to deal with specific datasets, or use existing tools that might be difficult to use, too general, or just not suitable for process analysis.

Different approaches exist to support the querying of process data. Some of them belong to the field of Business Process Management (BPM). In this field, events are the main source of information. They represent transactions or activities that were executed at a certain moment in time in the environment under study. Querying this kind of data allows to obtain valuable information about the behavior and execution of processes. There are other approaches originating

M. Dumas and M. Fantinato (Eds.): BPM 2016 Workshops, LNBIP 281, pp. 296–309, 2017.
DOI: 10.1007/978-3-319-58457-7_22

from the field of data provenance, which are mainly concerned with recording and observing the origins of data. This field is closely related to scientific workflows, in which the traceability of the origin of experimental results becomes crucial to guarantee correctness and reproducibility. In each of these fields, we find query languages and techniques that focus on the particularities of their input data. However, none of these approaches succeeds at combining process and data aspects in an integrated way. In addition, the development of a query mechanism that allows to exploit this combination, while being intuitive and easy to use, represents an additional challenge to overcome.

In order to make the querying of process event data easier and more efficient, we propose a new query language that exploits both process and data perspectives. Section 2 presents a systematic literature review, analyzing existing solutions and comparing them. Section 3 presents our approach together with examples of its use. Section 4 provides information about the implementation and, finally, Sect. 5 concludes the paper.

2 Systematic Literature Review

In order to get an overview of existing approaches, we first concluded a systematic literature review [1]. Figure 1 shows an overview of the procedure. First, a coarse set of candidate papers needs to be obtained from a scientific publications database or through a search engine (*Query*). Afterward, a *relevance screening* is performed in order to identify with papers are actually within the scope. To do so, a set of criteria are defined. Only papers that fulfill these criteria pass to the next phase. Next, a *quality screening* is conducted on the relevant papers. This is done by defining some minimum quality criteria that the papers must satisfy. Finally, with the selected papers that are relevant and have sufficient quality, a detailed review is performed.

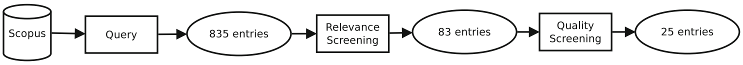

Fig. 1. Pipeline of the systematic review process

In accordance with the procedure described in Fig. 1, as a first step, we performed a search of related papers. To do so, we chose Scopus[1], one of the largest abstract and citation database of peer-reviewed literature, including scientific journals, books, and conference proceedings. This database provides a search engine that, by means of queries, allows to specify different kinds of criteria to filter the results.

In our case, we are interested in papers that refer to *business processes* or *workflows*, that relate to *queries* and that analyze *events*, *logs*, *provenance*, *data* or *transactions*. In addition, we want to filter out any work that does not belong to the area of *Computer Science*, or that is not written in English. The exact query as executed in the search engine can be observed in Listing 1.

[1] http://www.scopus.com.

Listing 1. Query as executed in Scopus

```
TITLE−ABS−KEY("business process"OR "workflow") AND
TITLE−ABS−KEY("query"OR "querying") AND
TITLE−ABS−KEY("event"OR "log" OR "provenance"OR
             "data" OR "transaction") AND
( LIMIT−TO(SUBJAREA,"COMP" ) ) AND (
LIMIT−TO(LANGUAGE,"English") )
```

The query above yielded 835 results, from the years 1994 to 2016, with the distribution depicted in Fig. 2.

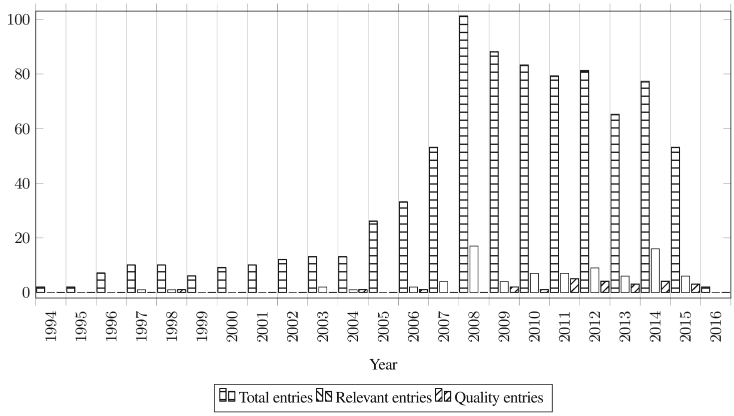

Fig. 2. Distribution per year of the related work as a result of the first query on Scopus.

