University Library



A gateway to Melbourne's research publications

Minerva Access is the Institutional Repository of The University of Melbourne

Author/s:
Polyvyanyy, A

Title:
Business Process Querying

Date:
2018

Citation:
Polyvyanyy, A. (2018). Business Process Querying. SpringerLink.

Persistent Link:
https://hdl.handle.net/11343/225755

Business Process Querying

Artem Polyvyanyy

The University of Melbourne, Parkville, VIC, 3010, Australia
e-mail: artem.polyvyanyy@unimelb.edu.au

Synonyms

Process retrieval, process filtering, process management.

Definitions

Business process querying studies methods for managing, e.g., filtering and/or manipulating, repositories of models that describe observed and/or envisioned business processes, and relationships between the business processes.

A *process querying method* is a technique that given a process repository and a process query systematically implements the query in the repository, where a *process query* is a (formal) instruction to manage a process repository.

Overview

This encyclopedia entry summarizes the state of the art methods for (business) process querying proposed in the publications included in the literature reviews reported in (Wang et al 2014; Polyvyanyy et al 2017b), and the related works discussed in the reviewed publications. The scope of this entry is limited to the methods for process querying that are based on special-purpose (programming) languages for capturing process querying instructions and were developed in the areas of Process Mining (van der Aalst 2016) and Business Process Management (Weske 2012; Dumas et al 2013).

Next, this section lists basic concepts in process querying. Let $\mathcal{U}_{an}$ and $\mathcal{U}_{av}$ be the universe of *attribute names* and *attribute values*, respectively.

Event. An *event e* is a relation between some attribute names and attribute values with the property that each attribute name is related to exactly one attribute value, i.e., it is a partial function $e : \mathcal{U}_{an} \nrightarrow \mathcal{U}_{av}$.

Process. A *process* is a partially ordered set $\pi := (E, \leq)$, where E is a set of events and $\leq$ is a partial order over E.

Trace. A *trace* is a process $\pi := (E, <)$, where $<$ is a total order over E.

Behavior. A *behavior* is a multiset of processes.

Behavior model. A *behavior model* is a simplified representation of real world or envisioned behaviors, and relationships between the behaviors, that serves a particular purpose for a target audience.

Process repository. A *process repository* is an organized collection of behavior models.

Process query. A *process query* is an instruction to filter or manipulate a process repository.

Let $\mathcal{U}_{pr}$ and $\mathcal{U}_{pq}$ be the universe of *process repositories* and the universe of *process queries*, respectively.

Process querying method. A *process querying method m* is a relation between ordered pairs, in which the first entries are process repositories and the second entries are process queries, and process repositories with the property that each pair is related to exactly one process repository, i.e., it is a function $m : \mathcal{U}_{pr} \times \mathcal{U}_{pq} \rightarrow \mathcal{U}_{pr}$.

Framework

In (Polyvyanyy et al 2017b), a *process querying framework* is proposed. A schematic visualization of the framework is shown in Fig. 1.

The framework is an abstract system of components that provide generic functionality and can be selectively replaced to result in a new process querying method. In the figure, rectangles and ovals denote *active components* and *passive components*, respectively. Active components represent actions performed by the process querying methods. Passive components stand for (collections of) data objects. The framework is composed of four parts. These parts are responsible for (i) designing process repositories and process queries, (ii) preparing and (iii) executing process queries, and (iv) interpreting results of the process querying methods, refer to (Polyvyanyy et al 2017b) for details.

The framework proposes four types of behavior models: event logs, process models, simulation models, and correlation models. An *event log* is a finite collection of traces, where each trace is a finite sequence of events observed and recorded in the real-world. A process model is a conceptual model that describes behaviors. A simulation model is a process model with a part of its behavior. Finally, a correlation model is a model that describes relations between two behaviors. All the reported below process querying methods operate over event logs and process models.

