Efficient & Expressive Semantic Information Push for Cultural Heritage Applications

Christos Tryfonopoulos

Dept. of Informatics and Telecommunications,
University of the Peloponnese, Tripoli, Greece
trifon@uop.gr

Abstract. In this work, we apply the semantic information push paradigm in the domain of cultural heritage and advocate for its usefulness in a number of diverse scenarios ranging from personalised content delivery to museum visitors to the curation of huge knowledge bases that integrate diverse cultural assets. We envision a large-scale semantic information push system that is able to perform efficient filtering of multiple RDF data streams based on expressive subscriptions that aim both for the structure and content of the stream. To this end, we put forward STIP, a new algorithm that indexes user subscriptions and utilises its index structures to efficiently match them against the stream of RDF events; STIP proves four orders of magnitude faster than its baseline competitor in an experimental evaluation with real-world data. To the best of our knowledge, this is the first approach in the literature to propose the usage of information push –along with an appropriate algorithm– as the technological substrate for a variety of high-level cultural heritage applications such as personalisation and recommender systems.

1 Introduction

As the Semantic Web initiative finds new and challenging application fields such as the annotation, integration, and linking of physically distributed, continuously evolving, inherently diverse and practically invaluable digital cultural heritage data, it becomes increasingly difficult for any stakeholder in this field to stay on top of the created data avalanche. Museum visitors are becoming increasingly more demanding in receiving a personalised museum experience [1, 10], humanities researchers become increasingly interested in discovering new facets and stories from new or existing cultural heritage data [4, 18, 27], while data scientists strive to provide accurate, semantically rich, interoperable, searchable, inter-linked cultural data repositories [2, 3, 15, 28] that are able to realise the vision of Semantic Web in the cultural heritage domain.

It is not difficult to observe that knowledge bases/graphs are the key technological factor that cross-cuts all the aforementioned cases. For the museum visitor the solution lies in the appropriate *exploitation* of the knowledge base, for the humanities researcher the solution lies in appropriate *evolution monitoring* mechanisms for the existing knowledge bases, while for the data scientist the solution lies in appropriate knowledge base *curation* mechanisms that facilitate easy integration and maintenance. In this work, we advocate that *semantic*

information push (often referred to as publish/subscribe, information dissemination, or information alert) is a technological paradigm that may be transparently applied to all the aforementioned cases and provide appropriate solutions for the different semantic data/information management needs of each user type.

Information push systems [5,11,14,16,26,30,32] have emerged as a promising paradigm that enables the user to cope with the high rate of information production and avoid the cognitive overload of repeated searches. In an information push system, users (or services that act on users' behalf) express their interests by submitting a *subscription* (often referred to also as continuous query or profile) and wait to be *notified* whenever a new *event* of interest occurs. The vast majority of modern information push services and systems are typically content-based (contrary to previous decades, where they used to be topic/channel based); users (often referred to as subscribers) express their (explicit or implicit) interest on the content of the publication (be it structure or data/text values) by appropriately specifying *constraints* in the submitted subscriptions. The information push system stores these subscriptions and matches them against the stream of published events to create *notifications* (that are delivered to the subscribers) every time a subscription matches a published event.

In this work, we propose a *novel* algorithm, coined STIP (Structural & Textual Information Push), designed to efficiently support *expressive* subscriptions (specified by users themselves or by services that act on users' behalf) with structural and textual constraints that can be used to filter *RDF data streams* in evolving knowledge bases. This algorithm is a good candidate for a cultural heritage setup since it is designed to suit different user needs and RDF streams with high rates. These characteristics make STIP a versatile solution that can be used as a *lower-level building block* for high-level semantically-aware cultural heritage applications such as profiling, personalisation, knowledge base quality control, and recommender systems [1, 10, 25]. In the light of the above the contributions of this work are threefold:

