

Wide-area Event Notification Service

University of Colorado at Boulder
Software Engineering Research Laboratory

University of California at Irvine
Irvine Research Unit in Software

Creating a Scalable and Flexible Event Service

Introduction: An event service is a general-purpose facility that provides for *observation* and *notification* of events among objects. Several classes of applications make use of some sort of event service. Examples of such applications are monitoring systems, user interfaces, integrated software development environments, active databases, software deployment systems, content distribution, and financial market analysis. Many of these applications are also inherently distributed, and thus they require interaction among components running on different sites and possibly distributed over a wide-area network.

A number of systems provide event-based infrastructures. However, these technologies are designed primarily for local-area networks and therefore fail to scale to the dimensions of a wide-area network such as the Internet. Scalability poses new specific requirements.

SIENA Design: SIENA is an event service designed for *scalability*, the new challenge for event-based infrastructure technology.

The *scalability* problem can be characterized by the following dimensions:

- ◆ large number of objects generating events and requesting notifications;

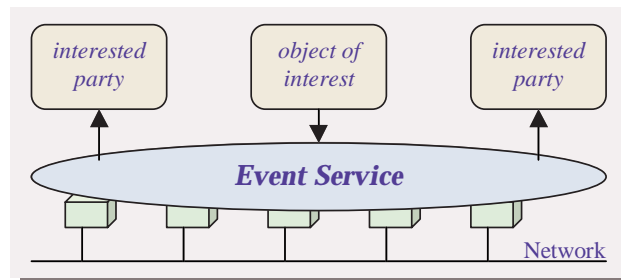


Figure 1: SIENA high-level architecture. Interested parties subscribe for events. Objects of interest publish events.

- ◆ large number of events;
- ◆ high event generation rates;
- ◆ objects distributed over a wide-area network (thus, low bandwidth, scarce connectivity and reliability);
- ◆ events of the same class generated by many different objects;

“Existing technologies fail to scale to the dimensions of a wide-area network such as the Internet. SIENA is an event service designed for scalability.”

- ◆ notifications of the same class of events requested by many objects;
- ◆ no centralized control nor global view of the structure of the event service.

SIENA also provides a *flexible notification model* that can serve application programmers as well as end users. Event notifications are structured as a set of *attributes*. Each attribute has a name, a type, and a value.

The interface of the SIENA event service allows objects to subscribe for specific classes of events, by setting up *filters*, or for specific sequences of events, by setting up *patterns*. Filters select events based on their *content* using a simple and yet

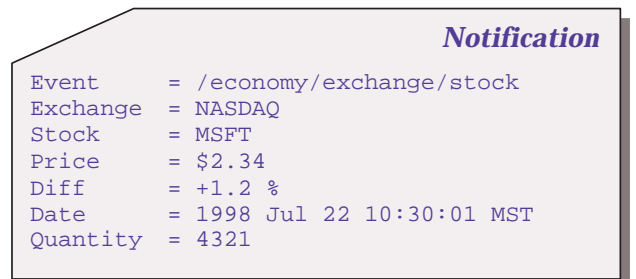


Figure 2: An example of SIENA notification model.

powerful query language. Patterns are combinations of filters that select temporal sequences of events. In addition to the *publish* and *subscribe* functions generally provided by most event-based middleware facilities, SIENA also exports a symmetric function called *advertise*. This function is used by objects of interests to declare which types of notifications they will publish. The additional information provided by advertisements is used by SIENA to optimize the delivery of notifications.

Event Service Architecture: SIENA delivers a scalable event service by adopting special dispatching protocols that aim at reducing network traffic and avoiding bottlenecks.

Depending on the topology of connections among SIENA servers, hierarchical or peer-to-peer, different algorithms have been implemented to deliver notifications. These are based on the propagation of subscriptions (*subscription forwarding*) or on the propagation of advertisements (*advertisements forwarding*). These two algorithms also roughly correspond to the main strategies that SIENA applies in filtering and multicasting notifications:

- ◆ *upstream filtering and assembly:* filters and patterns are pushed as close as possible to the sources of events, thereby immediately pruning the propagation of notifications that are not requested by any object.
- ◆ *downstream replication:* replication of notifications (multicasting) is pulled as close as possible to the targets of notifications. The idea being that, in order for a notification to reach several objects on distant networks, only one copy of that notification needs to traverse slow internetwork paths. That notification is then replicated and routed to all its destinations only when it gets to their local (less expensive) network.

