Making Web-Scale Semantic Reasoning More Service-Oriented: The Large Knowledge Collider

Alexey Cheptsov[1], Zhisheng Huang[2]
[1] High Performance Computing Center Stuttgart, Germany [2] Free University of Amsterdam, the Netherlands

Abstract **– Reasoning is one of the essential application areas of the modern Semantic Web. Nowadays, the semantic reasoning algorithms are facing significant challenges when dealing with the emergence of the Internet-scale knowledge bases, comprising extremely large amounts of data. The traditional reasoning approaches have only been approved for small, closed, trustworthy, consistent, coherent and static data domains. As such, they are not well-suited to be applied in data-intensive applications aiming on the Internet scale. We introduce the Large Knowledge Collider as a platform solution that leverages the service-oriented approach to implement a new reasoning technique, capable of dealing with exploding volumes of the rapidly growing data universe, in order to be able to take advantages of the large-scale and on-demand elastic infrastructures such as high performance computing or cloud technology.**

Keywords – Semantic Web, Reasoning, Big Data, Distribution, Parallelization, Performance.

I. Introduction

The large- and internet-scale data applications are the primary challenger for the Semantic Web, and in particular for reasoning algorithms, used for processing exploding volumes of data, exposed currently on the Web. Reasoning is the process of making implicit logical inferences from the explicit set of facts or statements, which constitute the core of any knowledge base. The key problem for most of the modern reasoning engines such as Jena [1] or Pellet [2] is that they can not efficiently be applied for the real-life data sets that consist of tens, sometimes of hundreds of billions of triples (a unit of the semantically annotated information), which can correspond to several petabytes of digital information. Whereas modern advances in the Supercomputing domain allow this limitation to be overcome, the reasoning algorithms and logic need to be adapted to the demands of rapidly growing data universe, in order to be able to take advantages of the large-scale and on-demand infrastructures such as high performance computing or cloud technology. On the other hand, the algorithmic principals of the reasoning engines need to be reconsidered as well in order to allow for very large volumes of data. Service-oriented architectures (SOA) can greatly contribute to this goal, acting as the main enabler of the newly proposed reasoning techniques such as incomplete reasoning [3]. This paper focuses on a service-oriented solution for constructing Semantic Web applications of a new generation, ensuring the drastic increase of the scalability for the existing reasoning applications, as elaborated by the Large Knowledge Collider (LarKC)[1] EU project.

The paper is organized as follows. In Section II, we collect our consideration towards enabling the large-scale reasoning. In Section III, we discuss LarKC – a service-oriented platform for development of fundamentally new reasoning application, with much higher scalability barriers as by the existing solutions. In Section IV, we introduce some successful applications implemented with LarKC, such as Bottari – the Semantic Challenge winner in 2011. In Section V, we discuss our conclusions and highlight the directions for future work in highly scalable semantic reasoning.

II. Towards Semantic Reasoning on the Web Scale

A. From Web to the Semantic Web

The Web as it is seen by the users "behind the browser" has traditionally been one of the most successful examples of the SOA realization. The possibility to transform the application's business logic into a set of the linked services supplied with the transparent access to those services over standardized protocols such as HTTP was a key asset for tremendous wide-spread of the Internet worldwide. However the possibility to organize business relationship between the data located on several hosts had been extremely poor. The research seeking for a concept of applying a data model on the Web scale resulted in the Semantic Web – the later advance of the Web, which offers a possibility to extend the Web-enabled data with the annotation of their semantics, thus making the context in which the data is used meaningful for the applications [4]. Nowadays, there are several existing well-established standards for annotation of data web-wide, such as for example Resource Description Framework (RDF)[2] schema.

The practical value of the Semantic Web is that it enables development of applications that can handle complex human queries based not only on the value of the analyzed data, but also on its meaning. Promotion of such platforms as (Friends-of-a-Friend) FOAF[3] at the early stages of the Semantic Web has forced a lot of data providers to actively expose and interlink their data on the Web, which resulted in many problem-oriented data

[1] http://www.larkc.eu/
[2] http://www.w3.org/RDF/
[3] http://www.foaf-project.org/

repositories, as for example Linked Life Data (LLD)[4], which is a collection of the data for biomedical domain; alone the LLD dataset comprises over one billion web resources presented in RDF. On the other hand, social networks like Twitter or Facebook encourage people to upload there personal data as well, thus drastically increasing the weight of the digital information on the Web.

B. Semantic Reasoning

Thanks to the ability to offer the structured data as the Web content, the Semantic Web has become de-facto an indispensable aspect of the human's everyday life. The application areas of the modern Semantic Web spawn a wide range of domains, from social networks to large-scale Smart Cities projects in the context of the future internet [5]. However, data processing in such applications goes far beyond a simple maintenance of the collection of facts; based on the explicit information, collected in datasets, and simple rule sets, describing the possible relations, the implicit statements and facts can be acquired from those datasets. For example, supposed that bulldogs are dogs, and cats hate dogs, cats must also hate bulldogs, which is however not explicitly stated but rather inferred from the content.

Many data collections as well as application built on top of them allow for rule-based inferencing to obtain new, more important facts. The process of inferring logical consequences from a set of asserted facts, specified by using some kinds of logic description languages (e.g., RDF/RDFS and OWL[5]), is in focus of semantic reasoning. The goal is to provide a technical way to determine when inference processes is valid, i.e., when it preserves truth. This is achieved by the procedure which starts from a set of assertions that are regarded as true in a semantic model and derives whether a new model contains provably true assertions.