However, not all of these proved relevant to our topic. When performing the *relevance screening*, we make a distinction between inclusive (I) and exclusive (E) criteria. In this way, candidates that satisfy *all* the inclusive criteria and do not satisfy any of the exclusive will be included in the next phase of the review. The ones that do not satisfy all the inclusive criteria or satisfy at least one of the exclusive ones will be discarded. The specific criteria used in this review are listed below:

1. Does the study consider process data as input? (I)
2. Does the study use only process models as input? (E)
3. Does the study propose a language to query the data? (I)

As a result of the *relevance screening* phase, the whole set of 835 entries was reduced to a set of 83 relevant works. These 83 entries are considered to be related and in the scope of process and provenance data querying. However, to guarantee a minimum level of detail, we defined the following criteria for the *quality screening* phase:

1. Does the study provide a sufficient description of the language?
2. If the language already exists, are the extensions, modifications, adaptations sufficiently explained?
3. Does the study include concrete examples of application of the language?

As a result of this phase, the set of 83 relevant works was reduced to a set of 25 papers with enough quality to be analyzed in detail. At the final stage, these papers have been analyzed to identify their most important features, and then compared to our approach. To do so, the content of each paper was reviewed, closely considering the main characteristics of the approach they describe. These characteristics refer to the kind of input data that is used by each approach (*Input data aspects*), qualities related to provenance (*Provenance aspects*), business processes (*Business process aspects*), database or artifact environments (*Database or Artifact aspects*), and the nature of the questions that can be queried with them (*Query aspects*). Table 1 presents the main characteristics of the remaining 25 references and how they can be classified when looking at the features listed below:

Input data aspects

- **Event data:** The approach allows to query and analyze event data.
- **Model-based:** The approach takes into account execution models such as workflows, BPEL, or Petri nets.
- **Storage model:** A meta-model for provenance or event data storage is proposed by the approach.
- **Complex event processing:** Different sources of events are analyzed by the approach to infer new kinds of more complex events.
- **Streams:** It is possible to query streams of events instead of complete logs.

Provenance aspects

- **Provenance-oriented:** The approach is provenance oriented or allows to record and query provenance information on the execution of workflows, scientific workflows, business processes, etc.
- **OPM-compliant:** The storage model used by the approach complies with the Open Provenance Model [2].
- **Data lineage:** The language allows to query about the life cycle of data, its origins and where it moves over time.
- **Dependency graphs/relations:** Relations between data entities are considered by the approach. For example dependency graphs, common in the provenance area, are used.

Business Process aspects

- **Business process oriented:** The approach is applied to the field of business processes management. In addition, it considers business process specific aspects while querying, e.g., using concepts such as activities, cases, resources, etc.

Database or Artifact aspects

- **Entities/artifacts:** The approach captures information about the objects, data entities or artifacts related to the event or provenance data. This information can be queried as well.
- **Database-oriented:** The approach captures database specific data such as schema, tables, column information, keys, objects, etc., in case the event information is extracted from that kind of environment.
- **State-aware:** The state of the analyzed system is considered by the approach at the time of capturing and querying data. This can be related to process state, data state, versioning, etc.

Query aspects

- **Graph-based:** Queries and results are expressed as graphs, in which edges are relations and nodes can be data objects, events, activities, etc.
- **Relevance querying:** It is possible to query about data relevance, i.e. relations between data that do not only reflect data origin.
- **Semantic querying:** The query language is based on or compatible with semantic technologies such as RDF, OWL, SPARQL, etc.
- **Regular path queries (RPQ):** The language allows to make queries that select nodes connected by a path on a graph based database.
- **Projection queries:** It is possible to query cases that fit a partial pattern using projection.
- **Temporal properties/querying:** The language or technique makes it possible to query temporal properties related to the data. It can also refer to temporal logic languages such as LTL, CTL, etc.
- **Event correlation:** The approach does not consider events in isolation, but allows to query the correlation between them, e.g. querying evens related to the same artifact.
- **Multi process:** The approach allows to query aspects related to several processes at the same time on the same dataset.
- **Multi log:** Several event logs can be queried at the same time in order to combine results for a single query.
- **Multi data schema:** Several data schemas can be considered in a single query.