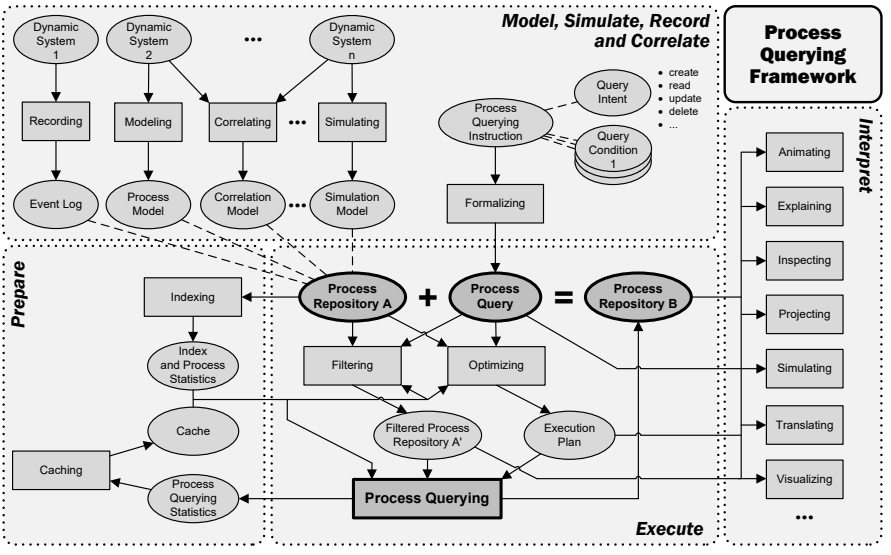


Fig. 1 A schematic view of the Process Querying Framework, refer to (Polyvyanyy et al 2017b).

Techniques

This section lists techniques that are often used as a basis for implementing process querying methods.

Model Analysis. Behavior models are often formalized as graphs. Several methods for process querying are grounded in the analysis of structural properties of graphs that encode behavior models, e.g., BP-QL, DMQL, GMQL, and VMQL. Examples of graph analysis tasks employed for process querying include the existence of a path problem and subgraph isomorphism problem.

SPARQL Protocol and RDF Query Language (SPARQL). SPARQL is a semantic query language for retrieving and manipulating data captured in Resource Description Framework (RDF) format (Hebeler et al 2009). SPARQL allows users to write queries against data stored in the form of a set of "subject-predicate-object" triples.

Several methods for process querying are based on SPARQL, e.g., BPMN VQL and FPSPARQL. These methods encode process repositories as RDF data and implement process querying by first transforming process queries into SPARQL queries and then executing SPARQL queries over the RDF data.

Given a model of a finite-state system, for example, a behavior model, and a formal property, *model checking* techniques verify whether the property holds for the system (Baier and Katoen 2008).

Temporal Logic. A temporal logic is a formal system for encoding and reasoning about propositions qualified in terms of time (Øhrstrøm and Hasle 2007). Several methods for process querying are grounded in model checking temporal logic properties, e.g., BPMN-Q, BPSL, CRL, and PPSL. Given a process

repository and process query, these methods proceed by translating the process query into a temporal logic expression, e.g., Linear Temporal Logic (LTL), Computation Tree Logic (CTL), and Metrical Temporal Logic (MTL). Then, each behavior model from the repository gets translated into a finite-state system and verified against the formula. Finally, behavior models that translate into systems for which the property captured in the formula holds constitute the result of the method.

First-order Logic. First-order Logic (FOL) is a formal logic that extends propositional logic which operates over declarative propositions with the use of predicates and quantifiers (Smullyan 1995). Several methods for process querying are grounded in model checking FOL properties, e.g., CRG and QuBPAL. These methods encode process repositories as formal FOL theories and implement process querying by verifying properties (encoded in process queries) of these systems.

Behavior Analysis. Several methods for process querying are grounded in the analysis of properties of behaviors encoded in models, e.g., APQL, BQL, and PQL. These methods support process queries that specify conditions over all the processes (of which there can be potentially infinitely many) encoded in the behaviors of models stored in a process repository.