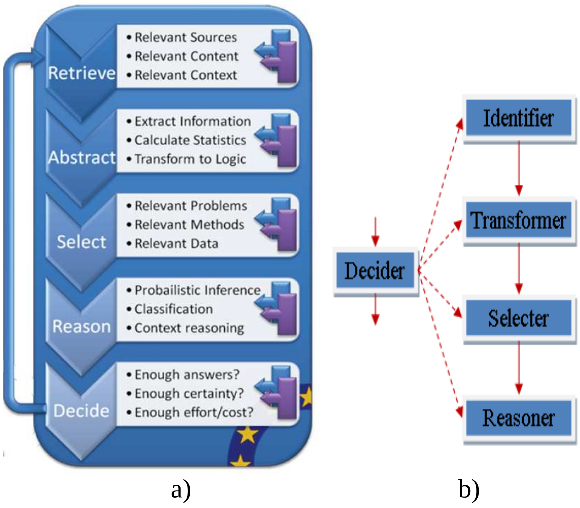
C. Big Data Challenge and new Reasoning Approaches

The latest research on the Internet-scale Knowledge Base Technologies, combined with the proliferation of SOA infrastructures and cloud computing, has created a new wave of data-intensive computing applications, and posed several challenges to the Semantic Web community. As a reaction on these challenges, a variety of reasoning methods have been suggested for the efficient processing and exploitation of the semantically annotated data. However, most of those methods have only been approved for small, closed, trustworthy, consistent, coherent and static domains, such as synthetic LUBM [6] sets. Still, there is a deep mismatch between the requirements on the real-time reasoning on the Web scale and the existing efficient reasoning algorithms over the restricted subsets.

Whereas unlocking the full value of the scientific data has been seen as a strategic objective in the majority of ICT- related scientific activities in EU, USA, and Asia [7], the “Big Data” problem has been recognized as the primary challenger in semantic reasoning [8][9]. Indeed, the recent years have seen a tremendous increase of the structured data on the Web with scientific, public, and even government sectors involved. According to one of the recent IDC reports [10], the size of the digital data universe has grown from about 800.000 Terabytes in 2009 to 1.2 Zettabytes in 2010, i.e. an increase of 62%. Even more tremendous growth should be expected in the future (up to several tens of Zettabytes already in 2012, according to the same IDC report [10]).

The “big data” problem makes the conventional data processing techniques, also including the traditional semantic reasoning, substantially inefficient when applied for the large-scale data sets. On the other hand, the heterogeneous and streaming nature of data, e.g. implying structure complexity [11], or dimensionality and size [12], makes big data intractable on the conventional computing resource [13]. The problem becomes even worse when data are inconsistent (there is no any semantic model to interpret) or incoherent (contains some unclassifiable concepts) [14].

The broad availability of data coupled with increasing capabilities and decreasing costs of both computing and storage facilities has led the semantic reasoning community to rethink the approaches for large-scale inferencing [15]. Data-intensive reasoning requires a fundamentally different set of principles than the traditional mainstream Semantic Web offers. Some of the approaches allow for going far beyond the traditional notion of absolute correctness and completeness in reasoning as assumed by the standard techniques. An outstanding approach here is interleaving the reasoning and selection [16]. The main idea of the interleaving approach (see Figure 1a) is to introduce a selection phase so that the reasoning processing can focus on a limited (but meaningful) part of the data, i.e. perform incomplete reasoning.



a) b)

Figure 1. Incomplete reasoning, the overall schema (a) and the service-oriented vision (b)

[4] http://linkedlifedata.com/

[5] http://www.w3.org/TR/owl-ref/

D. SOA Aspect in Semantic Reasoning

As we have discussed before, the standard reasoning methods are not valid in the existing configurations of the Semantic Web. Some approaches, such as incomplete reasoning, offer a promising vision how a reasoning application can overcome the “big data” limitation, e.g. by interleaving the selection with the reasoning in a single “workflow”, as shown in Figure 1a. However the need of combining several techniques within a single application introduces new challenges, for example related to ensuring the proper collaboration of team of experts working on a concrete part of the workflow, either it is identification, selection, or reasoning. Another challenge might be the adoption of the already available solutions and reusing them in the newly developed applications, as for example applying selection to the JENA reasoner [2], whose original software design doesn’t allow for such functionality. The SOA approach can help eliminate many of the drawbacks on the way towards creating new, service-based reasoning applications. Supposed that each of the construction blocks shown in Figure 1a is a service, with standard API that ensures easy interoperability with the other similar services, quite a complex application can be developed by a simple combination of those services in a common workflow (see Figure 1b).

