Rule-Based Application Development using Webdamlog

Serge Abiteboul, Émilien Antoine, Gerome Miklau, Julia Stoyanovich, Jules Testard

Rule-Based Application Development using Webdamlog *

Serge Abiteboul
Inria Saclay & ENS Cachan
first.last@inria.fr

Émilien Antoine
Inria Saclay & ENS Cachan
first.last@inria.fr

Gerome Miklau
Inria Saclay & UMass Amherst
miklau@cs.umass.edu

Julia Stoyanovich
Drexel University & Skoltech
stoyanovich@drexel.edu

Jules Testard
Inria Saclay & McGill U.
Jules.Testard@mail.mcgill.ca

May 14, 2013

We present the WebdamLog system for managing distributed data on the Web in a peer-to-peer manner. We demonstrate the main features of the system through an application called Wepic for sharing pictures between attendees of the SIGMOD conference. Using Wepic, the attendees will be able to share, download, rate and annotate pictures in a highly decentralized manner. We show how WebdamLog handles heterogeneity of the devices and services used to share data in such a Web setting. We exhibit the simple rules that define the Wepic application and show how to easily modify the Wepic application.

1 Introduction

Information management on the Internet relies on a wide variety of systems, each specialized for a particular task. A user's data and favorite applications are often distributed, making the management of personal data and knowledge (i.e., programs) a major challenge. Consider *Joe*, a typical Web user who has a blog on Wordpress.com, a Facebook account, a Dropbox account, and also stores data on his smartphone and laptop. *Joe* is a movie fan and he wants to post on his blog a review of the last movie he watched. He also wishes to advertise his review to his Facebook friends and to include a link to his Dropbox folder where the movie has been uploaded. This is a cumbersome task to carry out manually, yet automating it, for example by writing a script, is far beyond the skills of most Web users.

Some systems attempt to provide integrated services to support such needs. For instance, Facebook provides a wrapper service to integrate Dropbox accounts and blogs. However, such services are often limited in the functionality they support. Also, by delegating such services to systems like Facebook, a user needs to trust more and more of his information to one particular system. Our goal is to enable the user to easily specify distributed data management tasks *in place* (i.e., without centralizing his data to a single provider), while allowing him to keep full control over his own data [4]. Our system is not a replacement for Facebook, or any centralized system, but it allows users to launch their customized peers on their machines with their own personal data, and to collaborate using Web services.

This demonstration presents the Wepic application, a distributed picture manager. The Wepic application is specified using simple rules written in a declarative language called WebdamLog [2] and it runs on the WebdamLog system, demonstrated here for the first time. WebdamLog is a rule-based datalog-style language that emphasizes *cooperation* between *autonomous* peers by allowing both data, and programs that compute on data, to be easily exchanged. The declarative approach alleviates the conceptual complexity on the user while, at the same time, allowing for powerful performance optimizations on the part of the system.

A central issue in such a setting is the ease with which a casual user can write WebdamLog rules. We conducted a user study, showing that users are able to both understand and write simple WebdamLog programs with a minimal amount of training [5]. This demonstration further argues for the simplicity of using WebdamLog for designing applications that handle personal data.

SIGMOD attendees will use Wepic to share, download, rate and annotate pictures taken at the conference. Attendees will use the application via a Web GUI or launch their own Wepic peer. They will first inspect the basic WebdamLog rules of the provided application and will then be invited to customize the application by modifying or adding rules.

2 Language and System

WebdamLog language, in brief WebdamLog extends datalog in a number of ways, supporting updates, distribution, negation, and a novel feature called delegation.

Facts. A *fact* is an expression of the form $m@p(a_1, ..., a_n)$, where $a_1, ..., a_n$ are data values and $m@p$ is a relation described by relation name m and peer name p. Variables start with the symbol \$, e.g. $\$x$, and data values are quoted. An example of a fact is:

pictures@SIGMOD(32, "sea.jpg", "Émilien", 100..., ...)

where pictures@SIGMOD is a relation managed at the SIGMOD peer to represent a collection of pictures. The picture name is "sea.jpg", its content is in binary "100...", and the remainder of the fact consists of meta-data about the picture.