- We advocate the application of semantic information push paradigm in the cultural heritage domain by highlighting diverse application scenarios and useful subscriptions for each scenario. To the best of our knowledge, this is the first work in the literature to propose semantic information push as an enabler technology for different digital cultural heritage applications. Information push will enrich the technological arsenal of digital cultural heritage and will provide functionality beyond current state-of-the-art in the area.
- We adopt the RDF data model to define evolving knowledge graphs that contain both structural and textual information and subsequently propose an extension to the SPARQL syntax to support Boolean textual (in addition to the structural) subscriptions in RDF streams.
- We develop a proof-of-concept algorithm that indexes structural and textual subscriptions in a unified data structure and experimentally evaluate it against a brute force baseline to showcase the efficiency benefits that emerge from such indexing. Such an algorithm may function as an efficiency substrate to realise higher-level cultural heritage applications such as personalisation and recommender systems.

The rest of the paper is organised as follows. Section 2 discusses different application scenarios for semantic information push within the cultural heritage

domain, while Section 3 presents an overview of our data and query model. Subsequently, Section 4 presents algorithm STIP alongside its experimental evaluation with a real-world knowledge base. Finally, Section 5 discusses related work, and Section 6 provides future research directions.

2 Information push for cultural heritage

In this section, we provide three distinct application scenarios that demonstrate the versatility of the proposed algorithm and highlight its usefulness in diverse cultural heritage setups.

Information push for end users. Angela is a casual museum visitor and a World War II (WWII) aficionado. Since she is always interested in exhibitions and events about WWII, she has created a profile in a specialised service where users, through an appropriate user interface, create textual and structural subscriptions with constraints about events or topics of interest. Angela has utilised the textual constraints to explicitly specify a number of interests including poetry (e.g., by providing her favourite poets, styles, or subject), WWII, and art. Moreover, the system (or an independent service that acts on Angela's behalf) has augmented Angela's subscription with a number of textual and structural constraints based on her favourite reads over the last period of time, her visited web pages, and other implicit signals (such as current location). Angela is currently visiting a new destination and is now on her way to visit a museum of contemporary art, when she receives a notification that on her way to the museum there is a local WWII antique fair. After stopping by at the fair, Angela visits the museum, when her attention is drawn on an oil on paper painting by a Japanese artist. When she uses the QR code near the exhibit to get a description of the work on her phone, she receives a notification (based on her submitted subscription) that delivers content about the connection of the painting to the WWII bombing of Nagasaki.

Information push for researchers. Costas is a post-graduate student from an Art History department following an MSc in the history of History of Art. Costas is mainly interested in retrieving scientific publications on his topic of interest and also following the work of prominent researchers in the area. Due to the particularities of his research field[1], he regularly resorts to (semantically-aware) online resources –like Semantic Scholar[2] or Microsoft Academic[3]– to search for new works or new developments/interpretations of existing works in his area of study. Even though searching for interesting/related works this week turned up nothing, a search next week may return new information. Clearly, an information push system that is able to integrate a number of relative sources and also

[1] Art History is neither static nor does it develop in a linear or deterministic manner, as is the case with natural sciences or computer science. For example, although the work of a 19th century's artist might has already been inventoried, analysed and dated, hence classified within a School or an art tendency, this does not imply that it will not be reconsidered or reinterpreted, possibly more than once, in the future using a different methodology or newly available information.

[2] `https://www.semanticscholar.org`

[3] `https://academic.microsoft.com/`

capture his long-term information need (in the form of a subscription) would be a valuable tool that would allow him to save both time and effort. Having submitted a relevant subscription, Costas will be notified when an event (e.g., a paper, a re-interpretation of an existing piece of art, or an art review of a prolific author in the field) that matches his expressed interests is published.

Information push for curators. Nikki is a computer scientist working in an organisation that provides support in the construction and maintenance of a collaborative home-brewed knowledge base for cultural heritage applications. In this context, Nikki is responsible for the curation of the knowledge base and the enrichment/integration with existing Semantic Web resources like DBpedia[4] and various online thesauri (e.g., the Getty Art & Architecture Thesaurus[5]) using both manual/crowdsourced and automatic techniques [17, 25]. As this knowledge base is continuously evolving, monitoring its quality over time becomes an essential task. Having access to an appropriate information push system integrated with the actual knowledge base would allow Nikki and other moderators to subscribe (with appropriate textual and structural constraints) and get notified about (i) spurious and/or unusual connections in the knowledge graph, (ii) the creation/evolution of different structures, patterns, and subgraphs, and (iii) the trending of specific items. Such functionality would be an invaluable tool that would simplify graph moderation as Nikki would, for example, be notified about an unusual connection created by a graph update action that mistakenly links the painting "Alexander and his Doctor," to the oratorio composer "Le Sueur, Jean-Francois", instead of its actual creator, painter "Le Sueur, Eustache".