Although the workflow concept is not new for the semantic reasoning [17][18], there was quite a big gap in realizing the single steps of the reasoning algorithms (Figure 1b) as a service. This was due to many reasons, among them complexity of the data dependency management, ensuring interoperability of the services, heterogeneity of the service’s functionality. Realizing a system where a massive number of parties can expose and consume services via advanced Web technology was also a research highlight for Semantic Web. An example of very successful research on offering a part of the semantic reasoning logic as a service is the SOA4ALL[6] project, whose main goal was to study the service abilities of development platforms capable of offering semantic services. Several useful services wrapping such successful reasoning engines as IRIS [19] and several others had been developed in the frame of this project. Nevertheless, the availability of such services is only an intermediate step towards offering reasoning as a service, as a lot of efforts were required to provide interoperability of those services in the context of a common application. Among others, a common platform is needed that would allow the user to seamlessly integrate the service by annotating their dependencies, manage the data dependencies intelligently, being able to specify parts of the execution that should be executed remotely, etc.

An outstanding effort to develop such a platform was performed in the LarKC (Large Knowledge Collider) [20] project. In the following sections, we discuss the main ideas, solutions, and outcomes of this project.

III. Large Knowledge Collider – Making the Semantic Reasoning More Service Oriented

A. Objectives and Concepts

In order to facilitate the technology for creation of trend-new applications for large-scale reasoning, several leading Semantic Web research organizations and technological companies have joined their efforts around the project of the Large Knowledge Collider (LarKC), supported by the European Commission. The mission of the project was to set up a distributed reasoning infrastructure for the Semantic Web community, which should enable application of reasoning far beyond the currently recognized scalability limitations [21], by implementing the interleaving reasoning approach. The current and future Web applications that deal with “big data” are in focus of LarKC.

To realize this mission, LarKC has created an infrastructure that allows construction of plug-in-based reasoning applications, following the interleaving approach, facilitated by incorporating interdisciplinary techniques such as inductive, deductive, incomplete reasoning, in combination with the methods from other knowledge representation domains such as information retrieval, machine learning, cognitive and social psychology. The core of the infrastructure is a platform – a software framework that facilitates design, testing, and exploitation of new reasoning techniques for development of large-scale applications. The platform does this by providing means for creating very lightweight, portable and unified services for data sharing, accessing, transformation, aggregation, and inferencing, as well as means for building Semantic Web applications on top of those services. The efficiency of the services is ensured by providing a transparent access to the underlying resource layer, served by the platform, involving high performance computing, storage, and cloud resources, and in the other way around, providing performance analysis and monitoring information back to the user. The platform is built in a distributed, modular, and open source fashion. Moreover, the platform offers means for building and running applications across those plug-ins, provide them a persistent data layer for storing data, facilitate parallel execution of large-scale data operations on distributed and high-performance resources [22].

The two main issues solved by LarKC are development of a reasoning application combining solutions and techniques coming from diverse domains of the Semantic Web and Computer Science disciplines (e.g. High Performance Computing), and ensuring the requested QoS requirements, in particular by targeting the modern e-Infrastructures such as grid and cloud environments.

Guided by the preliminary goal to facilitate incomplete reasoning, LarKC has evolved in a unique platform, which can be used for development of a wide range of semantic web applications, following the SOA paradigm. The sections below discuss the main functional properties and features of the LarKC platform.

[6] http://www.soa4all.eu/

B. Architecture Overview

The LarKC's design has been guided by the primarily goal to build a scalable platform for distributed high performance reasoning. Figure 2 shows a conceptual view of the LarKC platform's architecture and the proposed development life-cycle. The architecture was designed to holistically cover the needs of the three main categories of users – semantic service (plug-in) developers, application (workflow) designers, and end-users internet-wide. The platform's design ensures a trade-off between the flexibility and the performance of applications in order to achieve a good balance between the generality and the usability of the platform by each of the categories of users.

Below we introduce some of the key concepts of the LarKC architecture and discuss the most important platform's services and tools for them.

1) Plug-ins

Plug-ins are standalone services implementing some specific parts of the reasoning logic as discussed previously, whether it is selection, identification, transformation, or reasoning algorithm, see more at [21]. In fact, plug-ins can implement much broader functionality as foreseen by the incomplete reasoning schema (Figure 1), hence enabling the LarKC platform to target much wider Semantic Web user community as originally targeted, e.g. for machine learning or knowledge extraction. The services are referred as plug-ins because of their flexibility and ability to be easily integrated, i.e. plugged into a common workflow and hence constitute a reasoning application, such as the ones in Figure 3a and Figure 3b. To ensure the interoperability of the plug-ins in the workflows, each plug-in should implement a special plug-in API, based on the annotation language [23]. Most essentially, the API defines the RDF schema (set of statements in the RDF format) taken as input and produced as output by each of the plug-ins. The plug-in development is facilitated by a number of special wizards, such as Eclipse IDE wizard or Maven archetype for rapid plug-in prototyping. The ready-to-use plug-ins are uploaded and published on the marketplace – a special web-enabled service offering a centralized, web-enabled repository store for the plug-ins[7].

2) Workflows

The workflow designers get access to the Marketplace in order to construct a workflow from the available plug-ins, combined to solve a certain task. In terms of LarKC, workflow is a reasoning application that is constructed of the (previously developed and uploaded on the Marketplace) plug-ins. The workflow's topology is characterised by the plug-ins included in the workflow as well as the data- and control flow connections between these plug-ins.

