A Preliminary Approach on Ontology-based Visual Query Formulation for Big Data

Ahmet Soylu[1], Martin G. Skjæveland[1], Martin Giese[1], Ian Horrocks[2], Ernesto Jimenez-Ruiz[2], Evgeny Kharlamov[2], and Dmitriy Zheleznyakov[2]

[1] Department of Informatics, University of Oslo, Norway
{ahmets, martige, martingi}@ifi.uio.no
[2] Department of Computer Science, University of Oxford, United Kingdom
{name.surname}@cs.ox.ac.uk

Abstract. Data access in an enterprise setting is a determining factor for the potential of value creation processes such as sense-making, decision making, and intelligence analysis. As such, providing friendly data access tools that directly engage domain experts (i.e., end-users) with data, as opposed to the situations where database/IT experts are required to extract data from databases, could substantially increase competitiveness and profitability. However, the ever increasing volume, complexity, velocity, and variety of data, known as the Big Data phenomenon, renders the end-user data access problem even more challenging. Optique, an ongoing European project with a strong industrial perspective, aims to countervail the Big Data effect, and to enable scalable end-user data access to traditional relational databases by using an ontology-based approach. In this paper, we specifically present the preliminary design and development of our ontology-based visual query system and discuss directions for addressing the Big Data effect.

Keywords: Visual Query Formulation, Ontology-based Data Access, Big Data, End-user Data Access, Visual Query Systems.

1 Introduction

A tremendous amount of data is being generated every day both on the Web and in public and private organisations; and, by all accounts, in this increasingly data-oriented world, any individual or organisation, who posses the necessary knowledge, skills, and tools to make value out of data at such scales, bears a considerable advantage in terms of competitiveness and development. Particularly, in an enterprise setting, ability to access and use data in business processes such as *sense-making* and *intelligence analysis* is key for its *value creation* potential (cf. [1]). Today, however, *data access* still stands as a major bottleneck for many organisations. This is mostly due to the sharp distinction between employees who have technical skills and knowledge to extract data (i.e., *database/IT experts*, *skilled users* etc.) and those who have domain knowledge and know how to

interpret and use data (i.e., *domain experts*, *end-users* etc.). The result is a workflow where domain-experts either have to use pre-defined queries embedded in applications or communicate their *information needs* to database-experts. The former scenario is quite limiting, since it is not possible to enumerate every possible information need beforehand, while the latter scenario is hampered by the ambiguity in communication. In such a workflow, the turn-around time from users' initial information needs to receiving the answer can be in the range of weeks, incurring significant costs (cf. [2]).

Approaches that eliminate the man-in-the-middle and allow end-users to directly engage with data and extract it on their own, have been of interest to researchers for many years. As anticipated, for end-users, the accessibility of traditional structured query languages such as *SQL* and *XQuery* fall far short, since such textual languages do require end-users to have a set of technical skills and to *recall* domain concepts and the terminology and syntax of the language being used. For this very reason, *visual query systems and languages* (cf. [3]) have emerged to alleviate the end-user data access problem by providing intuitive visual query formulation tools. A visual system or language follows the *direct-manipulation* idea (cf. [4]), where the domain and query language are represented with a set of visual elements. End-users *recognise* relevant fragments of the domain and language and formulate queries basically by instantly manipulating them. A good deal of research on visual query formulation has been carried out both for *structured* (e.g., relational data) and *semi-structured data* (e.g., XML), such as *QBE* [5] and *Xing* [6]. Early approaches (cf. [3]) successfully establish the fundamentals of research on visual query formulation. However, on the one hand, their success, in practical terms, remains within the confines of abstraction levels they operate on; database schemas, object-oriented models etc. are not meant to capture a domain per se and are not truly natural for end-users. Recent approaches (e.g., [7,8]) close this gap by employing ontologies for visual query formulation, due to their closeness to reality; and the emergence of *ontology-based data access* (OBDA) technologies (cf. [9]) complete the overall picture by making it possible to access data residing on traditional relational databases over ontologies. On the other hand, visual query formulation, still being a considerable challenge in itself, faces inevitable scalability issues both in terms of *query answering* and *query formulation* (aka *query construction*), mainly introduced by the ever increasing *volume*, *complexity*, *velocity*, and *variety* of data – the so-called *Big Data* phenomenon (cf. [10,11]).