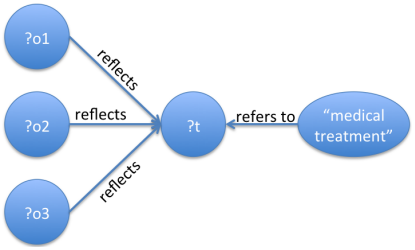
From the above scenarios, it becomes clear that an efficient algorithm, able to support expressive semantic information push for thousands of users and support high event rates, would be a valuable substrate to a number of cultural heritage applications. This is also highlighted by the exploitation of such approaches in other contexts, e.g., in web [22] and textual [31] information management, distributed setups [9], and middleware architectures [29].

3 Data and query model

Typically, knowledge bases/graphs evolve over time through (i) the addition of new entities and facts (corresponding to graph vertices or edges), either due to new relationships that emerge or due to better harvesting methods and richer data repositories that become available, and (ii) the deletion of relationships and entities either for noise removal (e.g., to delete wrong facts) or duplicate elimination. In our modelling the knowledge graph is defined as a directed labeled multigraph; RDF is used to represent the knowledge graph as a set of *(subject, predicate, object)* triples. A triple is an atomic entity that represents a statement about its subject; the subject and object part of the triple can either be unique resources (represented as $URIs$) or literals, while the predicate represents the relation between the subject and object. A graph update performed on the knowledge graph is defined as an *addition* or a *deletion*. An addition update results in the addition of new relations (edges) between existing and/or

[4] `http://wiki.dbpedia.org`

[5] `ttp://www.getty.edu/research/tools/vocabularies/lod/index.html/`




```
1  SELECT ?event
2  WHERE {?event type Artifact.
3          ?event title ?title.
4          ?event description ?descr.
5  FILTER contains
6   (?title, "alexander" NEAR[0,1] "great")
7  FILTER contains
8   (?descr, "doctor" AND
9   ("friendship" or "trust"))
10 }
```

Fig. 1: Examples of a structural and a textual constraint.

new vertices of the graph. Each triple that is inserted into the knowledge graph represents a labelled edge –having *predicate* as label– between vertices with labels *subject* and *object*. Contrary, a deletion update results in the removal of relations (edges) in the graph; the deletion of a vertex is defined as a series of deletions on all (incoming and outgoing) edges of that vertex. Thus, an evolving knowledge graph is represented as a *stream* or *ordered sequence* of updates, i.e., timestamped edge additions and deletions.

In the proposed query model, user subscriptions (specified by users themselves or by services that act on users' behalf) may contain both structural and textual constraints. For the structural part, a user subscribes with subgraphs or motifs that emerge through the evolution of the knowledge graph and match a given set of structural and attribute constraints. In this context, we use *subscription graph patterns* to capture the subgraph, and attribute restrictions of an evolving knowledge graph as directed labeled multigraphs.

Apart from structural constraints, we are also interested in providing Boolean textual constraints over RDF streams; thus for the textual part we are interested in property elements with a plain RDF literal as their content. In this context, the subject of an RDF triple is always a node element, the predicate denotes the relation to the object, which is expressed as a string. In the spirit of [6], we propose an extension to the SPARQL syntax to support Boolean textual subscriptions in RDF streams; to preserve SPARQL expressibility we view the textual operations as an additional filter of the subscription variables. In this context, we define a new binary operator *contains*, that takes as input a *variable* of the SPARQL subscription and a *Boolean textual expression* that operates on the values of this variable. The subscription signature of the operator is expressed as the function $xsd : boolean : contains(var, expr)$. A textual expression is evaluated only against a literal, so *var* is always the object of the SPARQL tuple pattern; the subject and/or predicate of the tuple pattern may be constants. The expressions supported involve the usual Boolean operators, phrase, and word proximity operators as in [8]. In addition to the textual extension of SPARQL, we also support the *wildcard (*)* operator applied in RDF triples, i.e., queries where the subject, predicate and/or object of a triple may match any value of the publication. An event, in this context, is represented as a set of RDF triples containing additional fields, where needed, to store the text parts. Hence, the overall underlying model is a directed graph, which contains a set of nodes that may serve as the subject or the object in a triple statement, and are connected via properties that are expressed as the predicate.