Looking at Table 1, it can be seen that most of the approaches can be categorized in one of two big groups: *provenance-oriented* and *business process oriented*. The *provenance-oriented* approaches [3–12] usually support some kind of provenance model, data lineage or so. However, not all the approaches under this category support every aspect of data provenance. Only one of them [7] is database-oriented and considers states and artifacts. Most of the *business process oriented* approaches [13–26] seem to ignore data provenance aspects, and focus mainly on capturing causal relations of business activities and supporting different ways to query the data. There is an outlier [27] that focuses only on the temporal analysis of event logs using temporal logic checking. However, this solution ignores all other aspects of the data. As can be seen, none of the existing

Table 1. Comparison of features for the references at the end of the systematic review.

		Input data					Provenance				BP	DB			Query aspects									
Ref	Title	Event data	Model-based	Storage model	Complex event processing	Streams	Provenance-oriented	OPM-compliant	Data lineage	Depencency graphs/Relations	Business process oriented	Entities/Artifacts	Database-oriented	State-aware	Graph-based	Relevance querying	Semantic querying	Regular path queries (RPQ)	Projection queries	Temporal properties/querying	Event correlation	Multi process	Multi log	Multi data schema
[3]	Answering regular path queries on workflow provenance	✓					✓								✓			✓						
[4]	Capturing and querying workflow runtime provenance with PROV: A practical approach	✓		✓			✓			✓														
[5]	Modeling and querying scientific workflow provenance in the D-OPM	✓		✓			✓	✓							✓			✓						
[6]	Towards a scalable semantic provenance management system	✓					✓	✓	✓	✓							✓							
[7]	Towards integrating workflow and database provenance	✓		✓			✓	✓	✓	✓		✓	✓	✓										
[8]	MTCProv: A practical provenance query framework for many-task scientific computing	✓		✓			✓	✓	✓	✓														
[9]	OPQL: A first OPM-level query language for scientific workflow provenance	✓					✓	✓	✓						✓									
[10]	Storing, reasoning, and querying OPM-compliant scientific workflow provenance using relational databases	✓		✓			✓	✓	✓	✓														
[11]	XQuery meets Datalog: Data relevance query for workflow trustworthiness	✓					✓			✓						✓								
[12]	A model for user-oriented data provenance in pipelined scientific workflows	✓		✓			✓		✓	✓														
[13]	A knowledge driven approach towards the validation of externally acquired traceability datasets in supply chain business processes	✓							✓		✓						✓							
[14]	Process query language: A way to make workflow processes more flexible	✓		✓					✓		✓													
[15]	Workflow History Management	✓		✓							✓													
[16]	The HIT model: Workflow-aware event stream monitoring	✓	✓		✓	✓					✓													
[17]	Semantic Enabled complex Event Language for business process monitoring	✓			✓				✓		✓				✓		✓		✓	✓				
[18]	A framework supporting the analysis of process logs stored in either relational or NoSQL DBMSS	✓		✓							✓				✓									
[19]	Business impact analysis: a framework for a comprehensive analysis and optimization of business processes	✓		✓						✓	✓	✓	✓											
[20]	Model-driven event query generation for business process monitoring	✓	✓		✓						✓													
[21]	Querying process models based on the temporal relations between tasks		✓								✓									✓				
[22]	A query language for analyzing business processes execution	✓							✓	✓	✓	✓			✓		✓				✓			
[23]	Top-k projection queries for probabilistic business processes	✓	✓	✓							✓								✓					
[24]	Integration of Event Data from Heterogeneous Systems to Support Business Process Analysis	✓	✓	✓							✓										✓			
[25]	Enabling semantic complex event processing in the domain of logistics	✓			✓						✓						✓							
[26]	Optimizing complex sequence pattern extraction using caching	✓			✓						✓								✓					
[27]	Log-based understanding of business processes through temporal logic query checking	✓																		✓				
	Our approach	✓	✓	✓			✓		✓	✓	✓	✓	✓	✓		✓				✓	✓	✓	✓	✓

approaches succeeds at combining data provenance and business processes with a good support for querying aspects.

The insight from this literature review is that in the field of process data querying, there is a need for a solution that combines business process analysis with the data perspective, which also allows to query all this information in an integrated way. Taking into account that, in most cases, the execution of business processes is supported by databases, the consideration of the data perspective becomes specially relevant.