Methods

This section discusses the state of the art methods for process querying. The methods can be seen as instantiations of the process querying framework, refer to Fig. 1, with various concrete components. In what follows, the methods are grouped based on the types of behavior models that they can take as input in process repositories and are referred to by the names of the corresponding languages for specifying process queries.

Log Querying

Process querying methods proposed in this section operate over event logs.

CRG&eCRG. Compliance Rule Graph (CRG) is a visual language for specifying compliance rules (Ly et al 2010). The semantics of CRG is defined based on FOL over event logs. Extended Compliance Rule Graph (eCRG) extends CRG to allow modeling compliance rules over multiple process perspectives (Knuplesch and Reichert 2017). At this stage, eCRG addresses the interaction, time, data, and resource perspectives of process compliance. Similar to CRG, the execution semantics of eCRG is defined based on FOL over event logs.

DAPOQ-Lang. The Data-Aware Process Oriented Query Language (DAPOQ-Lang) allows querying over events in traces of event logs, as well as over related data objects, their versions and data schemas, and temporal properties of all the aforementioned elements (de Murillas et al 2016).

FPSPARQL. The Folder-Path enabled extension of SPARQL (FPSPARQL) is a query language founded on the two concepts of folders and paths that allow grouping events and exploring se-

quences of events recorded in event logs (Beheshti et al 2011).

FPSPARQL operates over a data model that encodes an event log as a graph. In such a graph, nodes are events from the event log and edges stem from the orderings of events in the traces of the event log. Folders and paths are results of FPSPARQL queries. They constitute abstractions over the event log graphs. A folder abstraction refers to a collection of events, whereas a path abstraction encodes one or several paths in the transitive closure of the graph. Finally, FPSPARQL allows querying logs using standard SPARQL capabilities.

PIQL. Process Instance Query Language (PIQL) is an extension of Decision Model And Notation (DMN) for querying process performance indicators over executed traces (Pérez-Álvarez et al 2016). DMN is a standard language for describing logic of decisions within organizations. PIQL is a textual language that resembles natural language. It can be used during process execution to define decision logic that depends on the information kept in the so far executed traces.

Model Querying

This section lists methods for querying collections of process models. These methods can be split into those that operate over process model specifications and those that explore behaviors encoded in process models. Methods that operate over model specifications can be further split into two subclasses. The first subclass includes methods for querying conceptual models that were also reported useful for querying process models. The second subclass consists of dedicated techniques for querying process model collections.

DMQL, GMQL, and VMQL are the methods that belong to the first subclass of methods that operate over process model specifications.

DMQL&GMQL. The Generic Model Query Language (GMQL) is a textual query language designed to query collections of conceptual models created in arbitrary graph-based modelling languages (Delfmann et al 2015b). GMQL querying is implemented via searching for model subgraphs that correspond to a predefined pattern query. GMQL is proposed to query process models, as well as data models, organizational charts, and other types of models.

The Diagramed Model Query Language (DMQL) (Delfmann et al 2015a) extends GMQL with a visual concrete syntax and support for additional requirements, e.g., ability to approximate analysis of the execution semantics of process models in the case of loops.

VMQL. The Visual Model Query Language (VMQL) is a language for querying collections of conceptual models (Störrle 2009, 2011). Among other types of conceptual models, the authors of VMQL demonstrated its use over collections of process models, e.g., Unified Modeling Language (UML) Activity Diagrams and Business Process Model and Notation (BPMN) models. VMQL is based on the concept of a host language. For example, VMQL for querying BPMN models subsumes the syntax and notation of BPMN. This approach allows decreasing the user effort for reading and writing VMQL queries. VMQL queries can also specify

constraints that go beyond the syntax of the host language.

For querying purposes, process models and VMQL queries are encoded in a Prolog database. Executing a VMQL query with respect to a given model requires finding parts of the model that resemble the structure and properties of the query and satisfy the constraints.

Next, we discuss dedicated process querying methods that operate over model specifications.