Figure 1 shows examples of a structural and a textual constraint in the cultural heritage domain; the example structural constraint will create a notification when publications form a graph that matches it (essentially this is a subgraph isomorphism problem between the constraint and the graph created by the publication stream). The example textual constraint in Figure 1 will match all publications that are of type *Artifact* and have an attribute *title* with a string literal. The title of the publications must contain the term "great" at least zero and at most one words after the term "alexander". Additionally, the publications that match must have an attribute *description* that contains the term "doctor" and at least one of the terms "friendship" and "trust".

4 Algorithms and evaluation

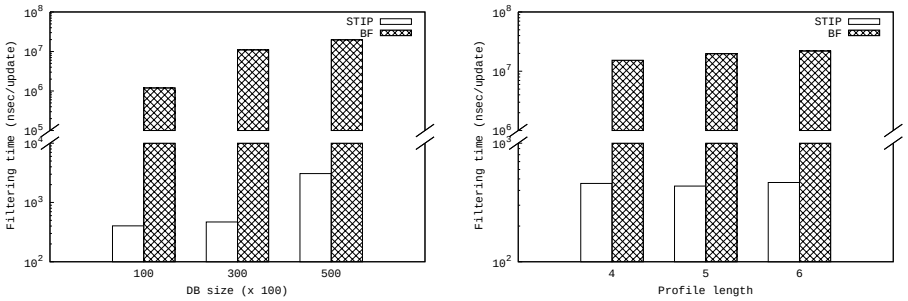
In this section, we outline two algorithms for semantic information push that are able to match subscriptions (with both textual and structural constraints) against graph updates and present a concise experimental evaluation of their performance. Our findings highlight the need for efficiency in the support of expressive semantic information push.

4.1 Algorithm STIP and competitor

In the context of our proof-of-concept implementation, we developed two algorithms that are able to filter subscriptions aiming at both structural and textual properties of the evolving graph. In our setup, subscriptions were arbitrary graphs with textual constraints that express user interests; subscriptions were subsequently indexed depending on the filtering algorithm at hand. The evolving graph (against which the subscriptions are constantly evaluated) is also indexed to allow for faster matching against the subscription database.

Algorithm STIP indexes the subscriptions in an appropriate data structure as follows. It decomposes the subscription to the vertex pairs that form the subscription graph and uses these pairs as keys to store the subscription identifier at an *inverted index*. In this way, a subscription graph with k edges is decomposed into k vertex pairs of the form (*subject*,*object*) and its identifier is stored at k different inverted index positions, one for each pair. An auxiliary table T is used to store the number of vertex pairs that are contained in each subscription and the number of already matched pairs. The same inverted index infrastructure is used to index also the textual constraints (using the terms as the index keys) in the spirit of [26, 30]. When a new graph update is published, the vertices that are involved in the update are used to locate all affected subscriptions in the inverted index and appropriately update T with the newly matched pairs. Subsequently, T is scanned to determine whether any newly matching subscriptions (i.e., subscriptions that have *all* their pairs matched) have arisen as a result of the new update. If so, the appropriate notifications are created and sent to the subscribed users.

To highlight the need for efficiency in semantic information push we have also implemented algorithm BF (Brute Force), that is used to serve as a simple baseline. BF has no indexing strategy and scans the subscription database sequentially on every graph update to determine matching subscriptions. The BF



(a) (b)

Fig. 2: Efficiency of STIP and BF when varying D and L.

algorithm stores the full subscription database in a linked list using an appropriate representation that allows it to match (at event publication time) each subscription against the indexed (evolving) graph.