The complexity of the workflow's topology is determined by the number of included plug-ins, data connections between the plug-ins (also including multiple splits and joins such as in Figure 3a or several end-points such as in Figure 3b), and control flow events (such as instantiating, starting, stopping, and terminating single plug-ins or even workflow branches comprising several plug-ins). Same as for plug-ins, the input and output of the workflow is presented in RDF, which however can cause compatibility issues with the user's GUI, which are not obviously based on an RDF-compliant representation. To confirm the internal (RDF) dataflow representation with the external (user-defined) one, the LarKC architecture foresees special end-points, which are the adapters facilitating the workflow usage in the tools outside of the LarKC platform. Some typical examples of end-points, already provided by LarKC, are e.g. SPARQL end-point (SPARQL query as input and set of RDF statements as output) and HTML end-point (HTTP request without any parameters as input and HTML page as output).

For the specification of the workflow configuration, a special RDF schema was elaborated for LarKC, aiming at simplification of the annotation efforts for the workflow designers. Figure 3c shows a simple example of the LarKC workflow annotation. Creation of the workflow specification can greatly be simplified by using upper-level graphical tools, e.g. Workflow Designer (Figure 3d) that offers a GUI for visual workflow construction (Figure 3d). The elaborated schema makes specification of the additional features such as remote plug-in execution extremely simple and transparent for the users and can be used for tuning the front-end graphical interfaces of the applications to adapt them to the user needs.

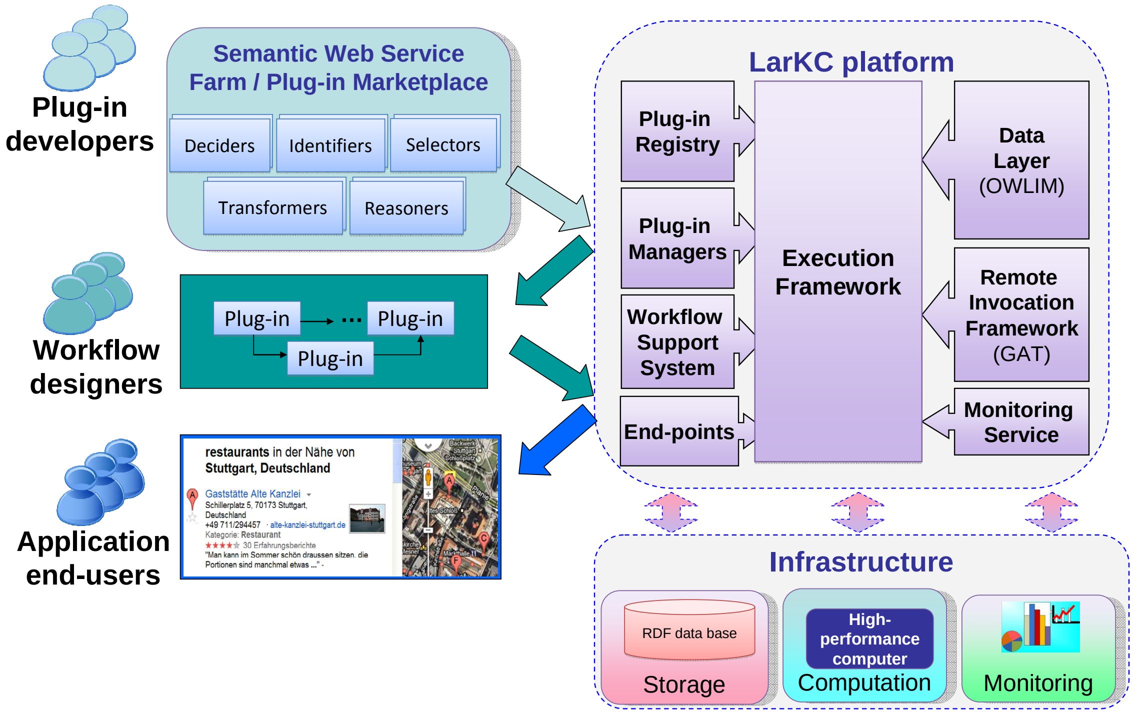
3) Applications

Workflows are already standalone applications that can be submitted to the platform and executed by means of such tools as Workflow Designer discussed above. Nevertheless, workflows can also be wrapped into much more powerful user interfaces, adapted to the needs of the targeted end-user communities, e.g. Urban Computing [24], and using LarKC as a back-end engine. The SO approach makes possible hiding the complexity of the LarKC platform, by enabling its whole power to the end-users through such interfaces. We discuss some of the most successful examples of the LarKC applications in Section 4.

4) Platform services

All above-described activities related to plug-in creation, workflow design, and application development are facilitated by an extensive set of the platform services, as shown in Figure 2.