3 Data-Aware Process Oriented Query Language

To illustrate our approach for process data querying, we propose the following running example. Consider that we want to study a process related to ticket sales and the organization of concerts. To support this, a database is used, which stores all the information relating to customers, bookings, tickets, concert halls, seats, bands, and concerts. The simplified schema of such a database is depicted in Fig. 3.

The analysis of such an environment presents many challenges. The most prominent one is the lack of explicit process execution information. The data

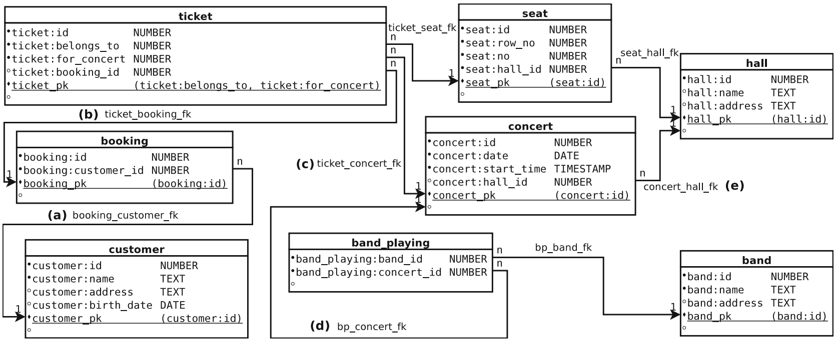

Fig. 3. Data schema of the example database

schema in Fig. 3 does not allow to record events or execution trails like cases or process instances. However, this issue has been dealt with in [28], where we explore several methods to obtain events from database settings and a meta model is proposed to store all the information about the process and data perspectives. From now on, whenever we refer to this example, we will assume the existence of a data set that complies with the meta model proposed in [28]. This meta model, that can be observed in Fig. 4, allows to combine the traditional process view (events, instances and processes), with the data perspective (data models, objects and object versions).

In order to provide a solution that enables the querying of event data considering process and data perspectives, we propose a new query language. The *Data-Aware Process Oriented Query Language* (DAPOQ-Lang), built as an interface on top of the meta model proposed in [28], allows to query not only process execution data such as events and cases, but related objects, its versions and data schemas as well, together with temporal properties of all these elements. The combination of all these aspects makes it possible to get better insights and improve the user experience. The process-oriented nature of this language improves query development time and readability, which business analysts will, hopefully, appreciate. A detailed definition of the syntax and additional documentation on the language can be found on the website of the project[2]. In the subsections below we provide some examples of use of the language related to the different characteristics.

3.1 Input Data Aspects

Event data: Our query language is based on the meta model proposed in [28]. An ER diagram of this meta model is depicted in Fig. 4. As can be observed, logs, cases, and events are part of it. Listing 2 provides an example of a query to obtain a certain subset of events.

[2] https://www.win.tue.nl/~egonzale/projects/dapoq-lang/.

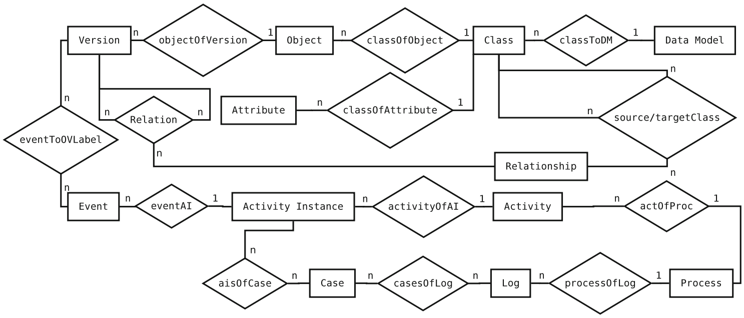

Fig. 4. ER diagram of the meta model

Listing 2. DAPOQ-Lang query to get all the events for which attribute "address" contains the substring "Madrid".

```
return allEvents where at.address CONTAINS "Madrid"
```

Model-based: Being model based, it is possible to make queries that combine event data with model information such as activities. For example, Listing 3 shows a query to obtain events that belong to a specific activity of a certain process.

Listing 3. DAPOQ-Lang query to get all the events corresponding to activities which label contains the substring "update"of the process named "concert-organizing".

```
return eventsOf(activitiesOf(allProcesses where name == "concert-organizing")
                where name CONTAINS "update")
```

Storage model: As has been mentioned before, the language builds on top of the meta model described in [28], and the ER diagram in Fig. 4. This meta model represents a valid storage model for event data combining the process and data perspectives on a single structure.