In this respect, a European project named *Optique*[1] – *Scalable End-user Access to Big Data*, with an industrial perspective, has been undertaken in order to enable end-user data access to traditional relational databases and to countervail the *Big Data effect* characterised by the aforementioned four dimensions. To this end, Optique employs an ontology-based approach for scalable query formulation and evaluation, along with other techniques such as query optimisation and parallelisation. The project involves two industrial partners, namely *Statoil*[2]

[1] http://www.optique-project.eu
[2] http://www.statoil.com

and *Siemens*[3], which provide real-life use cases. In this paper, we specifically present the preliminary design and development of our ontology-based visual query system and discuss directions for addressing the Big Data effect.

The rest of the paper is structured as follows. Section 2 sets the main research context, while Section 3 presents the related work. Section 4 describes our preliminary query formulation system, in terms of its architecture and interface. Finally, a discussion and conclusion are given in Section 5.

2 Background

Visual query formulation is indeed an *end-user development* (EUD) practice (cf. [12]), where the goal is to allow end-users to program without significant programming skills and knowledge. The evaluation criteria are *expressiveness*, i.e., the breadth of a system or language to characterise the domain knowledge and information need, and *usability*, i.e., in terms of *effectiveness*, *efficiency*, and *user-satisfaction* (cf. [3]). From the usability point of view, the selection of appropriate *visual attributes* (i.e., *perceptual* such as size, texture, and colour), *representation paradigms* (i.e., *cognitive* such as forms and diagrams) and *interaction styles* (e.g., navigation, range selection etc.), that lead end-users to *discern*, *comprehend*, and *communicate* a maximal amount of information with minimum effort, is of the utmost importance. From the expressiveness point of view, one must identify and support frequently repeating query tasks and necessary domain constructs, that could be easily communicated and realised by end-users through a visual tool. At this point, the difference between *visual query languages* (VQL) and *visual query systems* (VQS) comes into play. A VQL is a language that has a well-defined syntax and formal semantics independent of how queries are constructed, while a VQS is a system of interactions between a user and a computer, with or without an underlying visual language, that generates queries (cf. [13]). A VQS has a natural advantage over a VQL, since users might forget languages, but common knowledge and skills are mostly non-volatile (cf. *syntactic/semantic model* of user behaviour [4]). In any case, there are basically two types of activities, namely *exploration* (aka *understanding the reality of interest*) and *query construction* [3], that have to be supported by a data access system. The goal of the former is to establish an understanding of the domain by means of finding and identifying domain constructs, such as concepts and relationships, and their organisation. The goal of the latter is to formally express the information need. Exploration and construction have adverse (i.e., breadth vs. depth), yet complementary roles; therefore, they have to be addressed and intertwined adequately.

Visual query formulation relies on a domain model to enable end-users to communicate with the system. Experimental research suggests that approaches built on logical models, such as database schemata and object role models, are not as effective as conceptual approaches, where interaction is in terms of real world concepts and hence more natural [14]. In this respect, the use of

[3] http://www.siemens.com

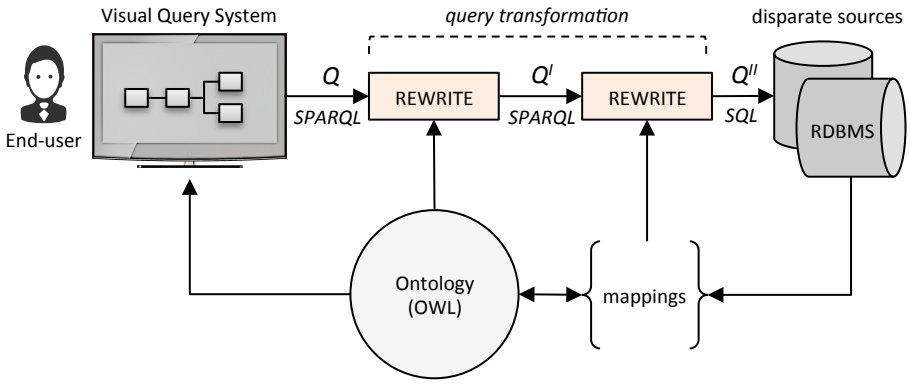

Fig. 1. Ontology-based visual query formulation and OBDA.