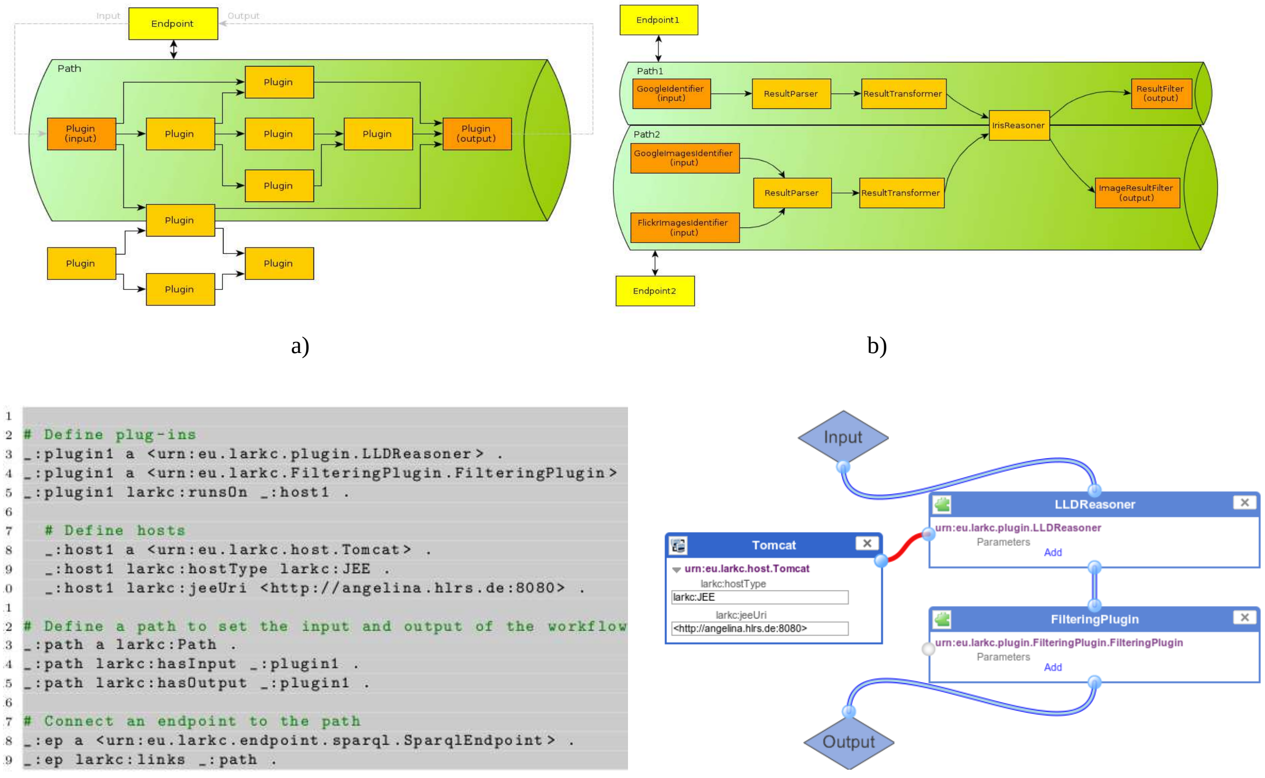
[7] Visit the LarKC Plug-in Marketplace at http://www.larkc.eu/plug-in-marketplace/



restaurants in der Nähe von
Stuttgart, Deutschland

Gaststätte Alte Kanzlei
Schillerplatz 5, 70173 Stuttgart,
Deutschland
+49 711/294457 · alte-kanzlei-stuttgart.de
Kategorie: Restaurant
★★★★☆ 30 Erfahrungsberichte
"Man kann im Sommer schön draussen sitzen. die Portionen sind manchmal etwas ..." -

Figure 2. Architecture of LarKC



a)

b)

```

# Define plug-ins
_:plugin1 a <urn:eu.larkc.plugin.LLDReasoner> .
_:plugin1 a <urn:eu.larkc.FilteringPlugin.FilteringPlugin>
_:plugin1 larkc:runsOn _:host1 .

  # Define hosts
  _:host1 a <urn:eu.larkc.host.Tomcat> .
  _:host1 larkc:hostType larkc:JEE .
  _:host1 larkc:jeeUri <http://angelina.hlrs.de:8080> .

# Define a path to set the input and output of the workflow
_:path a larkc:Path .
_:path larkc:hasInput _:plugin1 .
_:path larkc:hasOutput _:plugin1 .

# Connect an endpoint to the path
_:ep a <urn:eu.larkc.endpoint.sparql.SparqlEndpoint> .
_:ep larkc:links _:path .
```

urn:eu.larkc.plugin.LLDReasoner

urn:eu.larkc.host.Tomcat
larkc:hostType: larkc:JEE
larkc:jeeUri: <http://angelina.hlrs.de:8080>

urn:eu.larkc.plugin.FilteringPlugin.FilteringPlugin

c)

d)

Figure 3. LarKC workflows: a) workflow with non-trivial branched dataflow (containing multiple splits/joins), b) workflow with multiple end-points, c) example of RDF schema for workflow annotation, d) Workflow Designer GUI

Execution Framework is the “control centre" of the LarKC Platform. It is responsible for the services related to the plug-in (Plug-in Registry, Plug-in Managers), workflow (Workflow Support System, Workflow Designer), and application (End-points) support. It also provides a set of fundamental services indispensable for the organization of the data management (Data Layer), distributed execution (Remote Invocation Framework), and performance monitoring (Monitoring Services).

Plug-in Registry is a service that allows the platform to load the plug-ins as well as the external libraries needed by them to the internal plug-in knowledge base, where the plug-ins can be instantiated from when constructing and executing the workflow.

Plug-in Managers facilitate the integration of plug-ins within a workflow and the management of their execution. The managers allow a plug-in to be executed either locally or in a distributed fashion. The latter is facilitated by the remote invocation framework that is based on Grid Access Toolkit (GAT) [31] and support several categories of host, also in view of the Cloud paradigm.