3.2 Provenance Aspects

Provenance-oriented: Data provenance aims at tracing the origins of data and enabling reproducibility of experiments. An example of tracing the origins of such data is presented in Listing 4, in which we query all the events that affected a specific object in our dataset.

Listing 4. DAPOQ-Lang query to get all the events that affected a specific object.

```
return eventsOf(allObjects where id == "43")
```

Data lineage: Some of the data lineage aspects, such as the origins of data, have been covered in the example in Listing 4. However, data lineage also deals with the lifecycle of objects. Listing 5 is an example of lifecycle and data history, querying all the existing versions that existed through time of a certain object.

Listing 5. DAPOQ-Lang query to get all the versions of objects affected by a specific case.

```
return versionsOf(allCases where id == "77")
```

Dependency graphs/Relations: Data relations are a fundamental aspect of our approach. Relations hold between object versions during certain periods of time. Listing 6 shows that it is possible to query all the objects that have ever been related to another specific object.

Listing 6. DAPOQ-Lang query to get booking objects related to the customer "Customer35".

```
var _bookings = objectsOf(allClasses where name == "booking");
var _objectsC = objectsOf(versionsRelatedTo(
                    versionsOf(allClasses where name == "customer")
                    where at.CUSTOMER_NAME == "Customer35"));
return _objectsC INTERSECTION _bookings
```

3.3 Business Process Aspects

Business process oriented: Our language is able to query elements specific of business processes, such as processes, activities, logs, traces and events. Listing 7 demonstrates how to query concurrent activities for a certain process.

Listing 7. DAPOQ-Lang query to get all the activities of the same process that have ever happened concurrently with any "insert"activity for a process called "concert-organizing".

```
var _actCon = activitiesOf(allProcesses where name == "concert-organizing");
return concurrentWith(_actCon where name CONTAINS "insert") INTERSECTION _actCon
```

3.4 Database or Artifact Aspects

Entities/Artifacts: Being based on the meta model depicted by Fig. 4, the language considers objects as first class elements. Listing 8 shows how to obtain the objects affected by events in a certain period of time.

Listing 8. DAPOQ-Lang query to get all the objects affected by events that happened between a certain period of time.

```
return objectsOf(allEvents where (timestamp > "2014-01-31 13:45:23"
                         and timestamp < "2015-01-31 13:45:23"))
```

Database-oriented: Our language is essentially database oriented, and that is evident due to the fact that it considers data models, objects and versions. Listing 9 demonstrates how to obtain classes of a data model.

Listing 9. DAPOQ-Lang query to get all the classes of a data model named "concert-portal".

```
return classesOf(allDatamodels where name == "concert-portal")
```

State-aware: Given that the language is state-aware, it is possible to query the values of objects at certain moments in history (versions). Listing 10 shows how to get the state of a database at a certain moment in time.

Listing 10. DAPOQ-Lang query to get all the versions of objects of the class "customer"as they where at a certain point in time.

```
versionsOf(allClasses where name == "customer")
        where (start_timestamp =< "2014-01-31 13:45:23"
          and  end_timestamp => "2014-01-31 13:45:23")
```

3.5 Query Aspects

Relevance querying: Relevance querying refers to relations between data elements that not only reflect data origin. Our language is able to query relations of any nature. Listing 11 queries objects that were related at some point with certain object through a very specific relation.

Listing 11. DAPOQ-Lang query to get all the objects that have been related at some point to a certain ticket (35) by the relationship "ticket_concert_fk".

```
var _ticketVers = versionsOf(allClasses where name == "ticket") where at.id == "35";
return objectsOf(versionsOf(allRelationships where name == "ticket_concert_fk")
       INTERSECTION versionsRelatedTo(_ticketVers))
```

Temporal properties/querying: The meta model on which our language is based considers time attributes of different elements of the structure. This makes it easy to formulate queries with temporal properties with our language. For example, Listing 12 shows how to obtain activities executed during a certain period of time.

Listing 12. DAPOQ-Lang query to get all the activities that where executed during a period for the process "ticket-selling".

```
return activitiesOf(createPeriod("2014-01-31 13:45:23" , "2014-01-31 14:45:23"))
       INTERSECTION activitiesOf(allProcesses where name == "ticket-selling")
```

Event correlation: Events are not considered as single and isolated elements in DAPOQ-Lang. They can be correlated to cases, logs, versions, objects, etc. Listing 13 shows a query to obtain events correlated to the same object.

