Viewing the Web as a Distributed Knowledge Base

Serge Abiteboul, Émilien Antoine, Julia Stoyanovich

▶ **To cite this version:**

Serge Abiteboul, Émilien Antoine, Julia Stoyanovich. Viewing the Web as a Distributed Knowledge Base. International Conference on Data Engineering, 2012, Washington DC, United States. hal-00703210

HAL Id: hal-00703210
https://inria.hal.science/hal-00703210

Submitted on 1 Jun 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Viewing the Web as a Distributed Knowledge Base

Serge Abiteboul [1,2], Emilien Antoine [2] and Julia Stoyanovich [3]

[1]*Collège de France, Paris, France* [2]*INRIA Saclay & ENS Cachan, France* [3]*University of Pennsylvania, USA*

***Abstract*— This papers addresses the challenges faced by everyday Web users, who interact with inherently heterogeneous and distributed information. Managing such data is currently beyond the skills of casual users. We describe ongoing work that has as its goal the development of foundations for declarative distributed data management. In this approach, we see the Web as a knowledge base consisting of distributed logical facts and rules. Our objective is to enable automated reasoning over this knowledge base, ultimately improving the quality of service and of data. For this, we use Webdamlog, a Datalog-style language with rule delegation. We outline ongoing efforts on the WebdamExchange platform that combines Webdamlog evaluation with communication and security protocols.**

I. Introduction

Information of interest may be found on the Web in a variety of forms, in many systems, and with different access protocols. A typical user may have information on many devices (smartphone, laptop, TV box, etc.), many systems (mailers, blogs, Web sites, etc.), many social networks (Facebook, Picasa, etc.). This same user may have access to more information from family, friends, associations, companies, and organizations. Today, the control and management of the diversity of data and tasks in this setting are beyond the skills of casual users. Facing similar issues, companies see the cost of managing and integrating information skyrocketing.

We are interested here in the management of such data. Our focus is not on harvesting all the data of a particular user or a group of users and then managing it in a centralized manner. Instead, we are concerned with the management of Web data *in place* in a distributed manner, with a possibly large number of autonomous, heterogeneous systems collaborating to support certain tasks. More precisely, we are concerned with the foundations of such data management.

Centralized data management has matured with relational database systems, enabled by the combination and cooperation of a very active research community and a very successful industry. The success of the field rests on a solid formal foundation, which combines existing tools, e.g., first-order logic for specifying queries and dependencies, with others that were developed from scratch, e.g., query optimization or concurrency control. As a result, centralized data management systems are now very reliable and the corresponding science is well-developed.

For distributed data on the Web, the foundations of relational databases do not suffice, for the following reasons.

- Trees. The exchange standard for the Web is based on data trees (HTML, XML, JSON) and not on relations. This is already a very active area of research, and we will not focus on it here.
- Distribution. We are interested in enabling collaboration of autonomous systems and users. Our goal is to support distributed processing, allowing each peer/user to maintain better control of their data, and leveraging the available computational resources. Foundations for distributed data management are still insufficiently developed, and we focus on them in our research.
- Uncertainty. A critical dimension of the problem is the imprecise, uncertain, and noisy nature of data on the Web. Additionally, the actors in a task may have imprecise or conflicting opinions, and so modeling their collaboration calls for representing trust and probabilities. This is an important direction of research that we will only marginally address here.

Our thesis is that managing the richness and diversity of user-centric data residing on the Web can be tamed using a holistic approach based on a *distributed knowledge base*. Our approach, outlined in this paper, is to represent all Web information as *logical facts*, and Web data management tasks as *logical rules*. The automatic reasoning by the inference engine, operating over the Web knowledge base, will greatly benefit a variety of complex data management tasks that currently require intense work and deep expertise.

The paper is organized as follows. We argue for our point of view of the Web as a distributed knowledge base in Section II. We give an overview of Webdamlog, a novel Datalog-style rule-based language, in Section III. In Section IV, we describe our implementation of WebdamExchange, a system that combines Webdamlog evaluation with access control and communication protocols. We outline future directions and conclude in Section V.