Data Layer is a simplified realization of OWLIM [25] – a high-performance RDF data base that supports the plug-ins and applications with respect to storage, retrieval (including streaming), and lightweight inference on top of large volumes of RDF data. In particular, Data Layer is used for storing the data passed between the plug-ins, so that only a reference is passed; this reveals the plug-ins from the need of handling the RDF data and hence make them applicable for large data volumes stored in the Data Layer.

5) Infrastructure

With regard to the infrastructure layer, LarKC acts as a middleware that facilitates the successful application deployment and execution on the available resource base. The LarKC platform offers the plug-ins an abstraction layer, facilitated by the plug-in API, that allows applications based on those plug-ins to abstract from the specific resource layer properties, such as operating system, number of compute cores (for shared memory) or nodes (for distributed memory parallel systems), etc., hence making the deployment process as transparent as possible. This is facilitated by several know-how solutions for distributed execution, parallelization, and monitoring.

Distributed execution is the key feature of the LarKC execution model. It allows a plug-in to be executed on the resource that is remote with regard to the one where the platform is running [32]. Standard cases where the applications can benefit from the distributed execution include but not restrict shipping the execution closer to the data being processed, running a part of the workflow on the resource that ensures better performance but forbids the full deployment of the LarKC platform, e.g. production high performance supercomputers, etc.

Parallel execution is another added value of the LarKC platform to the applications. Parallelization is a key technology for performance-critical applications to meet the QoS requirements, also including the performance characteristics. The platform provides the needed abstractions (on the plug-in level) for implementation of both data- and instruction-level workload decomposition, e.g. for splitting up the big dataset into subsets with further parallel processing each subset by plug-in instances running on separated nodes, thus avoiding the possible competition for the hardware resource [13]. This is facilitated through the tight integration of such popular parallelization strategies as multithreading, message-passing (MPI), or MapReduce.

Monitoring is the essential feature of the LarKC platform that allows plug-ins to be (automatically) instrumented to produce some important metrics about their execution, e.g. execution time (performance), or size of the processed data (throughput). Those characteristics can be collected from different execution configurations and used for identifying possible bottlenecks or just collecting some interesting for the user statistics [26]. The visualization tools are provided by the platform as well, so a very little efforts is needed to get the complete trace of the application run.

IV. Success Stories and Application Examples

LarKC is the technology that not only enables the large-scale reasoning approach for the already existing applications, but also facilitates their rapid prototyping with low initial investments, leveraging the SOA approach through the solutions discussed in Section III. Furthermore, LarKC delivers a complete eco-system where the researches from very different domains can team up in order to develop new challenging mashup-applications, hence having a dramatic impact on a lot of problem domain. Below we describe some of the most prominent pilot applications developed with LarKC in 2010-2011.

1) Bottari

BOTTARI [27] is a location-based mobile application that leverages a place of interest recommendation system to support people who find themselves in the new place, which they are not familiar with. The application’s front-end is implemented at Android tablets, whereas the back-end is served by LarKC. BOTTARI is collecting relevant information from social media networks such as Twitter and blog posts, elaborates it and provides contextualized suggestions. At the current stage, the application was implemented for one of the most popular touristic districts in Seoul, South Korea. The recommendations given by BOTTARI include places of interest nearby the current location of the user, reputation ranking of the suggested places according to the other users’ feedback, identification of the most interesting place fitting well the user’s profile. To the main innovations of BOTTARI can be referred offering a location-based service through a simple and intuitive interface, advanced semantic features, and hiding the complexity of reasoning from the end-user. BOTTARI become the winner of the International Semantic Web challenge 2011.

2) *WebPIE*

WebPIE (Web-scale Parallel Inference Engine) [28] is a MapReduce-based parallel distributed RDFS/OWL inference engine. Being implemented as a LarKC plug-in, WebPIE can be used for materialization of an RDF graph expressed in the OWL Horst semantics, which is required by a lot of semantic reasoning workflows. The workflows that use WebPIE can take advantages of the distributed and parallel reasoning, facilitated by the underlying MapReduce implementation with Hadoop. Thanks to the parallel implementation, WebPIE vastly outperforms all the existing inference engines when comparing supported language expressivity, maximum data size and inference speed (according to the benchmarks in [29]). In LarKC, WebPIE can easily be integrated in any forward chaining reasoning workflow and thus improve its scalability. The distributed execution framework takes care of the execution of the WebPIE reasoner on a machine that can take full advantages of the parallel realization, e.g. a cluster of workstations or a parallel supercomputer. The WebPIE research won the first scalability prize at the IEEE Scale Challenge in 2010.

3) *GWAS*

Genome-wide association study (GWAS) is a research domain aiming to identify common genetic factors that influence health and disease apparition. GWAS use bio-probes (gene markers) to look for higher levels of association between genes in a diseased subject as opposed to controls. The large numbers of markers mean that huge numbers of samples are needed to achieve sufficient statistical power. Semantic Web helps the GWAS researchers apply common statistical models to raw experimental data to find the relevance of each marker, and then rank them in order of relevance to the disease. Only the genes that are close to the top few markers are then studied in more depth by conventional techniques, to narrow the problem and achieve better results. This last bit is expensive, and improving rankings could improve both the efficiency and the economics of the technique. The WHO’s cancer research unit, IARC, has chosen LarKC as the technology to combine prior knowledge about a gene with experimental data, thus improving statistical power [30]. The modular nature of LarKC plug-ins allowed for combination of those techniques with the modern advances of the Statistical Semantics as random indexing, term frequency inverse document frequency, or term expansion using UMLS. This allowed the researchers to scale knowledge discovery across the large amounts of biomedical knowledge now encoded in the data- and bibli-ome, and to apply it to the millions of data points in a typical GWAS.