Listing 13. DAPOQ-Lang query to get all the events that affected any version of the object corresponding to the customer named "Customer35".

```
return eventsOf(objectsOf(versionsOf(allClasses where name == "customer")
                         where at.CUSTOMER_NAME == "Customer35"))
```

Multi process/Multi log/Multi data schema: An advantage of our approach not found in other approaches is the support to query properties of several processes, logs, and data schemas at the same time. Listings 14, 15 and 16 show examples of querying all the processes of several activities, the logs of some processes and the data schemas that contain a certain relationship.

Listing 14. DAPOQ-Lang query to get the processes with an activity which name contains the substring “insert”.

```
return processesOf(allActivities where name CONTAINS "insert")
```

Listing 15. DAPOQ-Lang query to get all the logs of the processes named “ticket-selling”.

```
return logsOf(allProcesses where name == "ticket-selling")
```

Listing 16. DAPOQ-Lang query to get the data models with a relationship “ticket_seat_pk”.

```
return datamodelsOf(allRelationships where name == "ticket_seat_pk")
```

4 Implementation

The query language (DAPOQ-Lang) proposed in the previous section has been implemented as a library[3] in Java, and integrated as part of the Process Aware Data Suite[4] (PADAS). The parser for the grammar has been created using ANTLR[5], a parser generator widely used to build languages and other tools. DAPOQ-Lang queries are executed on OpenSLEX[6] files, which store all the data according to the meta model defined in [28]. In addition to the query engine,

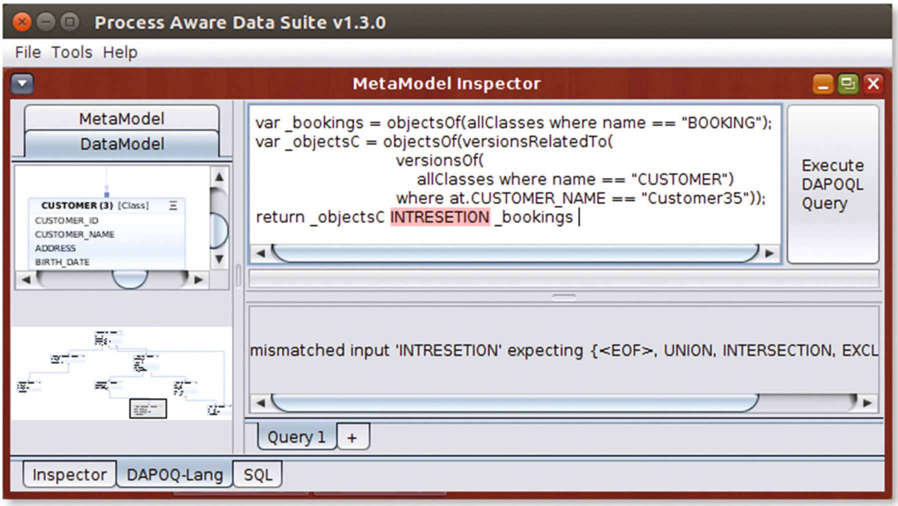

Fig. 5. Screenshot of the tool while writing a query, highlighting a misspelled word.

[3] https://www.win.tue.nl/~egonzale/projects/dapoq-lang/.
[4] https://www.win.tue.nl/~egonzale/projects/padas/.
[5] http://www.antlr.org/.
[6] https://www.win.tue.nl/~egonzale/projects/openslex/.

DAPOQ-Lang provides assistance on the query development by means of *predictive typing*. This functionality, implemented in the GUI of the tool, helps the user to write queries by suggesting language keywords as they are typed. Also, *syntax checking* is performed simultaneously, highlighting any problematic or incorrect parts of the query as shown in Fig. 5.

5 Conclusion

This work analyzes the existing approaches for querying execution trails such as process events, transactions and provenance data. The functionalities of these approaches have been identified and classified accordingly. As a result of this study, we have identified the need for a query mechanism that combines both process and data perspectives and helps with the task of obtaining insights about the process or workflow under analysis.

To fulfill this need, we proposed a new query language that, combining process and data perspectives, allows the analyst to ask meaningful questions in a simple way. Examples of the use of the language have been presented, covering different aspects of the data, in order to guarantee its usefulness and simplicity. In addition, we provide an implementation that not only enables the execution, but also assists on the writing and development of the queries by means of *predictive typing* and *syntax checking*.