This work is part of the Webdam European project. A more detailed presentations of the results of the project may be found on the Webdam Web site at http://webdam.inria.fr/.

II. A distributed knowledge base

Information that is associated with a particular Web user, and that we aim to represent and reason with, comes in a variety of forms. For example, user Alice may have the following types of information:

- Data: text and XML documents, photos, and videos.
- Metadata: who took a particular photo of Alice, when and where.
- Data localization: where is Alice keeping her photos.
- Access right: who is allowed to see Alice's photos.
- Credentials: Alice's passwords on Picasa and Facebook.
- Knowledge about data of other peers: replication policy, trust in a peer, belief in a particular statement.

- Provenance information: from which peer was a particular photo obtained, when, and how.

Currently, Alice carries the burden of managing these and other types of information in various ways, including bookmark lists, password files, and replicas of the data on one or several backup systems. Our goal is to have the system manage this information on Alice's behalf, and automatically access and combine appropriate pieces of information to accomplish tasks. We now illustrate this with an example.

Suppose that Alice is organizing a rock climbing outing in Fontainebleau, and wishes to put together an application for the event that she will share with the members of her rock climbing group. Part of the data she needs is the list of group members, which is stored on Facebook. To promote the event, she also wants to use pictures from previous outings, which members of the group store on public sites, such as Picasa or Flickr, private websites, and in an untrusted DHT that group member Bob has set up. Finally, Alice will need to obtain some information from public Web services, e.g., she may use Google maps to get the locations of bouldering areas in Fontainebleau.

Using existing technology, Alice will have to use several different tools and APIs to check what information is available, from which Web services, and whether she has access to it. Having identified all relevant and accessible information, Alice will then need to understand how to retrieve it. We believe that many of these tasks can be specified by declarative queries and evaluated efficiently by a system equipped with reasoning capabilities. The system will process the queries over a distributed knowledge base, automatically dealing with thorny issues such as where to find some data, how to obtain access rights, and which protocols/interfaces to use. In our example, the knowledge base will be used to obtain the list of group members from Facebook, find where each member stores their photos from rock climbing outings, and get the photos using proper credentials. Note that this reasoning has to be performed in an extremely heterogeneous setting, where systems, interfaces, access control and data organization (e.g., ontologies) may widely vary across members of the group.

The kinds of reasoning tasks we are envisioning, and that are to be captured by *rules*, therefore include:

- Accessing information. Knowledge is used to localize data, e.g., find which systems hold the information of interest. Also, when a new source of information is discovered, some simple reasoning may be required to understand how it should be used, and how to obtain access rights.
- Peer's policy. Each peer specifies its own policy, which includes choices such as where to store/search for particular information, which data to serve to other peers, and which data to replicate. Such policies in social networks are typically defined based on information such as the composition of user groups ("circles" in Google+ terminology).
- Ontology processing. A particular source may structure its information in a particular manner or even describe it using RDF or RDFS. Reasoning is necessary to query this information. In particular, when accessing different information sources, knowledge is needed to align their concepts and relations.
- Quality management. Reasoning may be needed to assess the truthfulness of some data or to choose between contradicting information. This is related to evaluating the confidence one has in some data, the trust in sources, and, more generally, the beliefs of a particular peer or user.
- Knowledge acquisition and dissemination. These are central issues in this context. Knowledge acquisition, i.e., acquiring new facts and rules and evaluating their quality, should provide principled mechanisms that protect against (i) accepting any kind of information that is published by anyone on the Web and (ii) revising opinions too easily and in an ad-hoc manner (e.g., believing the last person who spoke).

Carrying out data management tasks in the Web setting typically requires direct involvement from the user, e.g., to acquire and transform information, or to download and execute applications. Towards the goal of automating these tasks, we implemented the WebdamExchange system, which will be described in Section IV. In the first version of the system, peer policies were implemented in Java. However, as we quickly realized, a user's interaction with the system could be made much easier if policies were specified declaratively. Beyond supporting an intuitive way for understanding, creating, and customizing policies, a declarative specification of rules in a distributed knowledge base allows each peer to act independently, based on its own data and logic. We express the peer's logic in Datalog-style rules of Webdamlog, a language we will describe in Section III. As we will see, peers exchange facts (for information) and rules (in place of code).