ontologies for query formulation is quite promising, since they are semantically richer and close the gap between the end-user's understating of reality and the system's representation of it. Moreover, ontologies, due to their reasoning power, provide the capability of expressing more with less both in the query formulation stage and the answering stage by relating the whole set of implied information instead of what is explicitly stated and available. However, almost all of the world's enterprise data today resides in relational databases. At this point, the *Semantic Web* [15] and OBDA technologies (e.g., [16,17]) play an essential role by bridging traditional relational data sources and ontologies. An ontology-based VQS falls into the category of *ontology-driven information systems* (cf. [18]). The OBDA approach, over relational databases, is typically built on *mappings* (cf. [19]), to relate the concepts and relations of the ontology to data sources, and *query rewriting* (cf. [17]), to transform queries into complete, correct, and highly optimised queries over possibly disparate data sources (see Fig. 1). As such, an ontology-based VQS employs the visual representations of terms coming from an ontology (e.g., an OWL ontology) for visual query formulation. Once a query is formulated by an end-user, it is extracted in the form of an underlying formal linguistic language (e.g., SPARQL). Then, a query transformation process takes place with two query rewriting phases. The former, by taking constraints coming from the ontology into account, rewrites the query in the same language; while the latter translates query into the language of underlying data sources (e.g., SQL) through mappings defined between ontology and data sources.

Visual query formulation is still an open challenge, yet the Big Data effect has substantially rescaled the problem. Firstly, the volume and complexity dimensions hinder human perception and cognition respectively. A data access system, therefore, has to orient and guide users within the large conceptual space and should provide the right amount of information in intuitive forms. Secondly, the variety dimension necessitate more specific presentations and interaction experiences adapted to data at hand at any moment, while the velocity

dimension requires data access systems to address *reactive* scenarios, where data is automatically *detected*, *assessed*, and *acted upon*. Ontologies have potential to address some of these new challenges; however, in general, a data access system, should support users in various ways (e.g., visualisations, recommendations etc.) and should be integrated and adapted into the *context*, such as *personal*, *data-related*, *task-related*, and *organisational* (cf. [20,21]).

3 Related Work

Early approaches to visual query formulation rely on low level models (i.e., database schemas) (cf. [3]), while recent ontology-based approaches mostly target open web data (i.e., *linked data browsers*) (cf. [22]). Data access on the Web is inherently different from traditional database systems, where information needs have to be precisely described with a very weak tolerance for missing or irrelevant results. However, the apparent success of web search makes it a sensible direction to adapt web search approaches to traditional settings. *Faceted search* (cf. [23]) and *Query by Navigation* (QbN) (cf. [24]) are prominent in terms of their suitability for ontology-based query formulation and their inherent compatibility. Faceted search, being an advanced form-based approach, is based on series of orthogonal dimensions, that can be applied in combination to filter the information space; each dimension, called *facet*, corresponds to a taxonomy. In its most common form, each facet option is accompanied with the number of accessible instances upon a possible selection. This is to prevent users from reaching empty result sets. QbN exploits the graph-based organisation of information to allow users to construct queries by traversing the relationships between concepts. Each navigation from one concept to another is indeed a join operation. Actually, end-users are quite familiar with both types of search approaches; faceted search is widely used in commercial websites such as eBay and Amazon for listing and filtering products, while the navigation is the backbone of web browsing. Since, there is a fair share of primary query construction operations, i.e., *select* and *project* for faceted search and *join* for QbN, their combination is promising.

Examples of QbN are *Tabulator* [25], *SEWASIE project* [7], *ViziQuer* [26], and *Visor* [27], and well-known examples of faceted search are *Flamenco* [28], *mSpace* [29], and *Exhibit* [30]. The examples of first category provide weak or no support for select and projection operations; similarly the examples on the latter do not provide sufficient support for joining concepts. The hybrid of QbN and faceted search is available in two forms in the literature. The former is built on *menu-based* QbN; the prominent examples are *Parallax* [31], *Humboldt* [32], and *VisiNav* [33]. The latter is built on *diagram-based* QbN; the prominent examples are *OZONE* [34] and *gFacet* [35]. In menu-based QbN, domain concepts are repented in the form of lists and a user navigates by selecting a concept from the list every time; while, in a diagram-based QbN, concepts are represented as nodes in a graph and a user navigates by expanding and retracting nodes. Moving from once concept to another changes the focus (i.e., *pivoting*) and the user can impose constraints on the active concept by selecting options within each facet.