The paper shows that it is feasible to develop a query language that satisfies the needs of process analysts, while keeping simplicity and ease of use. As future work, a full specification of the language will be provided. Also, efforts will be made to improve the performance when dealing with big datasets and to keep the language evolving, adding new functionalities and constructs.

References

1. Vanwersch, R., Shahzad, K., Vanhaecht, K., Grefen, P., Pintelon, L., Mendling, J., Van Merode, G., Reijers, H.A.: Methodological support for business process redesign in health care: a literature review protocol. Int. J. Care Pathways **15**(4), 119–126 (2011)
2. Moreau, L., Clifford, B., Freire, J., Futrelle, J., Gil, Y., Groth, P., Kwasnikowska, N., Miles, S., Missier, P., Myers, J., et al.: The open provenance model core specification (v1. 1). Future Gener. Comput. Syst. **27**(6), 743–756 (2011)
3. Huang, X., Bao, Z., Davidson, S.B., Milo, T., Yuan, X.: Answering regular path queries on workflow provenance. In: 2015 IEEE Proceedings of the 31st International Conference on Data Engineering (ICDE), pp. 375–386. IEEE (2015)
4. Costa, F., Silva, V., De Oliveira, D., Ocaña, K., Ogasawara, E., Dias, J., Mattoso, M.: Capturing and querying workflow runtime provenance with PROV: a practical approach. In: Proceedings of the Joint EDBT/ICDT 2013 Workshops, pp. 282–289. ACM (2013)
5. Cuevas-Vicenttin, V., Dey, S., Wang, M.L.Y., Song, T., Ludascher, B.: Modeling and querying scientific workflow provenance in the D-OPM. In: High Performance Computing, Networking, Storage and Analysis (SCC), pp. 119–128. IEEE (2012)

6. Sakka, M.A., Defude, B.: Towards a scalable semantic provenance management system. In: Hameurlain, A., Küng, J., Wagner, R. (eds.) Transactions on Large-Scale Data- and Knowledge-Centered Systems VII. LNCS, vol. 7720, pp. 96–127. Springer, Heidelberg (2012). doi:10.1007/978-3-642-35332-1_4
7. Chirigati, F., Freire, J.: Towards integrating workflow and database provenance. In: Groth, P., Frew, J. (eds.) IPAW 2012. LNCS, vol. 7525, pp. 11–23. Springer, Heidelberg (2012). doi:10.1007/978-3-642-34222-6_2
8. Gadelha, L.M., Wilde, M., Mattoso, M., Foster, I.: MTCProv: a practical provenance query framework for many-task scientific computing. Distrib. Parallel Databases **30**(5–6), 351–370 (2012)
9. Lim, C., Lu, S., Chebotko, A., Fotouhi, F.: OPQL: A first OPM-level query language for scientific workflow provenance. In: 2011 IEEE International Conference on Services Computing (SCC), pp. 136–143. IEEE (2011)
10. Lim, C., Lu, S., Chebotko, A., Fotouhi, F.: Storing, reasoning, and querying OPM-compliant scientific workflow provenance using relational databases. Future Gener. Comput. Syst. **27**(6), 781–789 (2011)
11. Liu, D.: XQuery meets Datalog: data relevance query for workflow trustworthiness. In: Research Challenges in Information Science (RCIS 2010), pp. 169–174. IEEE (2010)
12. Bowers, S., McPhillips, T., Ludäscher, B., Cohen, S., Davidson, S.B.: A model for user-oriented data provenance in pipelined scientific workflows. In: Moreau, L., Foster, I. (eds.) IPAW 2006. LNCS, vol. 4145, pp. 133–147. Springer, Heidelberg (2006). doi:10.1007/11890850_15
13. Solanki, M., Brewster, C.: A knowledge driven approach towards the validation of externally acquired traceability datasets in supply chain business processes. In: Janowicz, K., Schlobach, S., Lambrix, P., Hyvönen, E. (eds.) EKAW 2014. LNCS (LNAI), vol. 8876, pp. 503–518. Springer, Cham (2014). doi:10.1007/978-3-319-13704-9_38
14. Momotko, M., Subieta, K.: Process query language: a way to make workflow processes more flexible. In: Benczúr, A., Demetrovics, J., Gottlob, G. (eds.) ADBIS 2004. LNCS, vol. 3255, pp. 306–321. Springer, Heidelberg (2004). doi:10.1007/978-3-540-30204-9_21
15. Koksal, P., Arpinar, S.N., Dogac, A.: Workflow history management. ACM Sigmod Rec. **27**(1), 67–75 (1998)
16. Poppe, O., Giessl, S., Rundensteiner, E.A., Bry, F.: The HIT model: workflow-aware event stream monitoring. In: Hameurlain, A., Küng, J., Wagner, R., Amann, B., Lamarre, P. (eds.) Transactions on Large-Scale Data- and Knowledge-Centered Systems XI. LNCS, vol. 8290, pp. 26–50. Springer, Heidelberg (2013). doi:10.1007/978-3-642-45269-7_2
17. Liu, D., Pedrinaci, C., Domingue, J.: Semantic enabled complex event language for business process monitoring. In: Proceedings of the 4th International Workshop on Semantic Business Process Management, pp. 31–34. ACM (2009)
18. Fazzinga, B., Flesca, S., Furfaro, F., Masciari, E., Pontieri, L., Pulice, C.: A framework supporting the analysis of process logs stored in either relational or NoSQL DBMSs. In: Esposito, F., Pivert, O., Hacid, M.-S., Raś, Z.W., Ferilli, S. (eds.) ISMIS 2015. LNCS (LNAI), vol. 9384, pp. 52–58. Springer, Cham (2015). doi:10.1007/978-3-319-25252-0_6
19. Radeschütz, S., Schwarz, H., Niedermann, F.: Business impact analysis: a framework for a comprehensive analysis and optimization of business processes. Comput. Sci. Res. Dev. **30**(1), 69–86 (2015)