Such a holistic approach based on declarative rules provides the following advantages:

- Reasoning. Peers may perform automatic reasoning using the available knowledge. The global system therefore becomes "intelligent", and an improvement in the quality of service and of information can be expected.
- Formal model. Because the model is formally defined, it becomes possible to prove (or disprove) desirable properties such as soundness (data is only acquired legitimately) and completeness (all legitimate data is acquired). This is in the spirit of [1] that uses logic to describe access control protocols.
- Performance. Because the model is based on a Datalog-style language, it can benefit from optimization techniques. This is in the spirit of works from UC Berkeley on declarative programming for distributed systems [2], who show encouraging performance results with Datalog-style languages for applications such as Internet routing [3].
- Quality control. Because our model represents provenance [4] and time, we can better control the quality of data. Provenance also plays an important role in

specifying and enforcing access control and security policies, e.g., detecting who is responsible for misuse of the system.

- Large spectrum. Because the model is general, it can represent very different scenarios, ranging from centralized to massively distributed systems, from fully trusted to untrusted peers, and providing both encrypted and unencrypted information. Furthermore, the model can capture arbitrarily rich combinations of scenarios, which is the reality of today's Web.

III. WEBDAMLOG

In [5], we introduced Webdamlog, a novel Datalog-style rule-based language. In Webdamlog, each piece of information belongs to a *principal*. We distinguish between two kinds of principals: *peer* and *virtual principal*. A peer, e.g., *AlicePhone* or *Picasa*, has storage and processing capabilities, and can receive and handle queries and update requests. A virtual principal, e.g., *Alice* or *RockClimbingClub*, represents a user or a group of users, and relies on peers for storage and processing. We further distinguish between *facts*, representing local tuples and messages between peers, and *rules*, which may be evaluated locally or delegated to other peers.

The model has a formal semantics. Although we showed that, under some strong conditions, the semantics coincides with that of a centralized system, Webdamlog is primarily meant to be used in a distributed setting. Perhaps the main novelty of the language is the notion of *delegation*, which amounts to a peer installing a rule on another peer. In its simplest form, delegation is a remote materialized view. In its general form, it allows peers to exchange knowledge beyond simple facts, providing the means for a peer to delegate work to other peers. We will not describe Webdamlog in detail here, but will illustrate it with examples, referring an interested reader to [5] for details.

The following are examples of Webdamlog facts:

agenda@*AlicePhone* (12/12/2012, 10:00, John, Orsay)
photos@*Picasa* (fileName:picture34.jpg,
date:09/12/2012, byteStream:010001)
writeSecret@*Picasa* (login:Alice, password:HG-FT23)

The first fact represents a tuple in relation *agenda* on peer *AlicePhone* with information about an upcoming meeting, and the second — a photo in Alice's Picasa account (a tuple in relation *photos* on peer *Picasa*). The third fact represents Alice's login credentials for her Picasa account (in relation *writeSecret* on peer *Picasa*). Suppose that Alice wishes to retrieve, and store on her laptop, photos from Fontainbleau outings that were taken by other members of her rock climbing group. To this effect, Alice issues the following rule:

outingPhotos@*AliceLaptop* (**$pic**) :-
rockClimbingGroup@*Facebook* (**$member**),
findPhoto@*AliceLaptop* (**$member**, **$photos**, **$peer**),
$photos**@**$peer (**$pic**, **$meta**),
contains@**$peer** (**$meta**, "Fontainbleau")

This rule is a standard Webdamlog rule that illustrates various salient features of Webdamlog. First, the rule is declarative. Second, the assignment of values to peer names (e.g., **$peer**) and relation names (e.g., **$photos**) is determined during rule evaluation. Third, for **$peer** assigned to a system other than *AliceLaptop* (e.g., Picasa or Flickr), the activation of this rule will result in activating rules (by delegation), or in some processing simulating them in other systems (e.g., Picasa or Flickr). The evaluation of rules such as this one is performed by the WebdamExchange system, which includes a Webdamlog evaluation engine, and is responsible for handing communication and security protocols, and for invoking the Wedbamlog engine at each peer. WebdamExchange is described in the next section.