However, the problem with these approaches is their strong focus on web data, which leads them to be highly explorative and instance oriented. That is, firstly, the result of user navigation in the conceptual space is mostly for data browsing purposes; a final query, which encompasses the visited concepts, is not generated. Hence, there is no clear distinction between explorative and constructive user actions and there is a lack of support for *view* (i.e., the active phase of a query task) and *overview* (i.e., the general snapshot of a query task). Secondly, a frequent interaction with the data is required (i.e., *database-intensive*), which is problematic with large scale data sources. The scalability problem is indeed more severe, since no support for tackling with large ontologies is provided.

4 Optique Approach

Our goal, from an architectural perspective, is extensibility and flexibility to ensure scalability and adaptivity to different contexts and needs, and, from a human-interaction perspective, is to minimise both the perceptual and cognitive load on users and to provide intuitive and natural experiences.

4.1 Architecture

The approach we pursue is built on *widget-based user-interface mashups* (i.e., UI mashups), which aggregate a set of applications in a common graphical space, in the form of *widgets*, and allow an interplay between them for achieving common goals (cf. [36]). In our context, widgets are the building blocks of a UI mashup and refer to *portable*, *self-contained*, *full-fledged*, and mostly *client side* applications with lesser functionality and complexity. In a query formulation scenario, a set of widgets could be employed together. For instance, one widget for QbN and one for faceted search could handle query construction synchronously, and one widget could represent query results in a tabular form.

Widgets are managed by a *widget run-time environment*, which provides basic *communication and persistence services* to widgets. The *orchestration* of widgets relies on the requirement that each widget discloses its functionality to the environment through a client side interface and notifies any other widget in the environment (e.g., broadcast, subscription etc.) and/or the widget environment upon each user action. Then, either each widget decides what action to execute in response, by considering the *syntactic* or *semantic signature* of the received event; or the environment decides on widgets to invoke. The core benefits of such an approach are that it becomes easier to deal with the complexity, since the management of functionality and data could be delegated to different widgets; each widget could employ a different representation paradigm that best suits its functionality; widgets could be used alone or together, in different combinations, for different contexts and experiences; and the functionality of the overall interface could be extended by introducing new widgets.

The preliminary architecture for our query formulation system is depicted in Fig. 2. The architecture assumes that each widget has client side and server side

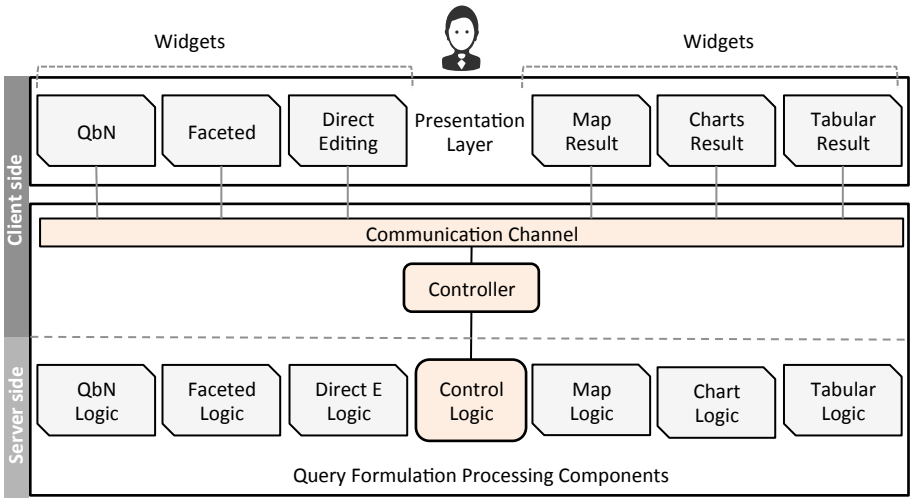

Fig. 2. OptiqueVQS architecture based on widget-based UI mashups.

components (for complex processing), and that widgets can communicate with each other and with the environment through a *communication channel*. Communication usually happens through the client side, but a server side communication mechanism could also be realised in order to support remote experiences (i.e., widgets running on distributed devices). There exists an *environment controller* at the client side and a *component control logic* at the server side. The former is responsible for operational tasks such as collecting event notifications from widgets and submitting control commands to them. The latter is responsible for the orchestration logic, that is, it decides how widgets should react to specific events. The widget specification of the W3C[4] and the widget run-time environment proposed in [36] guide our architectural design. Note that the architecture depicted here only concerns the visual query formulation system; the overall Optique architecture which includes other core components, such as for query evaluation, ontology management and evolution, mappings, and distributed query execution, has been discussed in another publication (cf. [37]).