Rules. A *term* is a constant or a variable. A *rule* in a peer p is an expression of the form:

$\$R@\$P(\$U) :- \$R_1@\$P_1(\$U_1), \ldots, \$R_n@\$P_n(\$U_n)$

where \$R and $\$R_i$ are relation terms, \$P and $\$P_i$ are peer terms, \$U and $\$U_i$ are vectors of terms.

Distribution/Delegation. WebdamLog supports distribution, allowing atoms in the body of a rule to refer to relations on remote peers (using variables denoting relations or peers). The evaluation of such a rule therefore requires the knowledge and processing of possibly more than one peer. In addition, delegation allows a peer to install a rule at a remote peer, where the rule itself may refer to local or remote relations. Rule bodies in WebdamLog are evaluated from left to right. The order matters, unlike in datalog where the order of atoms in a rule

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

body is not relevant. Although negation is supported by the language, it is not yet implemented in the WebdamLog system.

WebdamLog peers, in brief Each peer runs a WebdamLog program, i.e., a set of rules, using a WebdamLog engine. The evaluation of datalog has been studied for decades but a renewed interest in datalog [1, 8, 9] has led to the recent development of the Bud [6] datalog engine, which we use as part of our implementation. The Bud system supports efficient communication between peers and updates of base facts.

A computation stage of the WebdamLog engine is broken down into three steps. First, the peer loads the inputs received from the remote peers since the previous stage. Second, the peer runs a fixpoint computation of its program. Third, the peer sends facts (updates) and rules (delegations) to other peers. A main novelty is the possibility for WebdamLog rules to have variables as relation and peer names. Another main novelty is delegation, which leads to installing, at run-time rules, at other peers. As a result, WebdamLog programs may be very dynamic, peers may discover new peers and new relations and they may acquire new rules as the result of delegations.

The WebdamLog system follows prior work on a P2P system for sharing data called WebdamExchange [3, 7]. The present focus is very different (declarative specification and delegation in WebdamLog vs. security in WebdamExchange), so little of the previous system could be re-used.

Wrappers The WebdamLog system has been designed for integration with the personal data of regular Web users who already use popular Web systems such as Facebook, Picasa or Dropbox. The WebdamLog system supports the management of data in these systems by using wrappers. A wrapper to some existing system X provides software that exports to WebdamLog one or more relations corresponding to the data in X, as well as rules to access/update this data.

For example, in the demonstration we will use a Facebook wrapper. For a given Facebook user Émilien, our wrapper will simulate a peer ÉmilienFB with two relations:

friends@ÉmilienFB($userID, $friendName)

pictures@ÉmilienFB($picID, $owner, $URL)

These two relations provide an abstract view of Émilien's Facebook data relevant to this particular application, and can then be used in WebdamLog rules.

Access control Since many WebdamLog applications will manage personal or social data, access to sensitive information must be carefully controlled. Access control in WebdamLog is particularly challenging because of the distributed nature of computation and the ability of peers to delegate rules to other peers. We describe in the next section the features included in the demonstration for controlling the delegation of rules, in which a peer is asked to explicitly accept delegations initiated by other peers.

A complete access control model for WebdamLog is under active investigation and is not included in the demonstration. In that model, access to stored or derived relations is controlled by a novel combination of both discretionary methods (in which users have the power to grant rights to data they own) and mandatory methods (in which access rights are derived according to system-wide conventions). Users directly specify the accessibility of stored relations that they own. For derived relations (i.e. views), a user may rely on a default access control policy that is derived automatically from the provenance of the base relations. Alternatively, a user may override this policy in order to grant access to views, effectively "declassifying" some data. This flexible model subsumes the view-based access control of the standard SQL authorization model.

3 The Wepic application

To demonstrate the WebdamLog system, we introduce an application called Wepic, a conference picture manager for the SIGMOD conference. This application allows conference attendees to share their pictures and to rate, annotate and download the pictures of others.