IV. THE WEBDAMEXCHANGE SYSTEM

The main goal of the *WebdamExchange* system is to provide an abstraction layer for communication and security protocols over heterogeneous peers on the Web.

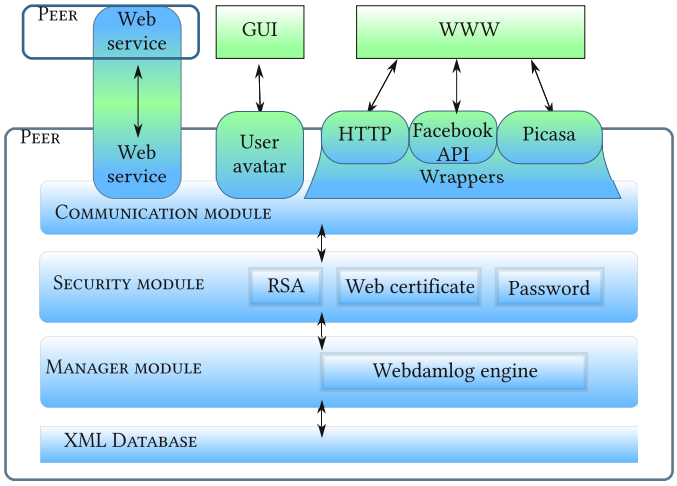

Fig. 1. Architecture of the *WebdamExchange* system

Figure 1 presents the architecture of the *WebdamExchange* system. We illustrate the execution of *WedbamExchange* using the rule *outingPhotos*@*AliceLaptop*, which retrieves and stores on Alice's laptop photos from Fontainbleau outings that were taken by other members of her rock climbing group.

Alice interacts with the system through the *GUI*, which provides a user-friendly way to specify the rule through a Web-based interface. *User Avatar* translates Alice's specification into the logical statement given at the end of Section III.

Communication Module annotates the rule with the communication protocol used by Alice, which in this case is the HTTP session started with Alice's login and password. The statement is then sent to the *Security Module*, which checks the authenticity of Alice using the *Password* component.

The authenticated statement is sent to the *Manager Module*, that decides whether to accept or reject it according to access control policies of the peer. In our current implementation [6] standard policies are defined using access control lists, and are

based on the name of the peer contained in the statement meta-data. Mechanisms for specifying and enforcing access control are described in [4]. If the rule is accepted, it is passed to the *Wedbamlog engine* for evaluation. All facts and rules processed by the *Manager Module*, along with their provenance meta-data, are recorded in the *XML Database* to be able to prove the correctness of reasoning if needed.

Suppose that Bob is a member of the rock climbing group, and so **$Member** = Bob. Suppose that Bob has stored some of his photos on peer *Picasa*, and some others on *BobLaptop*, in relation *BobPhotos*. To retrieve Bob's photos, the *Wedbamlog engine* issues the following two delegations:

outingPhotos@AliceLaptop (**$pic**) :-
  BobPhotos@BobLaptop (**$pic**, **$meta**),
  contains@BobLaptop (**$meta**, "Fontainbleau")

outingPhotos@AliceLaptop (**$pic**) :-
  BobPhotos@Picasa (**$pic**, **$meta**),
  contains@Picasa (**$meta**, "Fontainbleau")

The first delegation is sent to *BobLaptop*, which is a WedbamExchange peer. Then the *Manager module* annotates this delegation using RSA security and Web service communication protocols, as defined in our standard policy for *WebdamExchange* peer data transfer. The delegation is then passed to the *Security Module*, which signs and encrypts it. Finally, *Communication Module* sends the delegation using the Web services protocol.

