The Evolution of Internet-Scale Event Notification Services

Past, Present, and Future

Adam Rifkin
Rohit Khare

Workshop on Internet-Scale Event Notification
University of California, Irvine
July 13, 1998
The Evolution of Internet-Scale Event Notification Services

**Past**
- Event-Based Integration (EBI) occupied application niches, adapted to loosely-coupled systems
- Event Notification Services (ENS) expanded their range, from hosts to LANs to WANs

**Present**
- Crossing trust domains raises new ‘Internet’ concerns

**Future**
- Explosion of diverse, competing protocol proposals
- Selection criteria likely to lead toward convergence
Goals of this Presentation

- "You can’t tell the players without a scorecard!"
  - Over a hundred systems and eighty more papers
- "What are the design choices?"
  - Identifying primary axes of classification
- "What’s new research here?"
  - Issues which are truly unique to Internet-scale ENSs
- "What’s going on in the marketplace?"
  - New players have wider ambitions than past ISENSs
- "How can we select more principled designs?"
  - Lessons from software architecture for ISENSs
Who We Are

Adam Rifkin
- Seventh-year at Caltech with K. Mani Chandy
- Microsoft Research, HP Labs, Rome Labs, NASA
- Studying the semantics of event models
  - Can formal specification help explain the behavior and performance of distributed systems?

Rohit Khare
- Second-year at UC Irvine with Richard Taylor
- W3C, Web Journal, MCI Internet Architecture
- Studying the design of application-layer protocols
  - How did all these TPs evolve? Are they converging?
The Evolution of Event Systems

- Defining Events, Notifications, and Handlers
  - Enumerating and clustering existing systems
- Evolution into new niches and widening range
- Archetypal applications and notification services
- Taxonomy of ENS design space
  - Evaluating models from the literature
Defining our terms

- People have called lots of things event systems:
  - graphical interfaces, physical simulations, collaborative workflow, programming with callbacks...

- Events are Notifications to be Handled
  - Events, as in physics, are abstract and instantaneous
  - Notifications are messages with definite semantics
  - Handlers implement synchronization and semantic constraints of a pattern of notifications

- Events are not: RPC, Blackboards, Pipes, Documents, or FYI Messages
Things that go “notify” in the night:

- ACA (Digital)
- Active Databases
- Active Software
- Actors
- Amalgame
- Amiga REXX
- AppleEvents
- Atlantis
- Backweb
- Bart
- BEA/Tuxedo
- BLIP
- BOF/BOCK
- C2 Style
- CISCO Pub & Sub
- COM+ Events
- Consul
- CORBA Notification
- CORBA Transaction
- Cronus
- Desert
- DCE RPC
- DEEDS
- DHCP
- DIS (IEEE 1278)
- Discrete Event Sim
- DNS Notifications
- DRP (Marimba)
- DSN (mail)
- East
- e-cast (Lucent)
- Elvin
- ENP (Oracle)
- Ensemble
- Field
- finger
- GENA (MS)
- FUSE
- HORUS
- HTTP
- ICQ
- iFlame
- Information Bus
- Infospheres
- Intermind
- Iona OrbixTalk
- IP Multicast
- IRC
- ISIS
- Java AWT
- Java Beans
- Java Distr. Events
- Java InfoBus
- Java OS events
- Java Transactions
- JavaSpaces
- JEDI
- JFC (Swing)
- Keryx (HP)
- Leases
- Logical Clocks
- Maisie
- Majordomo
- Mariposa
- Mediator Pattern
- Mentat
- MFTP
- MMS
- MSMQ
- MQ*Series
- MTP
- MTS
- NNTP
- NSTP
- OpenDoc
- PIPR
- PLAN
- Pointcast
- POLYLITH
- RIP
- RMP
- RPC
- RVP
- SGAP
- SIMNET
- SIENA
- SIP
- SMTP
- SNMP Traps
- SoftBench (HP)
- SRM
- SWAP
- SwitchWare
- Talarian
- Taligent
- Talkd
- Teamwave
- Tibco
- ToolTalk (Sun)
- Ubique
- Vitria
- VMTP
- WhoDP
- Win32 events
- X Windows
- Yahoo Pager
- Yeast
- Zephyr
### Clustering by Application Context

#### Messaging
- Backweb
- DRP (Marimba)
- DSN (mail)
- e-cast (Lucent)
- HTTP
- Intermind
- Majordomo
- MFTP
- MSMQ
- MQ*Series
- NNTP
- Pointcast
- SMTP
- SNMP Traps
- Talarian
- Teamwave
- Tibco
- VMTP