Wepic consists of a small set of rules that we discuss further. In addition, the application uses two standard wrappers we implemented, one for Facebook, and one for email communications. The WebdamLog system provides a graphical interface, which has been customized to provide a user interface for Wepic. It is shown in Figure 1.

A user connected to a Wepic peer can:

1. Upload a picture from a file or a URL;
2. View pictures provided by a particular attendee;
3. Transfer pictures:
   a) send them by email to the SIGMOD group on Facebook, or to another Wepic peer,
   b) get pictures from another Wepic peer or from the SIGMOD group on Facebook;
4. Annotate pictures with ratings, comments or name tags (names of attendees appearing in the picture);
5. Select and rank photos based on their annotations.

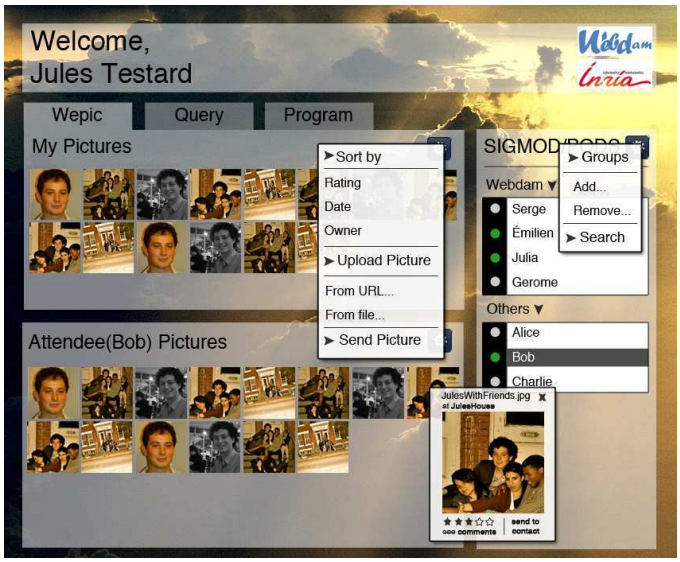

Figure 1: A screenshot of the Wepic user interface.

We now illustrate how some of these units of functionality are implemented with WebdamLog rules. To view pictures uploaded by a particular SIGMOD attendee, we use a relation selectedAttendees that contains one fact for each currently highlighted attendee (see right-hand side column in Figure 1). We also use a derived relation pictures, which is the view of all the pictures of a particular attendee. To obtain the pictures of all selected attendees, we use the rule:

attendeePictures@Jules($id, $name, $owner, $data) :-
selectedAttendee@Jules($attendee),
pictures@$attendee($id, $name, $owner, $data)

Note that this rule uses delegation, a feature novel to WebdamLog, to retrieve the contents of relation pictures of each attendee. The result of executing this rule is shown in the *Attendee pictures* frame at the bottom of Figure 1.

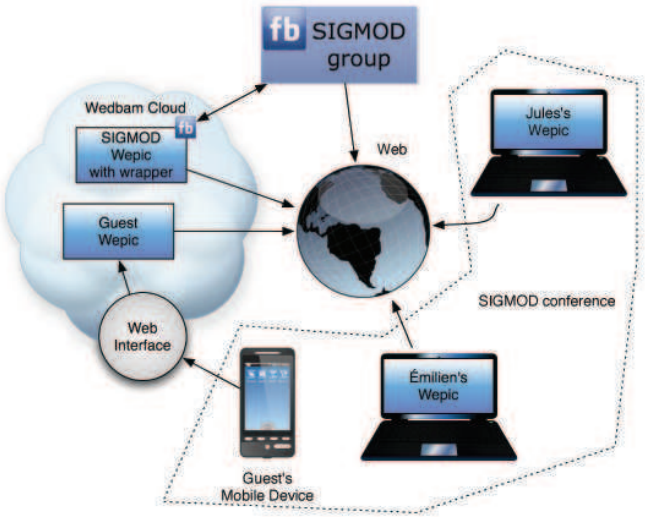

Figure 2: The distribution of peers in the network.