BPMN-Q. BPMN query language (BPMN-Q) was originally proposed to query business process definitions, i.e., (Awad 2007) the flow between activities, branching and joining structures, and data flow. BPMN-Q extends the abstract and concrete syntax of BPMN. Hence, BPMN-Q is a visual language. Later, BPMN-Q was extended to serve as a visual interface to temporal logics, e.g., LTL (Awad et al 2008) and CTL (Awad et al 2011).

BPMN VQL. BPMN Visual Query Language (BPMN VQL) is a visual query language (Francescomarino and Tonella 2008). The syntax and semantics of BPMN VQL are grounded in BPMN and SPARQL, respectively. The SPARQL translation of BPMN VQL queries is based on a formalization of the BPMN meta-model as an ontology. BPMN VQL queries consist of two parts. The matching part of a query specifies a structural condition to match in process models, whereas the selection part specifies parts of models to retrieve as query result.

BPSL. The Business Property Specification Language (BPSL) is a visual language that aims to facilitate the reasoning over business process models (Liu et al 2007). The authors proposed direct translations of BPSL queries in both LTL and CTL to simplify the integration of BPSL with the existing formal verification tools. The language proposes dedicated operators for the recurring patterns in regulatory process compliance and specifies extension points for the definition and reuse of domain-specific temporal properties.

CRL. Compliance Request Language (CRL) is a visual language that is grounded in temporal logic and encompasses a range of compliance patterns (Elgammal et al 2016). At this stage, CRL supports compliance patterns that address control flow, resource, and temporal aspects of business processes. During execution, control flow and resource patterns get mapped to LTL, whereas temporal patterns are translated into MTL formulas. MTL extends LTL and is interpreted over a discrete time domain using the digital clock model.

Descriptive PQL. Descriptive Process Query Language (Descriptive PQL) is a textual query language for retrieving process models, specifying abstractions on process models, and defining process model changes (Kammerer et al 2015). The aforementioned operations are grounded in manipulations with Single-Entry-Single-Exit subgraphs (Polyvyanyy et al 2010) of the process models.

IPM-PQL. Integrated Process Management-Process Query Language (IPM-PQL) is an XML based language to query process model collections based on structure matching (Choi et al 2007). IPM-PQL queries follow the structure of SQL queries and support four types of query conditions: process-has-attribute,

process-has-activity, process-has-sub-process, and process-has-transition. When executed, IPM-PQL queries are transformed into XML Query (XQuery) expressions.

PPSL. The Process Pattern Specification Language (PPSL) is a light-weight extension of the language for capturing UML Activity Diagrams that allows users to model process constraints, e.g., those related to quality management (Förster et al 2005). The semantics of the language is given as translations of PPSL queries into temporal logic, viz. LTL formulas (Förster et al 2007).

Finally, we summarize process querying methods that operate over behaviors encoded in models.

APQL. A Process-model Query Language (APQL) is a query language based on semantic relations between occurrences of tasks specified in process models (ter Hofstede et al 2013). The language allows composing the relations between tasks to obtain complex queries. APQL is precise, i.e., its semantics is defined over all possible execution traces of process models. The language is independent of any particular notation used to specify process models.

BQL. Business Query Language (BQL) is a textual query language based on semantic relations between occurrences of tasks specified in process models (Jin et al 2011). Given a repository of process models, BQL can be used to retrieve the models satisfying the requirements based on the behavior encoded in the models. To compute the relations, BQL relies on the technology of complete prefix unfolding (McMillan 1995).

QuBPAL. Querying Business Process Abstract modeling Language (QuBPAL) is an ontology-based language for the retrieval of process fragments to be reused in the composition of new business processes (Smith et al 2010). Using QuBPAL, the user can query over three process perspectives: the structure of process specifications, the behavioral semantics of process specifications, and the pre-modelled domain knowledge about the business scenaria encoded in models. QuBPAL queries are evaluated over Business Process Knowledge Base—a collection of FOL theories that formalize the process repository of semantically enriched process models.

PQL. Process Query Language (PQL) is a textual query language based on semantic relations between tasks in process models (Polyvyanyy et al 2015). PQL extends the abstract syntax of APQL and has an SQL-like concrete syntax. The semantic relations between tasks are computed as aggregations over all the possible occurrences of these tasks, as specified in the behavior of the corresponding process model.