Finally, notice that the semantics and the effectiveness of our approach are given by (i) subgraph isomorphism for the structural constraints and (ii) the adopted Boolean model for the textual constraints. Thus, in our setup, subscriptions are either satisfied or not, i.e., there are no false-positive or false-negative results. For more details on the effectiveness of the Boolean model the interested reader is referred to [24].

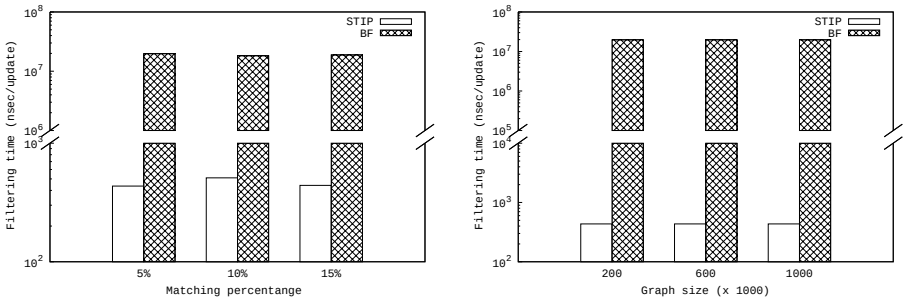
4.2 Experimental evaluation

In this section, we present a series of experiments that compare the performance of STIP and BF. Initially, we present the experimental setup and we subsequently discuss the evaluation results.

Evaluation setup. We utilised a fraction of the *DBpedia* dataset as our evolving graph; to simulate the graph evolution, we obtained a snapshot of $1M$ triples from the original graph and simulated the graph evolution by adding those triples to an (initially empty) graph. The publication events (graph updates) resulted to a directed graph of total size $S = 1M$ edges and more than $1.2M$ vertices.

Since no benchmarking database of subscriptions is publicly available, we used the *DBpedia* graph to artificially generate realistic subscription databases of varying sizes and characteristics. The generated subscriptions were equiprobably chosen to be chains, stars, or arbitrary graphs of various sizes, while 10% of them had a textual constraint. The baseline values for our evaluation were: (i) subscription databases D of size $10K, 30K$, and $50K$ entries, (ii) average subscription length L of $4, 5$ and 6 triples/query, (iii) percentage P of subscriptions that matched the final graph controlled at $5, 10$ and 15% of D, and (iv) total graph size S of $200K, 600K$ and $1M$ edges.

All algorithms were implemented in Java and were executed on a commodity PC running Ubuntu Linux. Graph indexing was performed with the JGraphT library, and the time shown in the graphs is wall-clock time averaged over 10 runs to eliminate fluctuations in time measurements. Please notice the cut and



(a) (b)

Fig. 3: Efficiency of STIP and BF when varying P and S.

the log scale in the y-axis of all graphs; this aims at better illustrating the two algorithms that have large performance differences.

Evaluation results. Figure 2(a) shows the filtering time required to match a graph update against a database of stored subscriptions when varying the size of the subscription database D, while using $L = 5$, $P = 5\%$, and $S = 1M$. We observe that the filtering time of all algorithms increases with the subscription database size; STIP is *four orders of magnitude faster* in filtering an update compared to BF. In Figure 2(b) we compare the filtering performance of STIP and BF for varying values of query length L, when $D = 50K$, $P = 5\%$, and $S = 1M$. The observations here are consistent with those of the previous experiment; STIP is four orders of magnitude faster than BF. Additionally, Figure 3(a) shows the filtering performance of STIP and BF when varying the percentage of subscriptions P that will match the final graph, for $D = 50K$, $L = 5$, and $S = 1M$. Results in this experiment regarding the performance of the algorithms are also in line with the previous ones. Finally, as expected, the performance of both algorithms is not affected from the size of the evolving graph (Figure 3(b)).