To transfer pictures between peers, we assume that each attendee specifies some preferred communication protocols in relation communicate, stating, e.g., whether he prefers to receive pictures by email, by posting on Facebook, or directly in his Wepic peer. The following rule is executed when Jules sends some pictures to some attendees:

```
$protocol@$attendee($attendee, $name, $id, $owner) :-
        selectedAttendee@Jules($attendee),
        communicate@$attendee($protocol),
        selectedPictures@Jules($name, $id, $owner)
```

Rules of this kind, and other rules implementing the basic functionality of Wepic, are available for inspection and customization through the user interface, see Figure 3.

Delegation and access control As noted in Section 2, by using delegation a user may write a rule and ask another peer to process it remotely. Consider again the previous rule:

```
attendeePictures@Jules($id, $name, $owner, $data) :-
        selectedAttendee@Jules($attendee),
        pictures@$attendee($id, $name, $owner, $data)
```

Suppose we have the facts:

```
selectedAttendee@Jules("Émilien")
```

The evaluation of the rule leads to delegating the following rule to Émilien:

```
attendeePictures@Jules($id, $name, $owner, $data) :-
        pictures@Émilien($id, $name, $owner, $data)
```

to send all the facts in his relation pictures to Jules. This is a simple case of delegation, which can be controlled by *inferring access* from the specifications described above. However delegated rules can be more complex, and general methods for effectively controlling delegation are a topic of continued investigation. The demonstration of Wepic will provide a simplified model for control of delegation, in which each delegation sent by an untrusted peer will be pending in a queue until the user explicitly accepts it via the Web interface. A notification of a pending delegation can be seen at the top of Figure 3, where Julia is sending a rule to Jules. By default, all peers except the SIGMOD peer will be considered untrusted.

4 Demonstration Scenarios

We now describe the general proceedings of the demonstration. The goal will be to share pictures taken during the SIGMOD conference. Émilien and Jules are attendees of the conference. They have used Wepic to install locally on their laptops a collection of pictures. They demonstrate how to use Wepic with the native functionalities described in Section 3 and how to customize the application. They will also allow a user at the conference to run his own Wepic peer to explore the system. This scenario will demonstrate the various aspects of WebdamLog, notably distribution, delegation and control of delegation.

Setup In the beginning of the demo, three peers are established: one on each of the laptops of Émilien and Jules, connected via a local network, and a third, the SIGMOD peer, hosted on *Webdam cloud*. To simplify the presentation, we will assume that Émilien and Jules organize their data and behave similarly. They both store their photos in pictures@Émilien and pictures@Jules on their respective Wepic peers. Both have Facebook accounts and are members of the SIGMODFB group, the official Facebook group of the conference. Finally, both users are subscribed to the SIGMOD peer, which stores the list of registered Wepic users.

The Wepic peers Émilien, Jules and SIGMOD are depicted in Figure 2. We will start the demonstration by quickly going over the setup, and will then ask the user to interact with Wepic according to the following scenarios.

Interaction via Facebook To illustrate the interaction between a Wepic peer and other Web services, we use a Facebook wrapper. For instance, the following rule is used by the SIGMOD peer to automatically publish, on the Facebook group of SIGMOD, the pictures belonging to SIGMOD attendees who have authorized this action:

```
pictures@SigmodFB($id, $name, $owner, $data) :-
        pictures@sigmod($id, $name, $owner, $data),
        authorized@$owner("Facebook", $id, $owner)
```

Conversely, the SIGMOD peer will automatically retrieve the pictures with their comments and tags from the Facebook group and publish them to SIGMOD peer. Note that the system thus allows any Wepic user to see or publish (via Wepic) pictures in SIGMODFB even without having a Facebook account. A user only needs to appropriately populate his authorized relation to control Facebook publication.

We will explain the WebdamLog rules that implement these interactions to audience members. We will then show that a photo uploaded by Émilien into his local relation pictures@Émilien is instantly published to pictures@SIGMOD, and then propagated to pictures@SIGMODFB.