#### Presence
- Blip
- finger
- ICQ
- NSTP
- PIPR
- RVP
- SGAP
- SIP
- WhoDP

#### Chat
- iFlame
- IRC
- Talkd
- Ubique
- Yahoo Pager
- Zephyr

#### Simulation/Graphics
- Actors
- DIS (IEEE 1278)
- Discrete Event Sim.
- Java AWT
- JFC (Swing)
- Living Worlds (HP)
- Logical Clocks
- Maisie
- Mentat
- SIMNET
- Taligent
- Win32 events
- X Windows

#### Application Integration
- ACA (Digital)
- Active Software
- Amalgame
- Amiga REXX
- AppleEvents
- BEA/Tuxedo
- East
- Field
- HORUS
- Information Bus
- Iona OrbixTalk
- Java Beans
- Java InfoBus
- Keryx (HP)
- OpenDoc
- POLYLITH
- SoftBench (HP)
- SWAP
- ToolTalk (Sun)
- Vitria
- Yeast
# Clustering by Notification Features

Myriad ways to slice and dice this survey set...

<table>
<thead>
<tr>
<th>Low Latency</th>
<th>High Latency</th>
<th>Polling</th>
<th>Invocation</th>
<th>Multicast</th>
<th>Reliable</th>
<th>Leases</th>
<th>Dead-reckoning</th>
</tr>
</thead>
<tbody>
<tr>
<td>COM+ Events</td>
<td>SMTP</td>
<td>Backweb</td>
<td>CORBA Notification</td>
<td>IP Multicast</td>
<td>BEA/Tuxedo</td>
<td>DIS (IEEE 1278)</td>
<td>SIMNET</td>
</tr>
<tr>
<td>Amiga REXX</td>
<td>Majordomo</td>
<td>Finger</td>
<td>Field</td>
<td>RMP</td>
<td>CORBA Transaction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AppleEvents</td>
<td>DSN (mail)</td>
<td>ICQ</td>
<td>SoftBench (HP)</td>
<td>SIP</td>
<td>MSMQ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SNMP Traps</td>
<td>DNS Notifications</td>
<td>Pointcast</td>
<td>ToolTalk (Sun)</td>
<td>SRM</td>
<td>MQ*Series</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IRelay Chat (IRC)</td>
<td>NNTP</td>
<td>Yahoo Pager</td>
<td>Zephyr</td>
<td>Tibco</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Low Latency
- COM+ Events
- Amiga REXX
- AppleEvents
- SNMP Traps
- IRelay Chat (IRC)

## High Latency
- SMTP
- Majordomo
- DSN (mail)
- DNS Notifications
- NNTP

## Polling
- Backweb
- Finger
- ICQ
- Pointcast
- Yahoo Pager

## Invocation
- CORBA Notification
- Field
- SoftBench (HP)
- ToolTalk (Sun)
- Zephyr

## Multicast
- IP Multicast
- RMP
- SIP
- SRM
- Tibco

## Reliable
- BEA/Tuxedo
- CORBA Transaction
- MSMQ
- MQ*Series

## Leases
- DHCP
- AFS Caching
- WebDAV

## Dead-reckoning
- DIS (IEEE 1278)
- SIMNET
- WhoDP
The Evolutionary March of Progress: New niches and widening range...

News (NNTP), channels
Rendezvous (RVP), WhoDP, NSTP, SGAP
Internet Relay Chat (IRC)
VRML Environments
Reliable mcast (Tibco, MFTP)
Session Invitation (SIP)
Zephyr, iFlame
Distributed Interactive Sim (DIS /IEEE)
eMail (SMTP)
Netshow
Discrete Event Simulation (Maisie)
Simple Net Management Traps (SNMP)
TalkD
X Windows System
Mach IPC
Netshow
MacOS event loop, Language exceptions
Simple Net Management Traps (SNMP)
TalkD
GUI event queue
who(1)
write(1)
Cut-n-paste, Pub-n-sub, Exceptions
Login information (utmp)
Pipe-and-filter
Presence
Simulation/ Graphics
Tool Integration

Exploring the Fossil Record

- Our evolutionary map is only descriptive as yet
  - These five applications are not a privileged frame of reference; merely popular clusters

- To evaluate EBI styles, outline orthogonal axes:
  - Rate of event occurrence
  - Topology of notification distribution
  - Content model of notifications
  - Naming model: sources, sinks, queues, subscriptions
  - Event transformations: filtering, aggregation, etc.
  - Security and Privacy requirements
Application: Messaging

- Goal: Delivering “human-actionable” content
  - News occurs at $O(\text{minutes})$ to $O(\text{days})$
  - Sample event: “Today’s lunch menu is… ”

- Distributed 1-N (many) or 1-K (known set)

- Notifications range from text to multimedia

- Names: mailboxes, newsgroups, topics, etc.