PQL is based on the semantic relations of the 4C spectrum (Polyvyanyy et al 2014)—a systematic collection of the co-occurrence, conflict, causality, and concurrency relations. In addition to filtering capabilities, PQL addresses manipulation of process models. For example, PQL queries can specify instructions to insert traces into process models, i.e., to augment process models to additionally describe fresh traces. PQL implements instructions to insert new traces into process models as solutions to the process model repair problem (Polyvyanyy et al 2017a).

Log and Model Querying

Methods listed below were proposed to query over collections that may include process models and/or event logs.

BPQL. Business Process Query Language (BPQL) is a textual language for querying over process models and event logs (Momotko and Subieta 2004). The language is defined as an extension of the Stack-Based Query Language (SBQL) (Subieta 2009). The semantics of the language is defined over the proposed abstract model for capturing process specifications and execution traces. The authors of BPQL proposed BPQL templates to use for monitoring process execution and integrated the language with the XML Process Definition Language (XPDL).

BP-QL. Business Process Query Language (BP-QL) is a visual language that allows retrieving paths in process specifications (Beeri et al 2006). BP-QL operates over a dedicated abstract model for representing collections of processes. This model is used to distinguish queries that can be efficiently computed, queries that have high computation costs, and queries that cannot be computed at all. When a BP-QL query is evaluated, its structural patterns are matched against process specifications. BP-QL uses the proposed abstract model to support queries that can be answered in time polynomial in the size of a process specification. This is achieved at a price of preciseness of query answers. BP-QL supports mechanisms for constructing concise finite representations of the (possibly infinite) query results. BP-Mon and BP-Ex are the two extensions of BP-QL to support process monitoring and querying of executed traces, respectively.

Applications

Due to their generic purpose, i.e., manipulation and filtering of process repositories, process querying methods have broad application in Process Mining and Business Process Management. The aforementioned process querying methods were proposed and/or successfully applied to solve various problems in these fields. Some examples of the addressed problems include: business process compliance management, business process weakness detection, process variance management, process model translation, syntactical correctness checking, process model comparison, infrequent behavior detection, process instance migration, process monitoring, process reuse, and process standardization.

Future Directions for Research

Future work on process querying will contribute to achieving *process querying compromise* (Polyvyanyy et al 2017b), i.e., will propose new and improve the existing process querying methods with the support of efficiently computable and practically relevant process queries. As of today, the existing works on process querying constitute non-coordinated efforts. Future work will contribute to the consolidation of various process querying methods leading to the definition of a standard meta-model for behavior models and behaviors that these models de-

scribe, and a standard language for capturing process queries.

Most existing methods for process querying operate over specifications of behavior models. Future work will lead to new methods for querying based on behaviors encoded in models and study their practical implications.

While there is a plethora of methods and languages for capturing instructions for filtering process repositories, methods for manipulating process repositories are still in their infancy.

Cross-References

These articles can be of further interest to the reader:

- Event logs;
- Business process model repositories;
- Business process model matching;
- Graph query languages;
- Graph pattern matching.

References

van der Aalst WMP (2016) Process Mining – Data Science in Action, 2nd edn. Springer Berlin Heidelberg, DOI 10.1007/978-3-662-49851-4

Awad A (2007) BPMN-Q: A language to query business processes. In: Enterprise Modelling and Information Systems Architectures, GI, LNI, vol P-119, pp 115–128

Awad A, Decker G, Weske M (2008) Efficient compliance checking using BPMN-Q and temporal logic. In: Business Process Management, Springer, Lecture Notes in Computer Science, vol 5240, pp 326–341, DOI 10.1007/978-3-540-85758-7_24

Awad A, Weidlich M, Weske M (2011) Visually specifying compliance rules and explaining their violations for business processes. Journal of Visual Languages & Computing 22(1):30–55, DOI 10.1016/j.jvlc.2010.11.002