Customizing rules The main advantage of a peer-to-peer system such as WebdamLog is the ability to customize a peer's behavior. Therefore the most novel trait of Wepic is that it lets the user customize existing rules and add his own rules. For example, a user who is interested only in the pictures that have a rating of 5 would customize the rule of the application as follows:

```
attendeePictures@Jules($id, $name, $owner, $data) :-
        selectedAttendee@Jules($attendee),
        pictures@($id, $name, $owner, $data),
        rate@$owner($id, 5)
```

Redefining this rule will change the contents of the frame *Attendee pictures* in Figure 1, which we will demonstrate. We will explain this rule and its effect to audience members, and will then allow them to customize the rule further, retrieving, e.g., only pictures that were taken by a certain SIGMOD attendee, or in which only certain attendees appear.

Illustration of the control of delegation To illustrate the control of delegation, Émilien will attempt to install a rule at Jules'

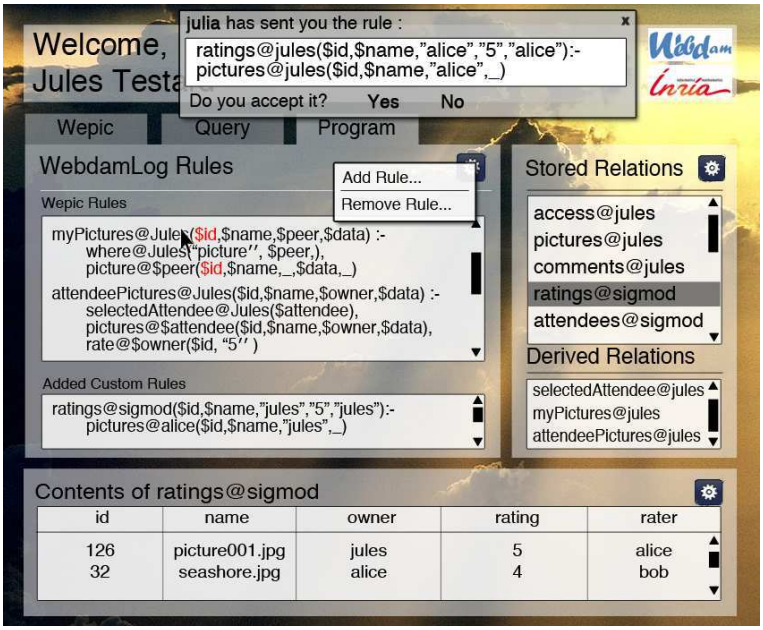

Figure 3: The interface to a WebdamLog program running Wepic.

peer. We will demonstrate that the system requires the approval of Jules before installing the rule, and that the program of Jules is changed once the approval is granted and the rule is installed.

Interaction via the Web Finally, we will invite audience members to launch their own autonomous Wepic peers in the *Webdam cloud*, and to interact with their peer through a UI, using their smartphone or a tablet that we provide. From that point on, they will be able to use all features of Wepic described in Section 3, e.g., upload their conference photos to their own peer. In addition, they will be able to use the *Query tab* to launch one of the pre-defined queries, or to write their own WebdamLog queries and rules with the assistance of the demo authors.

References

[1] S. Abiteboul et al. The active xml project: an overview. *VLDB J.*, 2008. 2

[2] S. Abiteboul et al. A rule-based language for Web data management. In *PODS*, 2011. 1

[3] S. Abiteboul et al. Web information management with access control. In *WebDB*, 2011. 2

[4] S. Abiteboul et al. Viewing the Web as a Distributed Knowledge Base. In *ICDE*, 2012. 1

[5] S. Abiteboul et al. The webdamlog system. Technical report, Inria, 2013. `http://hal.inria.fr/hal-00813300`. 1

[6] P. Alvaro et al. Consistency analysis in bloom: a calm and collected approach. In *CIDR*, 2011. 2

[7] É. Antoine et al. [Demo] Social Networking on top of the WebdamExchange System. In *ICDE*, 2011. 2

[8] J. M. Hellerstein. The declarative imperative: experiences and conjectures in distributed logic. *SIGMOD Rec.*, 2010. 2

[9] A. Morishima et al. Cylog/crowd4u: a declarative platform for complex data-centric crowdsourcing. *VLDB J.*, 2012. 2