- Transformations: compression, batch delivery

- Authenticated senders, content integrity & confidentiality
**Systems: Messaging**

- **E-mail and mailing lists**
  - Mail can be queued at relays; reliable, but $O(\text{days})$
  - Mlists require subscriber verification; allows digests

- **USENET News**
  - Articles, named by message-ID, can be posted to a set of groups within a distribution region until expiry

- **Publish & Subscribe**
  - Topic selection can be more specific than “group”, such as selection of characteristics of the messages

- **Web Push / “Channels”**
  - Poll for updated content; could be bundled, cached
Application: Presence

- Goal: maintaining awareness of people, devices
  - Changes in state occur on $O(\text{minutes})$
  - Sample Event: “Elvis has left the building…”
- Monitor “buddies” or “editors” (1-K)
  - Infrastructure typically designed for N-N, though
- Notifications can be lightweight (text)
- Names: users-from-directories, groups
- Transformations: batch update; state timeout
- Privacy requires knowing who’s watching
Systems: Presence

- **who (1), utmp**
  - Multiuser OSes log and report current logins
- **finger**
  - Returns last-seen, last-read, and .plan for username
- **WhoDP**
  - Switchboards maintain directory of clients; actual presence traffic redirected peer-to-peer
- **AOL Instant Messenger**
  - Centralized state server; client holds connection open
Application: Chat

- Goal: Delivering units of (human) conversation
  - Interaction occurs on O(seconds)
  - Sample event: “[Duke] smells a Wumpus...”
- Distributed 1-1, 1-K (lecture), or K-K (forum)
- Notifications range from text to multimedia
- Names: email addrs, handles, channels
- Security: speaker authentication, confidentiality through content encryption
- Privacy: audience enumeration
Systems: Chat

- **write (1)**
  - Displays to all users (except users blocking all writes)

- **TalkD**
  - Binary session request; requires active confirmation

- **Zephyr**
  - Delivers messagegrams with Kerberos authentication

- **Internet Relay Chat (IRC)**
  - Directed acyclic graph of servers using a ‘gossip’ alg.
  - Weak confidentiality of forum content (passwords)
  - Robots can automatically trigger events (invitations)
Application: Simulation

- Goal: Maintaining consistent (physical) state
  - Interaction occurs on $O(\text{milliseconds})$
  - Sample event: “[Duke] fires 9mm at the Wumpus... ”
  - Sample event: “Mouse button 2 clicked twice... ”
- Distribution limited to 1-1 or small K-K groups
- Notifications are usually small and stateful (for example, delta updates)
- Transformations: event aggregation, masking (filtering), batching, and dead-reckoning
GUI event queues
- Hardware devices deliver interrupts or are scanned, foreshadowing the invoke vs. poll options
- Event queue can select events by bounds, type, etc.
- Events can be updated, coalesced in the queue

Distributed Interactive Simulation (DIS)
- Physical and synthetic players in a wargame
- Myriad specific update message formats
- Lost updates replaced by predictions
Application: Integration

- Goal: wiring together component software
  - Data flows from \(O(\text{milliseconds})\) to \(O(\text{hours})\)
  - This wide range can be problematic
  - Sample Event: “Compiler finished foo.c as foo.o”
- 1-K; source usually unaware of consumers
- Notifications typically machine-readable streams
- Names: processes, hosts
- Transformations: data type adaptors
Systems: Integration

- Information Bus
  - Routes messages to groups based on content
- POLYLITH Software Bus
  - Redirects/repackages intermodule calls per bus wiring
- Field, Yeast
  - Central message repository triggers sw tools
- AppleEvents, REXX
  - Event-oriented user interface scripting languages
Developing an Architectural Model

- Zooming out from these details, we choose to separate the event infrastructure layer
- Above, applications rely on EBI services
  - Several classic papers speak to this model
- Below, EN services bound protocol designs

<table>
<thead>
<tr>
<th>Event Based Integration</th>
<th>Components/Tools, Connectors, Notifications, Requests</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>‘Logical’ routers, message transformers, application event semantics (e.g. dead-reckoning)</td>
</tr>
<tr>
<td></td>
<td>Reliable storage, synchronization, typing</td>
</tr>
<tr>
<td>Event Notification Services</td>
<td>Quality of Service, Link and message security</td>
</tr>
<tr>
<td></td>
<td>‘Physical’ routing topology, ‘physical’ naming, initiation rules (poll vs. interrupt)</td>
</tr>
<tr>
<td></td>
<td>Wire formats, stateful optimizations (session vs. packet), batching</td>
</tr>
</tbody>
</table>
Field