4.2 Interface

The choice of visual representation and interaction paradigm, along with underlying *metaphors*, *analogies* etc., is of primary importance for the query formulation interface. We have observed that a single representation and interaction paradigm is not sufficient for developing successful query formulation interfaces. Therefore, we strive to combine the best parts of different paradigms (cf. [3]).

[4] http://www.w3.org/TR/widgets/

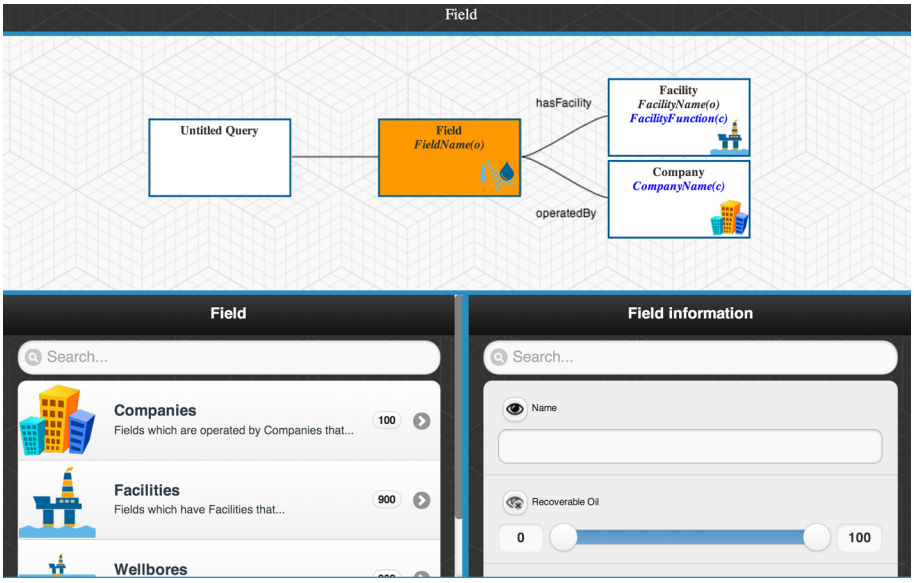

Fig. 3. OptiqueVQS interface – an example query is depicted.

We have implemented a preliminary visual query system, *OptiqueVQS*,[5] which is depicted in Fig. 3. The OptiqueVQS is initially composed of three widgets. The first widget (W1), see the bottom-left part of Fig. 3, employs menu-based representation and QbN interaction paradigms. This widget also supplements domain concepts with meaningful icons and short descriptions. The role of this widget is to allow end-users to navigate concepts through pursuing relationships between them, hence joining relations in a database. Every time a concept is selected, the focus of the interfaces changes to the selected concept (i.e, *pivot operation*). The second widget (W2), see the top part of Fig. 3, follows a diagram-based representation paradigm with QbN; it utilises geometric symbols to depict relationships among schema concepts in a graph. The role of this widget is to provide an overview of the constructed query; it also supports pivoting limited to the concepts involved in a query. Every node appearing in the diagram corresponds to a variable (i.e., of SPARQL) and called *variable-node*. The last widget (W3), see the bottom-right part of Fig. 3, is meant to employ a faceted search approach. However, in the current form, it follows a generic form-based representation paradigm. The role of this widget is to allow end-users to identify attributes that will be included in the result list and to place constraints on the attributes of the active (i.e., focus/pivot) concept. W1 and W3 support view (i.e., the active phase), while W2 supports overview (i.e., the snapshot). Concerning the design rationale behind each individual component, in terms of representation

