oBIX XML Standards Teleconference
September 22, 2004, 11:00am – 11:30pm (EST)

Overview
Decided to implement subscribe using a poll model and to put asynchronous server messages to bed for awhile.

Attendees

<table>
<thead>
<tr>
<th>Name</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aaron Hansen</td>
<td>Tridium</td>
</tr>
<tr>
<td>Brian Frank</td>
<td>Tridium</td>
</tr>
<tr>
<td>Chuck Watson</td>
<td>Eaton</td>
</tr>
<tr>
<td>David Richards</td>
<td>Trane</td>
</tr>
<tr>
<td>Doug Ransom</td>
<td>Power Measurement</td>
</tr>
<tr>
<td>Dr. Ken Wacks</td>
<td>Ken Wacks Associates</td>
</tr>
<tr>
<td>Samuel Yang</td>
<td>Echelon</td>
</tr>
</tbody>
</table>

Discussion

• Do we need asynchronous messaging?
  • The two issues we'd like to address with asynchronous messaging are alarms and change of value messages.
  • Why would someone at the enterprise level want to subscribe to changes unless they were significant alarms?
    • Because there may be enterprises that want to monitor very large numbers of points. Individual polling doesn't scale.
  • Don't we want to be reporting to the enterprise at a higher abstraction?
    • Enterprise developers will scrub the data for presentation. How they scrub it is unknown because the enterprise hasn't generally had access to this data before. oBIX leaves open the ability to expose everything if desired. After all, if someone in the enterprise simply wants to know if something is on, we have to be able to expose any data point.
    • The oBIX SysService (discovery/data) doesn't dictate what is in it. It could be summary type data.
  • The biggest risk to adopting an async specification is none of them are finished yet.
  • Brian prefers poll event model (like OPC XML DA). This is where the client subscribes to items, but has to poll for new messages. If there are no new messages when the client polls, nothing is returned.
    • Server doesn't have to initiate communications to the client.
    • Nearly the same performance benefits.
    • Don't have to worry about lease time outs.
    • Client doesn't have to run a web server.
  • What about a client that can't contact a server? For example, the server is preconfigured to push messages to the client.
    • On the internet, communications are largely initiated by the entity who desires the data (like a browser requesting a web page).
  • Because of firewalls and not knowing whether it will be the client or server behind them, we probably want to support both client poll-subscribe and async server push.
• Our model is predicated on having subsystem controllers that have web servers built in them. The enterprise manager has the opportunity to request information from the controllers that the subsystem developer considers relevant. This information could go down to the points or be a summary.

• How about putting async messaging to bed for a couple of months?
  • Will this increase the burden on the clients?
    • At the moment, no. Java and .NET don’t support anything like this.
  • That sounds good. Let the IT world work this issue out before we tackle it.