Active XML: Peer-to-Peer Data and Web Services Integration

Serge Abiteboul, Omar Benjelloun, Ioana Manolescu, Tova Milo, Roger Weber

INRIA, Tel-Aviv University, ETH-Zurich
What is Active XML (AXML)?

AXML documents: XML documents with embedded calls to (AXML) web services.

AXML web services: defined using XQuery over AXML documents.

Peer-to-peer architecture, where each peer:
1. manages AXML documents
2. provides AXML web services

The Goal: scalable data integration
May contain calls to any SOAP web service
- e-bay.net, google.com, babel.org, etc.
- AXML peers also offer web services.
- Locally defined services can be called.

Are enriched by each service call's results
The returned nodes are inserted as brothers of the corresponding <sc> element.

Can use XPath expressions for call parameters
Relative path expressions are evaluated starting from the <sc> element.

Activation of calls and data lifespan are controlled
- frequency: when is the service called?
- validity: how long is the retrieved data kept?
- mode: immediate or lazy?

AXML Documents are powerful data integrators.
A simple, declarative way to create web services...

- A service operation is specified as a query with parameters.
- It may query (local) AXML documents.
- It is made available on the web using the SOAP protocol.

Basic AXML services are compatible with current standards for web services invocation.

... which allows for new, powerful features.

- Intentional parameters and results: AXML documents (containing service calls) can be exchanged.
- Continuous services send back a stream of answers (SOAP messages) to the caller.

Used in AXML documents, AXML services are powerful tools for data integration.
Technical environment:
- SUN’s Java SDK 1.4 (includes XML parser, XPath processor, XSLT engine)
- Apache Tomcat 4.0 servlet engine
- Apache Axis SOAP toolkit 1.0 beta 3
- X-OQL query processor, persistent DOM repository
- JSP-based user interface, using JSTL 1.0 standard tag library
Communications

Peer to peer:
1. Peers communicate together only through service calls, using SOAP.
2. They don’t directly access other peers’ documents.

User interface:
1. Users interact with each peer through an HTML-based interface.
2. AXML Documents are displayed using XSLT stylesheets.
3. Transformations are done server-side using JSP pages.
4. Forms allow to invoke locally defined services, which update the documents.

Specific distributed applications can be easily built as AXML documents and services, and XSLT stylesheets.
Demonstration highlights

Each peer provides some auctions:
1. The document myAuctions.xml contains the peer's items and their bids
2. Services offered to other peers:
   1. getAuctions(),
   2. getHighestBid(auctionId),
   3. bid(auctionId, amount)

Each peer can bid on any auction:
1. myBids.xml keeps track of the peer's bids
2. Manually, using the Bid(auctionId, amount)
3. Automatically, using the local service bidUpTo(peer, auctionId, increment, limit)

Each peer knows about some peers' auctions:
1. knowAuctions.xml is an AXML document, containing calls to other peers that transitively retrieve their know auctions.
2. Offered service: getKnownAuctions()

When an auction closes, the winner is notified.

→ Functional bidding system, without a centralized server