[5] http://sws.ifi.uio.no/project/optique/pubshare/mtsr2013/

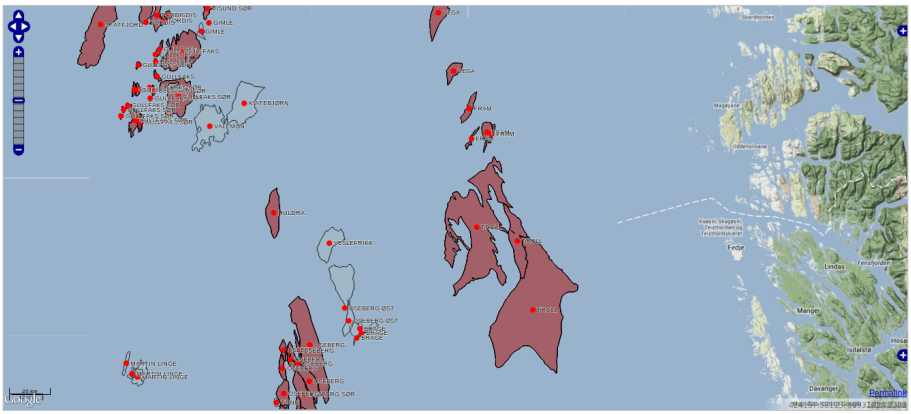

Fig. 4. OptiqueVQS interface – the result of an example query is depicted.

paradigm, lists (i.e., W1) are good at communicating large amount of information, forms (i.e., W3) are a well-known metaphor for end-users, and a diagram-based approach (i.e., W2) is good at communicating relationships between concepts; and, in terms of interaction paradigm, navigation (i.e., W1 and W2), matching, and range selection (i.e., W3) are known to be intuitive (cf. [3,38]).

In a typical query construction scenario, a user first selects a *kernel concept*, i.e., the starting concept, from W1, which initially lists all available domain concepts. The selected concept immediately becomes the focus concept (coloured in orange), appears in the the graph (i.e., W2), its attributes are displayed by W3, and W1 displays all the concept-relationship pairs pertaining to this concept. The user can select attributes to be included in the result list (i.e., using the "eye" button of each attribute) and/or impose constraints on them through form elements (i.e., W3). Note that W1 does not purely present relationships, but combine relationship and concept pairs (i.e., relationship and range) into one selection; this helps us to reduce the number of navigational levels a user has to pass through. The user can select any available option from the list and this results in a join between two variable-nodes over the specified relationship and moves focus to the selected concept. The user has to follow the same steps to involve new concepts into the query and can always jump to a specific part of the query by clicking on the corresponding variable-node on the graph. An example query is depicted in Fig. 3 for the Statoil use case in energy domain. The query asks for all fields which has a specific facility (e.g., an oil producing platform) and is operated by a specific company (e.g., Statoil).

Regarding the access to result lists, the system will provide several specialised presentation widgets specific to the nature of the data. An example has been depicted in Fig. 4, which shows all fields on a map, colouring those that are operated by Statoil and have an oil producing facility. In this context, it is important that widgets should have the ability to intelligently detect the data

type (e.g., items with a geo-spatial extent) and act accordingly, which is not hard to realise in an ontology-driven environment. The data source and widgetised map application used in this example comes from a project providing a semantic gateway to the Norwegian Petroleum Directorate's FactPages dataset [39].

5 Discussion and Conclusion

In this paper, we have introduced a preliminary ontology-based approach on query formulation for Big data. As far as the approach itself is concerned, the multi-paradigm approach that we follow firstly allows us to provide a good balance between view and overview. Secondly, one should be aware that an ontology is more than a class hierarchy and includes complex axioms. In our context, each representation paradigm could handle different kinds of ontology axioms, for instance, a faceted search paradigm is better suited for representing disjointness, a menu-based paradigm with QbN may be a better option for handling cyclic queries, and a diagrammatic representation is better in visualising them.

Concerning the expressiveness, we categorise queries into three levels with respect to their *perceived complexity*. First level corresponds to simple *three-shaped conjunctive queries*, while the second level refers to *cyclic* and *disjunctive queries*. The last level corresponds to queries with *universal quantifiers*, and *negation*. We postulate that most of the end-user queries will be centred around first level. The current interface at the moment addresses the first level queries and basic cycles (i.e., where a concept appears twice). However, we do see possibilities to address second level and third level queries to support advanced users.