Baier C, Katoen J (2008) Principles of Model Checking. MIT Press

Beeri C, Eyal A, Kamenkovich S, Milo T (2006) Querying business processes. In: Very Large Data Bases, ACM, pp 343–354

Beheshti S, Benatallah B, Nezhad HRM, Sakr S (2011) A query language for analyzing business processes execution. In: Business Process Management, Springer Berlin Heidelberg, Lecture Notes in Computer Science, vol 6896, pp 281–297, DOI 10.1007/978-3-642-23059-2_22

Choi I, Kim K, Jang M (2007) An XML-based process repository and process query language for integrated process management. Knowledge and Process Management 14(4):303–316, DOI 10.1002/kpm.290

Delfmann P, Breuker D, Matzner M, Becker J (2015a) Supporting information systems analysis through conceptual model query – the diagramed model query language (DMQL). Communications of the Association for Information Systems 37:24

Delfmann P, Steinhorst M, Dietrich H, Becker J (2015b) The generic model query language GMQL – conceptual specification, implementation, and runtime evaluation. Information Systems 47:129–177, DOI 10.1016/j.is.2014.06.003

Dumas M, Rosa ML, Mendling J, Reijers HA (2013) Fundamentals of Business Process Management. Springer, DOI 10.1007/978-3-642-33143-5

Elgammal A, Turetken O, Heuvel WJ, Papazoglou M (2016) Formalizing and appling compliance patterns for business process compliance. Software & Systems Modeling 15(1):119–146, DOI 10.1007/s10270-014-0395-3

Förster A, Engels G, Schattkowsky T (2005) Activity diagram patterns for modeling quality constraints in business processes. In: Model Driven Engineering Languages and Systems, Springer, Lecture Notes in Computer Science, vol 3713, pp 2–16, DOI 10.1007/11557432_2

Förster A, Engels G, Schattkowsky T, Straeten RVD (2007) Verification of business process quality constraints based on visual process patterns. In: Theoretical Aspects of Soft-

ware Engineering, IEEE Computer Society, pp 197–208, DOI 10.1109/TASE.2007.56

Francescomarino CD, Tonella P (2008) Crosscutting concern documentation by visual query of business processes. In: Business Process Management Workshops, Springer Berlin Heidelberg, Lecture Notes in Business Information Processing, vol 17, pp 18–31, DOI 10.1007/978-3-642-00328-8_3

Hebeler J, and Ryan Blace and Andrew Perez-Lopez MF, Dean M (2009) Semantic Web Programming, 1st edn. Wiley

ter Hofstede AHM, Ouyang C, Rosa ML, Song L, Wang J, Polyvyanyy A (2013) APQL: A process-model query language. In: Asia Pacific Business Process Management, Springer, Lecture Notes in Business Information Processing, vol 159, pp 23–38, DOI 10.1007/978-3-319-02922-1_2

Jin T, Wang J, Wen L (2011) Querying business process models based on semantics. In: Database Systems for Advanced Applications, Springer Berlin Heidelberg, Lecture Notes in Computer Science, vol 6588, pp 164–178, DOI 10.1007/978-3-642-20152-3_13

Kammerer K, Kolb J, Reichert M (2015) PQL – A descriptive language for querying, abstracting and changing process models. In: Enterprise, Business-Process and Information Systems Modeling, Springer, Lecture Notes in Business Information Processing, vol 214, pp 135–150, DOI 10.1007/978-3-319-19237-6_9

Knuplesch D, Reichert M (2017) A visual language for modeling multiple perspectives of business process compliance rules. Software & Systems Modeling 16(3):715–736, DOI 10.1007/s10270-016-0526-0

Liu Y, Müller S, Xu K (2007) A static compliance-checking framework for business process models. IBM Systems Journal 46(2):335–362, DOI 10.1147/sj.462.0335

Ly LT, Rinderle-Ma S, Dadam P (2010) Design and verification of instantiable compliance rule graphs in process-aware information systems. In: Advanced Information Systems Engineering, Springer Berlin Heidelberg, Lecture Notes in Computer Science, vol 6051, pp 9–23, DOI 10.1007/978-3-642-13094-6_3