V. Conclusions and Future Directions

We proposed a solution for the problem statement done at the beginning of the paper – “Where SOA meets the Semantic Reasoning”, which is the Large Knowledge Collider (LarKC). LarKC is very promising platform for creation of new-generation semantic reasoning applications. The LarKC’s main value is twofold. On the one hand, it enables a new approach for large-scale reasoning based on the technique for interleaving the identification, the selection, and the reasoning phases. On the other hand, through over the project’s life time (2008-2011), LarKC has evolved in an outstanding, service-oriented platform for creating very flexible but extremely powerful applications, based on the plug-in’s realization concept. The LarKC plug-in marketplace has already comprised several tens of freely available plug-ins, which implement new know-how solutions or wrap existing software components to offer their functionality to a much wider range of applications as even originally envisioned by their developers. Moreover, LarKC offers several additional features to improve the performance and scalability of the applications, facilitated through the parallelization, distributed execution, and monitoring platform. LarKC is an open source development, which encourages collaborative application development for Semantic Web. Despite being quite a young solution, LarKC has already established itself as a very promising technology in the Semantic Web world. Some evidence of its value was a series of Europe- and world-wide Semantic Web challenges won by the LarKC applications. It is important to note that the creation of LarKC applications, including the ones discussed in the paper, was also possible and without LarKC, but would have required much more (in order of magnitude) development efforts and financial investments.

We believe that the availability of such platform as LarKC will make a lot of developers to rethink their current approaches for semantic reasoning towards much wider adoption of the service-oriented paradigm. Another added value of LarKC is a number of very promising future researches that will be done as LarKC’s spin-offs, including streaming data support, decision making in large systems, and many others. Among others, a lot of challenges are introduced by Smart Cities applications, which provide static data pools of Petabyte size range as well as deliver Terabytes of new dynamically-acquired data on the daily basis. We would be interested to apply LarKC to such challenging application scenarios and evaluate its ability to meet the real-time requirements of such large-scale systems.

Acknowledgmentes

We would like to thank to the consortium of the LarKC project (co-funded by the European Commission under the grant agreement ICT-FP7-215535) for the work done and materials provided.

REFERENCES

[1] E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, Y. Katz, “Pellet: a practical owl-dl reasoner”, *Journal of Web Semantics*, http://www.mindswap.org/papers/PelletJWS.pdf

[2] P. McCarthy, “Introduction to Jena”, *IBM developerWorks*, http://www.ibm.com/developerworks/xml/library/j-jena/

[3] D. Fensel, F. van Harmelen, “Unifying Reasoning and Search to Web Scale”, *IEEE Internet Computing*. 11(2), 2007, 96-95.

[4] Broekstra, J.; Klein, M.; Decker, S.; Fensel, D.; van Harmelen, F.; Horrocks, I. (2001): Enabling knowledge representation on the Web by extending RDF schema. In: *Proceedings of the 10th international conference on World Wide Web (WWW '01)*. 467-478, ACM.

[5] M. Donovang-Kuhlisch, “Smart City Process Support and Applications as a Service – from the Future Internet”, *Future Internet Assembly*, 2010, http://fi-ghent.fi-week.eu/files/2010/12/1430-Margarete-Donovang-Kuhlisch.pdf

[6] Y. Guo, Z. Pan, and J. Heflin, “LUBM: A Benchmark for OWL Knowledge Base Systems”, *Web Semantics*, 3(2) July 2005. pp.158-182

[7] High Level Expert EU Group, “Riding the wave - How Europe can gain from the rising tide of scientific data”, *Final report*, October 2010, http://ec.europa.eu/information_society/newsroom/cf/document.cfm?action=display&doc_id=707

[8] B. Thompson, M. Personick, “Large-scale mashups using RDF and bigdata”, *Semantic Technology Conference*, 2009.

[9] U. Hustadt, B. Motik, and U. Sattler, “Data Complexity of Reasoning in Very Expressive Description Logics”, *In Proc. IJCAI* 2005, pages 466–471, Edinburgh, UK, July 30–August 5 2005. Morgan Kaufmann Publishers.

[10] J. McKendrick, “Size of the data universe: 1.2 zettabytes and growing fast”, *ZDNet*.

[11] E. Della Valle, S. Ceri, F. van Harmelen, and D. Fensel, “It’s a streaming world! Rreasoning upon rapidly changing information”, *IEEE Intelligent Systems*, 24(6):83–89, 2009

[12] D. Fensel, F. van Harmelen, “Unifying Reasoning and Search to Web Scale”, *IEEE Internet Computing*. 11(2), 96-95

[13] A. Cheptsov, M. Assel, “Towards High Performance Semantic Web – Experience of the LarKC Project”, *inSiDE - Journal of Innovatives Supercomputing in Deutschland*, vol. 9 No. 1, Spring 2011.

[14] Z. Huang, F. van Harmelen, A. Teije, A., “Reasoning with inconsistent ontologies”, *In: Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI'05*, pp. 454-459, 2005.