At this stage, the current proposal attacks the query formulation challenge itself; our work for addressing the Big Data effect is under progress. Particularly for large ontologies, guiding users among hundreds of concepts, attributes, and relationships is of crucial importance. The possible approaches that we have identified include *adaptations*, that take previous interaction/query logs into account, for pruning the navigational space and to provide recommendations; *heuristics*, that consider what really occurs in data; and *annotations* to rule out unreasonable cases. Such information could be used to rank concepts, relationships, and attributes and then to gradually present them to users.

Acknowledgements. This research is funded by the FP7 of the European Commission under Grant Agreement 318338, "Optique".

References

1. Nunameker, J.F., Briggs, R.O., de Vreede, G.J.: From Information Technology to Value Creation Technology. In: Information Technology and the Future Enterprise: New Models for Managers. Prentice-Hall, New York (2001) 102–124
2. Giese, M., Calvanese, D., Horrocks, I., Ioannidis, Y., Klappi, H., Koubarakis, M., Lenzerini, M., Möller, R., Özcep, Ö., Rodriguez Muro, M., Rosati, R., Schlatte, R., Soylu, A., Waaler, A.: Scalable End-user Access to Big Data. In Rajendra, A., ed.: Big Data Computing. Chapman and Hall/CRC (2013)

3. Catarci, T., Costabile, M.F., Levialdi, S., Batini, C.: Visual query systems for databases: A survey. Journal of Visual Languages and Computing **8**(2) (1997) 215–260
4. Shneiderman, B.: Direct Manipulation: A Step Beyond Programming Languages. Computer **16**(8) (1983) 57–69
5. Zloof, M.M.: Query-by-Example: a data base language. IBM System Journal **16**(4) (1997) 324–343
6. Erwig, M.: Xing: a visual XML query language. Journal of Visual Languages and Computing **14**(1) (2003) 5–45
7. Catarci, T., Dongilli, P., Di Mascio, T., Franconi, E., Santucci, G., Tessaris, S.: An ontology based visual tool for query formulation support. In: 16th European Conference on Artificial Intelligence (ECAI'2004). Volume 110 of Frontiers in Artificial Intelligence and Applications., IOS Press (2004) 308–312
8. Barzdins, G., Liepins, E., Veilande, M., Zviedris, M.: Ontology Enabled Graphical Database Query Tool for End-Users. In: 8th International Baltic Conference on Databases and Information Systems (DB&IS 2008). Volume 187 of Frontiers in Artificial Intelligence and Applications., IOS Press (2009) 105–116
9. Kogalovsky, M.R.: Ontology-Based Data Access Systems. Programming and Computer Software **38**(4) (2012) 167–182
10. Laney, D.: 3D Data Management: Controlling Data Volume, Velocity and Variety. Technical report, META Group (2001)
11. Madden, S.: From Databases to Big Data. IEEE Internet Computing **16**(3) (2012) 4–6
12. Lieberman, H., Paternò, F., Wulf, V., eds.: End User Development. Volume 9 of Human-Computer Interaction Series. Springer (2006)
13. Epstein, R.G.: The TableTalk Query Language. Journal of Visual Languages and Computing **2** (1991) 115–141
14. Siau, K.L., Chan, H.C., Wei, K.K.: Effects of query complexity and learning on novice user query performance with conceptual and logical database interfaces. IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans **34**(2) (2004) 276–281
15. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web – A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American **284**(5) (2001) 34–43
16. Rodriguez-Muro, M., Lubyte, L., Calvanese, D.: Realizing ontology based data access: A plug-in for Protégé. In: IEEE 24th International Conference on Data Engineering Workshop (ICDEW 2008), IEEE (2008) 353–356
17. Rodriguez-Muro, M., Calvanese, D.: Quest, a System for Ontology Based Data Access. In: OWL: Experiences and Directions Workshop 2012 (OWLED'2012). Volume 849 of CEUR Workshop Proceedings., CEUR-WS.org (2012)
18. Ruiz, F., Hilera, J.R.: Using Ontologies in Software Engineering and Technology. In Calero, C., Ruiz, F., Piattini, M., eds.: Ontologies for Software Engineering and Software Technology. Springer-Verlag (2006) 49–102
19. Spanos, D.E., Stavrou, P., Mitrou, N.: Bringing relational databases into the Semantic Web: A survey. Semantic Web **3**(2) (2012) 169–209
20. Coutaz, J., Crowley, J.L., Dobson, S., Garlan, D.: Context is key. Communications of the ACM **48**(3) (2005) 49–53
21. Marchionini, G., White, R.: Find What You Need, Understand What You Find. International Journal of Human-Computer Interaction **23**(3) (2007) 205–237
22. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data – The Story So Far. International Journal on Semantic Web and Information Systems **5**(3) (2009) 1–22