In summary, for a database of 50K subscriptions, algorithm STIP is able to support a steady throughput of more than 2M updates/sec in contrast to its competitor that supports four times lower throughput rates. This significant difference between STIP and BF highlights the need for efficient subscription indexing structures that will enable the use of efficient and expressive semantic information push in the cultural heritage domain.

5 Related work

In the early days of semantic information push, the structure of a publication was nothing more than a (usually static) collection of named attributes with values of different types (e.g., text) [26, 30]. Later, and as XML gained popularity and started becoming the standard for data/information representation and exchange on the web, various XML-based information push systems have arisen [5, 7, 11–13, 21]. Publications were, at that time, expressed in XML and extensions of XPath/XQuery were used to express subscriptions. Since structural/value matching of publications and subscriptions was unable to capture the underlying semantics, ontology-based information push systems [19,22,23,29,32]

had emerged. Those systems typically used RDF for representing publications and SPARQL extensions/modifications for expressing user interests through subscriptions. Although this research direction has produced interesting results, it still currently lacks important functionality that is provided with STIP, such as subgraph pattern matching and textual constraints.

Many RDF stores (Apache Jena[6] text module, Virtuoso[7], Allegrograph[8], OntoText GraphDB[9]) also offer text indexing and retrieval combined with SPARQL. However, these solutions are targeted to the information pull (retrieval) paradigm which copes with different problems and challenges compared to our setup.

Apart from centralised solutions, there is a number of works that focus on distributed/P2P approaches for semantic information push [14,16,20]. However, none of these works is able to support both structural and textual subscriptions. Finally, notice that it is possible to extend STIP in a decentralised environment (such as a P2P or cloud setup) to enhance scalability by distributing the index among different nodes and modifying the filtering process to visit only the nodes that may contain matching subscriptions.

6 Outlook

We are currently working on a prototype semantic information push system for the cultural heritage domain based on the presented ideas. We also plan to extend this work by (i) devising more sophisticated indexing to exploit commonalities between subscriptions, (ii) supporting subscriptions under the Vector Space Model, and (iii) adapting our solutions for distributed/parallel environments.

References

1. Antoniou, A., Katifori, A., Roussou, M., Vayanou, M., Karvounis, M., Kyriakidi, M., Pujol-Tost, L.: Capturing the visitor profile for a personalized mobile museum experience: an indirect approach. In: UMAP (2016)
2. Bampatzia, S., Bravo-Quezada, O.G., Antoniou, A., López Nores, M., Wallace, M., Lepouras, G., Vassilakis, C.: The use of semantics in the crosscult h2020 project. In: KEYSTONE (2017)
3. Bartalesi, V., Meghini, C.: Using an ontology for representing the knowledge on literary texts: The dante alighieri case study. Semantic Web 8(3), 385–394 (2017)
4. Bourlakos, I., Wallace, M., Antoniou, A., Vassilakis, C., Lepouras, G., Karapanagiotou, A.V.: Formalization and visualization of the narrative for museum guides. In: KEYSTONE (2018)
5. Bruno, N., Gravano, L., Koudas, N., Srivastava, D.: Navigation- vs. Index-Based XML Multi-Query Processing. In: ICDE (2003)
6. Case, P., Dyck, M., Holstege, M., Amer-Yahia, S., Botev, C., Buxton, S., Doerre, J., Melton, J., Rys, M., Shanmugasundaram, J.: XQuery and XPath Full Text. (2011), `http://www.w3.org/TR/xpath-full-text-10/`
7. Chan, C.Y., Felber, P., Garofalakis, M.N., Rastogi, R.: Efficient Filtering of XML Documents with XPath Expressions. In: ICDE (2002)