Validation: The design of Internet-scale systems requires a special effort for validation. In particular, it is important to assess the impact of routing strategies and event pattern recognition with respect to costs such as network traffic, CPU, and memory usage.

The architectures of SIENA and its routing algorithms were studied by means of systematic simulations in various network scenarios with different ranges of loads and different configurations (see Figure 3).

Experience: Currently, a prototype of SIENA is used as wide-area messaging and event system of the Software Dock. There are two main implementations of the SIENA server—one (written in Java) realizes a hierarchical server, while the other (written in C++) has a peer-to-peer

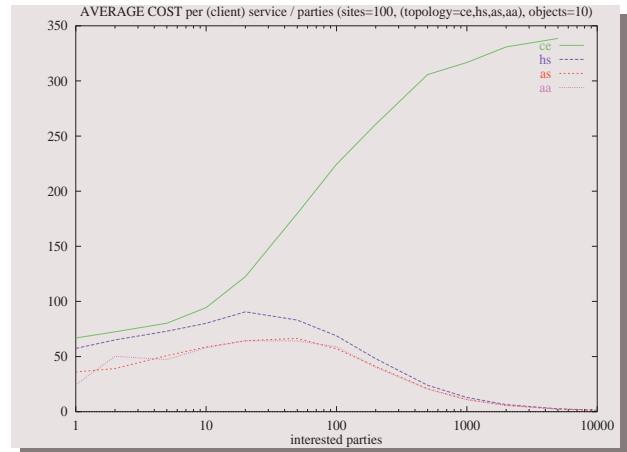


Figure 3: Simulation results: *cc=centralized*, *hs=hierarchical*, *as,aa=acyclic (peer-to-peer)*

architecture. The client interface is currently available for both Java and C++.

Future Work: SIENA will expand in several directions:

- ◆ We will complete the implementation of the SIENA servers, extending their capability to fully support advertisements forwarding and pattern observation.
- ◆ We will improve both the server and the client interface by extending the support for data types such as dates and URIs.

References:

- [1] D.S. Rosenblum and A.L. Wolf, "A Design Framework for Internet-Scale Event Observation and Notification", In *Proceedings of the 6th European Software Engineering Conference*, Zurich, Switzerland, September 1997.
- [2] A. Carzaniga, "Architectures for an Event Notification Service Scalable to Wide-area Networks", *PhD Thesis*, Politecnico di Milano, Italy, December 1998.
- [3] A. Carzaniga, D.S. Rosenblum and A.L. Wolf, "Challenges for Distributed Event Services: Scalability vs. Expressiveness", In *Engineering Distributed Objects '99*, ICSE99 Workshop, Los Angeles CA, USA, May 1999.

Principal Investigators

<p>Alexander L. Wolf</p> <p>University of Colorado at Boulder Department of Computer Science Campus Box 430 Boulder, CO 80309-0430</p> <p>Phone: +1 (303) 492-5263 Fax: +1 (303) 492-2844 E-mail: alw@cs.colorado.edu</p>	<p>David S. Rosenblum</p> <p>University of California, Irvine Information and Computer Science ICS2 209 Irvine, CA 92697-3425</p> <p>Phone: +1 (949) 824-6534 Fax: +1 (949) 824-1715 E-mail: dsr@ics.uci.edu</p>
---	--

Additional Information and Available Software
<http://www.cs.colorado.edu/serl/dot/siena.html>

Contact

Antonio Carzaniga
 Phone: +1 (303) 492-4463 - E-mail: carzanig@cs.colorado.edu

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. CCR-9701973. This effort was also sponsored by the Defense Advanced Research Projects Agency, and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-97-2-0021, and by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number F49620-98-1-0061. This work was also supported in part by the Air Force Materiel Command, Rome Laboratory, and the Advanced Research Projects Agency under Contract Number F30602-94-C-0253. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency, Air Force Research Laboratory, the Air Force Office of Scientific Research or the U.S. Government.