Reactive Web Rules: A Demonstration of XChange

François Bry Michael Eckert Hendrik Grallert Paula-Lavinia Pătrânjan
University of Munich, Institute for Informatics
http://www.pms.ifi.lmu.de, {bry,eckert,grallert,patranjan}@pms.ifi.lmu.de

Abstract

Many Web applications involve reactive behavior such as updating data in response to events. Their development can be considerably eased by using XChange, a high-level and rule-based reactive Web language. Emphasizing the Web's distributed nature, our demonstration applies XChange to shared information and reactive services provided on the distributed Web sites of a scientific community.

1. Introduction

Many Web and Semantic Web applications are reactive: they respond to events such as user requests, messages from other applications, or changes of remote or local Web data. Common reactions include updating data or raising new events, which are sent to other Web nodes as messages [3].

In our demonstration, we realize the different Web sites of a (fictitious) distributed scientific community of historians called "Eighteenth Century Studies Society" (ECSS). ECSS consists of participating universities, thematic working groups, and a project management, each with their own Web site. The different Web sites are autonomous, but cooperate to react to and mirror relevant changes from other Web sites. For example, Web sites maintain information about personal data of members; a change of member data at a university entails further changes at the Web sites of the management and (some) working groups.

Rules play an important role in the Web and Semantic Web [8]. Production and Event-Condition-Action (ECA) rules provide a declarative means to specify reactive behavior. However, current rule systems assume a centralized, closed world and are not fitted for the distributed, open Web.

The reactive, rule-based language XChange [1, 5] simplifies programming both local (at a single Web node) and distributed (over several Web nodes) reactivity on the Web. XChange is based on ECA rules and tailored for the distributed nature and common data formats of the Web We demonstrate how the distributed reactive behavior in ECSS can be programmed easily and conveniently with XChange.

2. The Language XChange

We refer to [5] for a full introduction to XChange, and only describe its foundations and benefits here:

(i) XChange programs consist of ECA rules. These allow programming on a high abstraction level and are easy to analyze for both humans and machines.

(ii) XChange embeds the versatile Web and Semantic Web query language Xcerpt [7, 9] allowing to access and reason with Web data, and also provides an integrated update language (based on Xcerpt) for modifying Web data. Accordingly, the various parts of a rule all follow the same paradigm of specifying patterns for tree- or graph-structured data (e.g., XML, RDF) that is queried, newly constructed, or updated. This makes XChange elegant and easy to learn.

(iii) Both atomic and composite events can be detected and relevant data extracted from events [4, 2].

(iv) XChange enforces a clear separation of persistent data (Web resources, relating to *state*) and volatile data (events, relating to *changes in state*).

(v) XChange's high abstraction level and its powerful constructs allow for short and compact code.

Each Web node has it own XChange program, which consists of rules of the form `ON` *Event query* `IF` *Web query* `DO` *Action*. Such an ECA rule means: When events answering the event query are received and the Web query evaluates successfully, perform the action. Both event and Web query can extract data through variable bindings, which can then be used in the action. With this, we can see that both event and Web queries serve a double purpose of detecting *when* to react and, through binding variables, *how* to react.

For distributed applications XChange programs at different Web sites can coordinate each other by sending and receiving events as XML messages in a push-manner. We see this as an important advantage of ECA rules over production rules [3].

3. Description of the Demonstration

The different Web sites in ECSS maintain XML data about members, publications, meetings, library books, and

$r1$: ON change member
DO update LMU data

$r2$: ON change member
DO forward to management

$r3$: ON change member
DO update management data

$r4$: ON change member (w/WG3)
IF was not member of WG3
DO send add member to WG3

$r5$: ON change member (w/o WG2)
IF was member of WG2
DO send remove member to WG2

$r6$: ON remove member
DO update WG2 data

$r7$: ON add member
DO update WG3 data

Figure 1. Example: Affected Rules

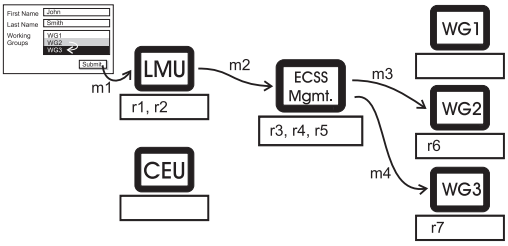

Figure 2. Example: Distributed Event Flow

newsletters. Data is often shared; e.g., a member's personal data is present at her home university, at the management node, and in the working groups she participates in. Events that occur in this community relate to changing member data, keeping track of library inventory, or distributing email newsletters for working groups. To keep the shared data consistent among the different Web sites, these events require reactions such as insertions, deletion, alteration, or propagation of data.