McMillan KL (1995) A technique of state space search based on unfolding. Formal Methods in System Design 6(1):45–65, DOI 10.1007/BF01384314

Momotko M, Subieta K (2004) Process query language: A way to make workflow processes more flexible. In: Advances in Databases and Information Systems, Springer Berlin Heidelberg, Lecture Notes in Computer Science, vol 3255, pp 306–321, DOI 10.1007/978-3-540-30204-9_21

de Murillas EGL, Reijers HA, van der Aalst WMP (2016) Everything you always wanted to know about your process, but did not know how to ask. In: Business Process Management Workshops: Process Querying, Lecture Notes in Business Information Processing, vol 281, pp 296–309, DOI 10.1007/978-3-319-58457-7_22

Øhrstrøm P, Hasle P (2007) Temporal Logic: From Ancient Ideas to Artificial Intelligence. Studies in Linguistics and Philosophy, Springer Netherlands

Pérez-Álvarez JM, López MTG, Parody L, Gasca RM (2016) Process instance query language to include process performance indicators in DMN. In: IEEE Enterprise Distributed Object Computing Workshops, IEEE Computer Society, pp 1–8, DOI 10.1109/EDOCW.2016.7584381

Polyvyanyy A, Vanhatalo J, Völzer H (2010) Simplified computation and generalization of the refined process structure tree. In: Web Services and Formal Methods, Springer, Lecture Notes in Computer Science, vol 6551, pp 25–41, DOI 10.1007/978-3-642-19589-1_2

Polyvyanyy A, Weidlich M, Conforti R, Rosa ML, ter Hofstede AHM (2014) The 4C spectrum of fundamental behavioral relations for concurrent systems. In: Application and Theory of Petri Nets and Concurrency, Springer International Publishing, Lecture Notes in Computer Science, vol 8489, pp 210–232, DOI 10.1007/978-3-319-07734-5_12

Polyvyanyy A, Corno L, Conforti R, Raboczi S, Rosa ML, Fortino G (2015) Process querying in Apromore. In: Business Process Management Demo Session, CEUR-WS.org, CEUR Workshop Proceedings, vol 1418, pp 105–109

Polyvyanyy A, van der Aalst WMP, ter Hofstede AHM, Wynn MT (2017a) Impact-driven process model repair. ACM Transactions

on Software Engineering and Methodology 25(4):1–60, DOI 10.1145/2980764

Polyvyanyy A, Ouyang C, Barros A, van der Aalst WMP (2017b) Process querying: Enabling business intelligence through query-based process analytics. Decision Support Systems 100:41–56, DOI 10.1016/j.dss.2017.04.011

Smith F, Missikoff M, Proietti M (2010) Ontology-based querying of composite services. In: Business System Management and Engineering, Springer Berlin Heidelberg, Lecture Notes in Computer Science, vol 7350, pp 159–180, DOI 10.1007/978-3-642-32439-0_10

Smullyan RM (1995) First-order Logic. Courier Corporation

Störrle H (2009) VMQL: A generic visual model query language. In: IEEE Visual Languages and Human-Centric Computing, IEEE Computer Society, pp 199–206, DOI 10.1109/VLHCC.2009.5295261

Störrle H (2011) VMQL: A visual language for ad-hoc model querying. Journal of Visual Languages & Computing 22(1):3–29, DOI 10.1016/j.jvlc.2010.11.004

Subieta K (2009) Stack-based query language. In: Encyclopedia of Database Systems, Springer US, pp 2771–2772, DOI 10.1007/978-0-387-39940-9_1115

Wang J, Jin T, Wong RK, Wen L (2014) Querying business process model repositories – A survey of current approaches and issues. World Wide Web 17(3):427–454, DOI 10.1007/s11280-013-0210-z

Weske M (2012) Business Process Management – Concepts, Languages, Architectures, 2nd edn. Springer Berlin Heidelberg, DOI 10.1007/978-3-642-28616-2