[6] `http://jena.apache.org/`

[7] `http://virtuoso.openlinksw.com/`

[8] `http://franz.com/agraph/allegrograph/`

[9] `http://ontotext.com/products/graphdb/`

8. Chang, C.C.K., Garcia-Molina, H., Paepcke, A.: Predicate Rewriting for Translating Boolean Queries in a Heterogeneous Information System. ACM TOIS (1999)
9. Chirita, P.A., Idreos, S., Koubarakis, M., Nejdl, W.: Publish/Subscribe for RDF-based P2P Networks. In: ESWS (2004)
10. Deladiennee, L., Naudet, Y.: A graph-based semantic recommender system for a reflective and personalised museum visit. In: SMAP (2017)
11. Diao, Y., Altinel, M., Franklin, M.J., Zhang, H., Fischer, P.M.: Path sharing and predicate evaluation for high-performance XML filtering. ACM TODS (2003)
12. Green, T.J., Gupta, A., Miklau, G., Onizuka, M., Suciu, D.: Processing XML streams with deterministic automata and stream indexes. ACM TODS (2004)
13. Hou, S., Jacobsen, H.: Predicate-based Filtering of XPath Expressions. In: ICDE (2006)
14. Kaoudi, Z., Miliaraki, I., Koubarakis, M.: RDFS Reasoning and Query Answering on Top of DHTs. In: ISWC (2008)
15. Kyvernitou, I., Bikakis, A.: An ontology for gendered content representation of cultural heritage artefacts. Digital Humanities Quarterly 11(3) (2017)
16. Liarou, E., Idreos, S., Koubarakis, M.: Publish/Subscribe with RDF Data over Large Structured Overlay Networks. In: DBISP2P (2005)
17. Marketakis, Y., Minadakis, N., Kondylakis, H., Konsolaki, K., Samaritakis, G., Theodoridou, M., Flouris, G., Doerr, M.: X3ML mapping framework for information integration in cultural heritage and beyond. IJDL 18(4), 301–319 (2017)
18. Meghini, C., Bartalesi, V., Metilli, D., Benedetti, F.: A software architecture for narratives. In: IRCDL (2018)
19. Park, M.J., Chung, C.W.: iBroker: An Intelligent Broker for Ontology Based Publish/Subscribe Systems. In: ICDE (2009)
20. Pellegrino, L., Huet, F., Baude, F., Alshabani, A.: A Distributed Publish/Subscribe System for RDF Data. In: GLOBE (2013)
21. Peng, F., Chawathe, S.S.: XPath Queries on Streaming Data. In: SIGMOD (2003)
22. Petrovic, M., Burcea, I., Jacobsen, H.A.: S-ToPSS: Semantic Toronto Publish/-Subscribe System. In: VLDB (2003)
23. Petrovic, M., Liu, H., Jacobsen, H.A.: G-ToPSS: fast filtering of graph-based metadata. In: WWW (2005)
24. Salton, G., Fox, E.A., Wu, H.: Extended boolean information retrieval. CACM (1983)
25. Stavropoulos, T.G., Kontopoulos, E., Meroño-Peñuela, A., Tachos, S., Andreadis, S., Kompatsiaris, Y.: Cross-domain semantic drift measurement in ontologies using the semadrift tool and metrics. In: MEPDaW & LDQ @ ESWC (2017)
26. Tryfonopoulos, C., Koubarakis, M., Drougas, Y.: Information filtering and query indexing for an information retrieval model. TOIS (2009)
27. Vassilakis, C., Poulopoulos, V., Antoniou, A., Wallace, M., Lepouras, G., Nores, M.L.: exhistory: Smart exhibits that tell their own stories. FGCS 81, 542–556 (2018)
28. Vlachidis, A., Bikakis, A., Kyriaki-Manessi, D., Triantafyllou, I., Antoniou, A.: The crosscult knowledge base: A co-inhabitant of cultural heritage ontology and vocabulary classification. In: ADBIS (2017)
29. Wang, J., Jin, B., Li, J.: An Ontology-Based Publish/Subscribe System. In: Middleware (2004)
30. Yan, T.W., Garcia-Molina, H.: The SIFT Information Dissemination System. ACM TODS (1999)
31. Zervakis, L., Tryfonopoulos, C., Skiadopoulos, S., , Koubarakis, M.: Query Reorganisation Algorithms for Efficient Boolean Information Filtering. IEEE TKDE (2017)
32. Zervakis, L., Tryfonopoulos, C., Skiadopoulos, S., Koubarakis, M.: Full-text Support for Publish/Subscribe Ontology Systems. In: ESWC (2016)