Corresponding reactions to events are specified as XChange rules. Abiding by the Web's distributed and decentralized nature, each node has its own local set of rules, which receive and process local and remote events.

Figure 1 gives an example of the rules triggered when a member's data, including working group affiliation, is changed. The distribution of the rules and the flow of events between the different Web nodes is shown in Fig. 2.

The initial change is made with a Web form at the member's home institution, sending event $m1$ to the LMU node. There, rule $r1$ reacts and locally updates the member's data at LMU accordingly, while rule $r2$ forwards the change to the management node as event $m2$. The management node has rules for updating its own local data about the member ($r3$) and for propagating the change to the affected working groups ($r4$ for adding, $r5$ for deleting a member). The working groups finally each have rules ($r6$, $r7$) reacting to deletion and insertion events ($m2$, $m3$).

Apart from this simple example of managing distributed member data, our demonstration realizes a community-owned and distributed virtual library (e.g., lending books, monitions, reservations), meeting organization (e.g., scheduling panel moderators), and newsletter distribution. These other tasks are also implemented by ECA rules that are in place at the different nodes.

For presentation purposes, the demonstration includes facilities for displaying the rules of each node and logging received and sent events. To illustrate its distributed nature, the demonstration runs on several machines.

4. Conclusions

Reactive rules for the Web are an emerging field [8]. XChange's ECA rules integrate reactivity, querying, and updating on the Web, and thus provide an elegant, declarative, easy-to-learn, and intuitive solution for realizing distributed, reactive Web applications. For ECSS, less than one month of an independent study project has been spent writing rules by a student without prior experience in XChange, Xcerpt, and rule-based programming.

XChange is an ongoing research project [10]; a prototype implementation is available and used to run the demonstration. Development focuses on further use cases and structuring rule sets [6], automatic generation of ECA rules, querying of composite events [2], and efficient evaluation.

Acknowledgments

This research has been funded by the European Commission and the Swiss Federal Office for Education and Science within the 6th Framework Programme project REWERSE number 506779 (`http://rewerse.net`).

References

[1] J. Bailey, F. Bry, M. Eckert, and P.-L. Pătrânjan. Flavours of XChange, a rule-based reactive language for the (Semantic) Web. In *Proc. Int. Conf. on Rules and Rule Markup Languages for the Semantic Web*. Springer, 2005.

[2] F. Bry and M. Eckert. A high-level query language for events. In *Proc. Int. Workshop on Event-driven Architecture, Processing and Systems*. IEEE, 2006. To appear.

[3] F. Bry and M. Eckert. Twelve theses on reactive rules for the Web. In *Proc. Int. Workshop Reactivity on the Web*, 2006.

[4] F. Bry, M. Eckert, and P.-L. Pătrânjan. Querying composite events for reactivity on the Web. In *Proc. Intl. Workshop on XML Research and Applications*. Springer, 2006.

[5] F. Bry, M. Eckert, and P.-L. Pătrânjan. Reactivity on the Web: Paradigms and applications of the language XChange. *J. of Web Engineering*, 5(1):3–24, 2006.

[6] F. Bry, M. Eckert, P.-L. Pătrânjan, and I. Romanenko. Realizing business processes with ECA rules: Benefits, challenges, limits. In *Proc. Int. Workshop on Principles and Practice of Semantic Web*. Springer, 2006.

[7] S. Schaffert and F. Bry. Querying the Web reconsidered: A practical introduction to Xcerpt. In *Proc. of Extreme Markup Languages Conf.*, 2004.

[8] W3C Rule Interchange Format. http://w3.org/2005/rules.

[9] Xcerpt. http://xcerpt.org.

[10] XChange. http://reactiveweb.org/xchange.