23. Tunkelang, D., Marchionini, G.: Faceted Search. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan and Claypool Publishers (2009)
24. ter Hofstede, A., Proper, H., van der Weide, T.: Query Formulation as an Information Retrieval Problem. Computer Journal **39**(4) (1996) 255–274
25. Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: Tabulator: Exploring and Analyzing linked data on the Semantic Web. In: 3rd International Semantic Web User Interaction Workshop (SWUI06). (2006)
26. Zviedris, M., Barzdins, G.: ViziQuer: A Tool to Explore and Query SPARQL Endpoints. In: 8th Extended Semantic Web Conf, Part II (ESWC'11). Volume 6644 of LNCS., Springer-Verlag (2011) 441–445
27. Popov, I.O., Schraefel, M.C., Hall, W., Shadbolt, N.: Connecting the Dots: A Multi-pivot Approach to Data Exploration. In: 10th International Semantic Web Conf., Part I (ISWC'11). Volume 7031 of LNCS., Springer-Verlag (2011) 553–568
28. Yee, K.P., Swearingen, K., Li, K., Hearst, M.: Faceted metadata for image search and browsing. In: SIGCHI Conference on Human Factors in Computing Systems (CHI'03), ACM (2003) 401–408
29. Schraefel, M.C., Wilson, M., Russell, A., Smith, D.A.: mSpace: improving information access to multimedia domains with multimodal exploratory search. Communications of the ACM **49**(4) (2006) 47–49
30. Huynh, D.F., Karger, D.R., Miller, R.C.: Exhibit: Lightweight Structured Data Publishing. In: 16th International Conference on World Wide Web (WWW '07), ACM (2007) 737–746
31. Huynh, D.F., Karger, D.R.: Parallax and Companion: Set-based Browsing for the Data Web. Available online (2009)
32. Kobilarov, G., Dickinson, I.: Humboldt: Exploring Linked Data. In: Linked Data on the Web Workshop. (2008)
33. Harth, A.: VisiNav: A system for visual search and navigation on web data. Journal of Web Semantics **8**(4) (2010) 348–354
34. Suh, B., Bederson, B.B.: OZONE: A Zoomable Interface for Navigating Ontology Information. In: Working Conference on Advanced Visual Interfaces (AVI'02), ACM (2002) 139–143
35. Heim, P., Ziegler, J.: Faceted Visual Exploration of Semantic Data. In: Second IFIP WG 13.7 conference on Human-computer interaction and visualization (HCIV'09). Volume 6431 of LNCS., Springer-Verlag (2011) 58–75
36. Soylu, A., Moedritscher, F., Wild, F., De Causmaecker, P., Desmet, P.: Mashups by orchestration and widget-based personal environments: Key challenges, solution strategies, and an application. Program: Electronic Library and Information Systems **46**(4) (2012) 383–428
37. Calvanese, D., Giese, M., Haase, P., Horrocks, I., Hubauer, T., Ioannidis, Y., Jiménez-Ruiz, E., Kharlamov, E., Kllapi, H., Klüwer, J., Koubarakis, M., Lamparter, S., Möller, R., Neuenstadt, C., Nordtveit, T., Özcep, O., Rodriguez Muro, M., Roshchin, M., Ruzzi, M., Savo, F., Schmidt, M., Soylu, A., Waaler, A., Zheleznyakov, D.: The Optique Project: Towards OBDA Systems for Industry (Short Paper). In: OWL Experiences and Directions Workshop (OWLED). (2013)
38. Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., Giannopoulou, E.: Ontology visualization methods - A survey. ACM Computing Surveys **39**(4) (2007) 10:1–10:43
39. Skjæveland, M.G., Lian, E.H., Horrocks, I.: Publishing the Norwegian Petroleum Directorate's FactPages as Semantic Web Data. To be published in the proceedings of the International Semantic Web Conference (ISWC) 2013.