[15] E. Bozsak, M. Ehrig, S. Handschuh, A. Hotho, A., Maedche, B. Motik, D. Oberle, C. Schmitz, S. Staab, L. Stojanovic, N. Stojanovic, R. Studer, G. Stumme, Y. Sure, J. Tane, R. Volz, V. Zacharias, “KAON - Towards a Large Scale Semantic Web”. In *Tjoa, AM., Quirchmayr, G., Bauknecht K. (eds.) Proceedings of the Third international Conference on E-Commerce and Web Technologies.* LNCS, 2455, 304-313 Springer.

[16] Z. Huang, “Interleaving Reasoning and Selection with Semantic Data”, *Proceedings of the 4th International Workshop on Ontology Dynamics (IWOD-10), ISWC2010 Workshop*.

[17] E. Deelman, D. Gannon, M. Shields, I. Taylor, “Workflows and e-Science: An overview of workflow system features and capabilities”, *Future Generation Computer Systems*, 25(5), 2009.

[18] Y. Gil, V. Ratnakar, C. Fritz, “Assisting Scientists with Complex Data Analysis Tasks through Semantic Workflows”, In *Proceedings of the AAAI Fall Symposium on Proactive Assistant Agents* , Arlington, VA.

[19] “IRIS - Integrated Rule Inference System - API and User Guide”, http://iris-reasoner.org/pages/user_guide.pdf

[20] D. Fensel, F. van Harmelen, B. Andersson, P. Brennan, H. Cunningham, E. Della Valle, F. Fischer, Z. Huang, A. Kiryakov, T. Lee, L. Schooler, V. Tresp, S. Wesner, M. Witbrock, N. Zhong, “Towards LarKC: A Platform for Web-Scale Reasoning”, In: *Proceedings of the 2008 IEEE international Conference on Semantic Computing ICSC.* 524-529, IEEE Computer Society.

[21] M. Assel, A. Cheptsov, G. Gallizo, I. Celino, D. Dell'Aglio, L. Bradeško, M. Witbrock, E. Della Valle, “Large knowledge collider: a service-oriented platform for large-scale semantic reasoning”, *Proceedings of the International Conference on Web Intelligence, Mining and Semantics (WIMS'2011)*.

[22] M. Assel, A. Cheptsov, G. Gallizo, K. Benkert, A. Tenschert, “Applying High Performance Computing Techniques for Advanced Semantic Reasoning”, In: *eChallenges e-2010 Conference Proceedings*. Paul Cunningham and Miriam Cunningham (Eds). IIMC International Information Management Corporation, 2010

[23] D. Roman, B. Bishop, I. Toma, G. Gallizo, and B. Fortuna, "LarKC Plug-in Annotation Language," *in Proceedings of The First International Conferences on Advanced Service Computing – Service Computation 2009*, 2009.

[24] E. Della Valle, I. Celino, and D. Dell’Aglio, "The Experience of Realizing a Semantic Web Urban Computing Application", *T. GIS*, vol. 14, iss. 2, pp. 163-181, 2010

[25] A. Kiryakov, D. Ognyanoff, D. Manov, “OWLIM – a Pragmatic Semantic Repository for OWL”, In *Proc. of Int. Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2005), WISE 2005*, 20 Nov, New York City, USA.

[26] I. Toma, M. Chezan, R. Brehar, S. Nedevschi, and D. Fensel, "SIM, a Semantic Instrumentation and Monitoring solution for Large Scale Reasoning Systems," in *Proceedings of the 5th IEEE International Conference on Semantic Computing 2011 (ICSC2011)*, Stanford University, Palo Alto, CA, USA, 2011.

[27] I. Celino, D. Dell'Aglio, E. Della Valle, Y. Huang, T. Lee, S. Kim, V. Tresp, “Towards BOTTARI: Using Stream Reasoning to Make Sense of Location-Based Micro-Posts”. In *Garcia-Castro, R., et al., eds.: ESWC 2011 Workshops*, LNCS 7117, Springer, Heidelberg (2011) 80-87

[28] Urbani J., Kotoulas, S., Maaseen J., van Harmelen, F. & Bal, H. (2010), OWL reasoning with WebPIE: calculating the closure of 100 billion triples, In Proceedings of the ESWC '10.

[29] J. Urbani, S. Kotoulas, J. Maaseen, N. Drost, F. Seinstra, F. van Harmelen, “WebPIE: a Web-scale Parallel Inference Engine”, *Submission to the SCALE competition at CCGrid '10*.

[30] M. Johansson, Y. Li, J. Wakefield, M. A. Greenwood, T. Heitz, I. Roberts, H. Cunningham, P. Brennan, A. Roberts, and J. Mckay, “Using Prior Information Attained From The Literature To Improve Ranking In Genome-Wide Association Studies”, 2009.

[31] R. van Nieuwpoort, T. Kielmann, and H. Bal, “User-friendly and reliable grid computing based on imperfect middleware”. In *Proceedings of the ACM/IEEE Conference on Supercomputing (SC'07)*, nov 2007.

[32] J. Urbani, S. Kotoulas, E. Oren, F. van Harmelen, “Scalable Distributed Reasoning Using MapReduce”, In: *Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) The Semantic Web - ISWC 2009*, LNCS, vol. 5823, pp. 634--649, Springer (2009)