- Connects clients ("tools") with anonymous broadcast
- Central message server as a separate process
- Tools register interest in message-expressions
- Forwarded in order received
- No exception handling
  - e.g. access control, delivery constraint violations
- Policy tool can intercept, replace messages
- "tools advertise operations" informed SoftBench

[Reiss90]
Polythith

Tools bind their I/O ports to a Software Bus
  - ports identified by name, allowing retargeting
Module Interconnection Lang to wrap tools
  - Simple, Structured, and Pointer message types
  - Limited to simple filtering on channels
  - No explicit support for groups
  - No exception handling
    - e.g. ill-formed messages or incompatible connections
## EBI Framework

**EBI Framework** [Barrett, et al 96]

<table>
<thead>
<tr>
<th>Feature</th>
<th>FIELD</th>
<th>POLYLITH</th>
<th>CORBA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message Types</td>
<td>String</td>
<td>Simple, Structured, Pointer types</td>
<td>Simple, Structured, and Interface types</td>
</tr>
<tr>
<td>Registrar</td>
<td>Msg message server</td>
<td>Bus</td>
<td>ORBs</td>
</tr>
<tr>
<td>Router</td>
<td>Msg pattern matching</td>
<td>Bus</td>
<td>ORBs</td>
</tr>
<tr>
<td>Message Sending</td>
<td>Multicast</td>
<td>Point-to-Point, Multicast</td>
<td>Point-to-Point</td>
</tr>
<tr>
<td>Message Delivery</td>
<td>Non-polling (passive)</td>
<td>Polling (active)</td>
<td>Unspecified</td>
</tr>
<tr>
<td>Message Transform Functions (MTFs)</td>
<td>Filters, Policies</td>
<td>Filtering by bus channel</td>
<td>None</td>
</tr>
<tr>
<td>Delivery Constraints</td>
<td>Policy Priorities</td>
<td>Not User-definable</td>
<td>At-most-once, Best-effort</td>
</tr>
<tr>
<td>Grouping</td>
<td>Participant Groups</td>
<td>None</td>
<td>Participant and Router Groups (&quot;domains&quot;)</td>
</tr>
</tbody>
</table>

Presents a taxonomy of causes, not effects
C2 Style [Taylor, et al 96]

- Components respond to notifications, emit requests (asynchronously)
- Connectors coordinate all communication
  - First-class objects, function as routers, broadcasters, filters, prioritizers
- Messages = name + typed parameters
- Notifications of state changes flow “down”
- Requests for action flow “up”
ISE Observation and Notification

[Rosenblum, Wolf 97]

- Attributes of ‘Internet-Scale’
  - Geographical reach, autonomy, security, QoS
- Lifecycle of an event
  - Determination of which events shall be observable
  - Expression of interest in an event or pattern
  - Occurrence of an event
  - Observation
  - Relation of an event to a pattern of interest
  - Notification to an application
  - Receipt by the application
  - Response of the application
Framework [Rosenblum, Wolf 97]

- Object model of senders and receivers
- Event model characterizes event phenomena
- Naming model of references to items of interest
- Observation model of identifiable patterns
- Time model of events causing notifications
- Notification model of mechanisms to express interest and receive them
- Resource model for allocation and accounting
Moving to the Lower Layer: The ENS Design Space

- It isn’t easy. Confusion abounds:
  - “A Rendering may be received in one of two ways: either... returned as a value from a synchronous call [‘client-pull’], or... requested and then sent asynchronously, in chunks [‘server-push’]” -- HTTP-NG Interfaces
  - “Messages that represent commands must be synchronous and must provide the caller with a reply.” -- Field
- Often conflate blocking, synchronization, timing, and initiation -- these are all separable
Our Taxonomy of ENS Design Space

- Initiation
  - Source- (interrupt) vs. Sink-initiated (poll)
- Synchronization
  - Synchronous (batched) vs. Asynchronous (deferred)
- Blocking
  - Blocking vs. Non-blocking handlers
- Causality
  - Ordered vs. unordered, duplicate, and/or missing
- Timing (may be a spectrum)
  - Real- (deadlined) to Virtual-time (eventual) delivery
Secondary Taxonomy Concerns

- **Transport**
  - Reliability above vs. at the bearer service
  - Multicasting above vs. at the bearer service