20. Backmann, M., Baumgrass, A., Herzberg, N., Meyer, A., Weske, M.: Model-driven event query generation for business process monitoring. In: Lomuscio, A.R., Nepal, S., Patrizi, F., Benatallah, B., Brandić, I. (eds.) ICSOC 2013. LNCS, vol. 8377, pp. 406–418. Springer, Cham (2014). doi:10.1007/978-3-319-06859-6_36
21. Song, L., Wang, J., Wen, L., Wang, W., Tan, S., Kong, H.: Querying process models based on the temporal relations between tasks. In: 2011 15th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW), pp. 213–222. IEEE (2011)
22. Beheshti, S.-M.-R., Benatallah, B., Motahari-Nezhad, H.R., Sakr, S.: A query language for analyzing business processes execution. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) BPM 2011. LNCS, vol. 6896, pp. 281–297. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23059-2_22
23. Deutch, D., Milo, T.: Top-K projection queries for probabilistic business processes. In: Proceedings of the 12th International Conference on Database Theory. ACM (2009)
24. Baquero, A.V., Molloy, O.: Integration of event data from heterogeneous systems to support business process analysis. In: Fred, A., Dietz, J.L.G., Liu, K., Filipe, J. (eds.) IC3K 2012. CCIS, vol. 415, pp. 440–454. Springer, Heidelberg (2013). doi:10.1007/978-3-642-54105-6_29
25. Metzke, T., Rogge-Solti, A., Baumgrass, A., Mendling, J., Weske, M.: Enabling semantic complex event processing in the domain of logistics. In: Lomuscio, A.R., Nepal, S., Patrizi, F., Benatallah, B., Brandić, I. (eds.) ICSOC 2013. LNCS, vol. 8377, pp. 419–431. Springer, Cham (2014). doi:10.1007/978-3-319-06859-6_37
26. Ray, M., Liu, M., Rundensteiner, E., Dougherty, D.J., Gupta, C., Wang, S., Mehta, A., Ari, I.: Optimizing complex sequence pattern extraction using caching. In: 2011 IEEE Proceedings of the 27th International Conference on Data Engineering Workshops (ICDEW), pp. 243–248. IEEE (2011)
27. Räim, M., Ciccio, C., Maggi, F.M., Mecella, M., Mendling, J.: Log-based understanding of business processes through temporal logic query checking. In: Meersman, R., et al. (eds.) OTM 2014. LNCS, vol. 8841, pp. 75–92. Springer, Heidelberg (2014). doi:10.1007/978-3-662-45563-0_5
28. González López de Murillas, E., Reijers, H.A., van der Aalst, W.M.P.: Connecting databases with process mining: a meta model and toolset. In: Proceedings of the 17th International Conference on Enterprise, Business-Process and Information Systems Modeling, BPMDS (2016)