The second delegation has to be sent to Picasa, which is not a *WebdamExchange* peer, and so cannot install the rule in its program. However, the semantics of this rule correspond to an operation that Picasa is able to perform, which is to select the photos with "Fontainbleau" in their meta-data. Communication with Picasa is enabled by a wrapper, which translates the statement into an HTTP request. In this case, we assume that the wrapper has created a schema, containing relations *BobPhotos@Picasa*(...) and *contains@Picasa*(...). This schema corresponds to the mapping between the data residing on Picasa and its representation in our system. To summarize, the communication and security policies defined by the *Manager*, *Communication* and *Security* modules allow us to communicate with a variety of peers.

A prototype implementation of *WebdamExchange*, with limited delegation and access control, was described in [6]. Supporting full rule delegation, and specifying and enforcing access control policies that arise as a result of rule delegation are a subject of our ongoing work.

V. Conclusion

In this paper, we outlined challenges faced by everyday Web users, who manage and interact with information that is inherently heterogeneous and distributed. We proposed to view the Web as a distributed knowledge base of logical facts and rules, and described Webdamlog, a Datalog-style language that enables automated reasoning over this knowledge base. The main novel feature of Webdamlog is its use of rule delegation (the installation of a rule by a peer on some other peer).

The information found on the Web is typically uncertain, imprecise, possibly inconsistent. We are building on some recent works, notably probabilistic XML [7] and corroboration [8] to enrich Webdamlog with probabilities. More precisely, we are working on introducing probabilities (for imprecision) and functional dependencies (for inconsistencies) in Webdamlog.

We described the architecture of WebdamExchange, a system that combines Webdamlog evaluation with communication and security protocols, and that we are currently developing. In our current work on WedbamExchange, we tackle two important questions: (i) how can a Wedbamlog program be evaluated efficiently, given that rules may be added or removed as a result of delegation; and (ii) how can we specify and enforce access control policies in presence of rule delegation.

Acknowledgment. This work has been partially funded by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013) / ERC grantWebdam, agreement 226513. http://webdam.inria.fr/

References

[1] M. Abadi, "Logic in Access Control (Tutorial Notes)," in *Foundations of Security Analysis and Design V*, A. Aldini, G. Barthe, and R. Gorrieri, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, vol. 5705, ch. 5, pp. 145–165. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-03829-7_5

[2] J. M. Hellerstein, "Datalog redux: experience and conjecture," in *Proceedings of the Symposium on Principles of Database Systems*, 2010.

[3] B. T. Loo, T. Condie, M. Garofalakis, D. E. Gay, J. M. Hellerstein, P. Maniatis, R. Ramakrishnan, T. Roscoe, and I. Stoica, "Declarative networking: language, execution and optimization," in *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, ser. SIGMOD '06. New York, NY, USA: ACM, 2006, pp. 97–108. [Online]. Available: http://doi.acm.org/10.1145/1142473.1142485

[4] S. Abiteboul, A. Galland, and N. Polyzotis, "A model for web information management with access control," in *Proceedings of the International Workshop on the Web and Databases*, 2011.

[5] S. Abiteboul, M. Bienvenu, A. Galland, and E. Antoine, "A rule-based language for Web data management," in *Proceedings of the Symposium on Principles of Database Systems*, 2011.

[6] E. Antoine, A. Galland, K. Lyngbaek, A. Marian, and N. Polyzotis, "[Demo] Social Networking on top of the WebdamExchange System," in *Proceedings of the International Conference on Data Engineering*, 2011. [Online]. Available: http://hal.inria.fr/inria-00536361/en/

[7] B. Kimelfeld and P. Senellart, "Probabilistic XML: Models and complexity," 2011, preprint available at http://pierre.senellart.com/publications/kimelfeld2012probabilistic.pdf.

[8] A. Galland, S. Abiteboul, A. Marian, and P. Senellart, "Corroborating Information from Disagreeing Views," in *Proceedings of the International Conference on Web Search and Data Mining*, 2010. [Online]. Available: http://hal.inria.fr/inria-00429546/en/