- **Notification Content Model**
  - Externally-visible typing (MIME encodings)
  - Size
  - Streamable
  - Lossy content (multimedia)

- **Security**
  - Trusted event notifiers vs. trusted notifications
Why Event-Based Integration won on

- Loose coupling is a hallmark of Internet-scale development
- Allows dynamic communication topology
- Separates engineering tradeoffs for latency, efficiency
- But, these were mission-specific and not yet Internet-Scale...
New issues

- Not merely about scaling across space, time, and number of participants and events...
- ... but across Organizations:
  - Security concerns
  - Interoperability
  - Semantic (Ontological) agreement
  - Administrative Decentralization
  - Mobility
- Most of all, Evolvability of an ISEN Service
  - ... we have an opportunity to define a generic service
Unaddressed issues

- Performance models: nonexistent
  - We believe a model would converge with messaging performance, modulo “event-handling” time
  - Parameters such as event frequency, size, ...

- Queuing policies across a connection topology
  - Critical factor in scaling, yet usually unspecified in protocols

- Evaluation criteria
  - Scenarios, benchmarks, metrics, models

- Others?
Past, Present, Future

Competing protocol proposals
Selection criteria
The great thing about standards is...

- At WISEN alone:
  - Presence/Chat
    - NSTP/SGAP/RVP/WhoDP
  - Tool Integration
    - SWAP/DAV/IPP
  - Generic Notification Services
    - GENA/BLIP
- Lots of others in the quiet race to market
- Does this welter of proposals overlook anything?
Perhaps.

- "Connectors should be first-class objects"
  - Don’t hide subscription and queueing
- "Transport is an engineering decision"
  - ... not a semantic one
  - Beware of "reliable" datagrams and multicast
    - reinvent TCP and risk ACK implosion, respectively
- "With security aforethought"
  - Need security at the message level
- If we are aiming at a global infrastructure, community involvement is paramount
Our Conclusions

Revisiting our goals
“You can’t tell the players without a scorecard!”

- A staggering range of systems can be considered event-oriented
  - Events are notifications which trigger commands
- Need to tease apart EBI applications from EN Ss
  - In the past, EN systems presented both together
“What are the design choices?"

- We believe event notifications across the Internet are necessarily identifiable as messages
  - ISEN design space is slightly larger than IS messages
- Message delivery initiated by source or sink
  - Polling vs. Interrupts
    - often conflated with non-blocking vs. blocking semantics
    - often conflated with which-side-establishes-a-connection
- End-to-end delivery or via mediators (queues)
  - often conflated with “real time” vs. deferred
- Reliable delivery provided by or above transport
“What’s new research here?”

- Evolvability: how flexible can an ISENS be?
  - An opportunity to build “great infrastructure”
- Security: should the ENS be trusted or not?
  - Hop-by-hop trust may not be dynamic enough
- Performance modeling: what are the limits?
  - Analytic and statistical models not developed yet
- There’s also good engineering to be done...
  - (Not to mention standardization leadership)
“What’s going on in the market?”

- New applications and protocols are emerging for EBI and ENS across organizational (trust) lines
- Entrants usually leveraging a technology
  - Transport (UDP, mcast) or HTTP or both
- Some glimmer of a layered solution
  - Event notification separable from event schema
- Collaboration and groupware tools are leaders
“How can we select more principled designs?”

- We believe the C2 and “representational state transfer” architectural styles show promise.

- C2’s connectors, components, and notifications:
  - Can model (bridge) a range of current proposals.
  - Hint at design rules for verifying EBI.
  - Reuses a common ENS at varying levels of abstraction.
  - Perhaps a lattice of event notification services.

- Representational State Transfer’s messages:
  - Separate the artifact (wire) and ideal (remote) form.
  - Allow dynamism and scale through statelessness.
Recommendation: A Layered ISENS

- Wire protocol for notifications
  - Perhaps an “asynchronous HTTP”
- Notification management
  - Interfaces for advertising and subscribing
  - Queue management policies
  - Generic notification typing
- WebEvents Package
  - Trapping HTTP Method × Resource
  - Link maintenance
  - New-content (“push”)
- These are missing from current proposals!
For Further Information...

- NOTIFY BOF at IETF-Chicago
  - Chaired by Jim Whitehead, UC Irvine
- This presentation and our events bibliography
  - http://www.cs.caltech.edu/~adam/isen/
- Notifications mailing list
  - notification-subscribe@makelist.com
  - http://www.findmail.com/list/notification/
- Contact us
  - adam@cs.caltech.edu
  - rohit@uci.edu