Some notes from the SIAM and Instrument PUCK Design Review
November 17-18, 2003

Duane Edgington

1. Comment on contents of Puck
    a. At least a portion should be language independent (header)
    b. XML for metadata in the puck, so language and implementation neutral
2. Jeff Burch: 1451
    a. 1451.1 – ID in the header for language independent access
    b. 1451.2 – header plus industry extensions
3. Discussion of Puck Header:
    a. Binary ?
    b. XML?
    c. Revision field is important (to support understanding revisions of the format)
    d. Code: Permit multiple kinds of code, which to read to be decided by platform
        1. Java class files (byte code)
        2. Java source files (so you can regenerate byte codes if the JVM is different than what the class files were compiled for)
    ii. OR
        1. C++ code
        2. etc
4. IETF-like approach to the problem of making the contents standard.
5. Key to the standard being useful and accepted:
    a. Simplify the backend
        i. Provide working code
        ii. Provide a complete solution: “buy this lump”
    b. Needs to be appear to be simple (not too complicated)
    c. Standard works with few exceptions across all the applications
6. Idea: what about a battery powered puck?
    a. Removes design constraints that the puck be very low powered
7. Question: can you program the puck in the field?
    a. Answer: yes, you can, if you need to.
    b. Answer: idea is that you will not need to ordinarily:
        i. power cycle the instrument, the puck delivers a known good state.
8. Andrew Maffei: Support for external programs that already know how to interact with the instrument?
    a. Make the puck “time out” and revert to pass through sensor mode, allowing the external program to get complete control of the sensor.
    b. Have the first two bytes (or whatever) that comes out of the puck deliver enough information so whatever is watching the stream can decide what to do
    c. Include a revision number for the format, so whatever program is watching the stream knows if it is capable of understanding that revision.
9. Jeff Burch: lessons learned from different styles of standardization efforts:
    a. Sun Microsystems approach: Java:
        i. Get a group of experts together, gather their input
        ii. Do implementation and get it out there for people to use
        iii. Stay involved in growing based on what works
            1. probably best for this project
    b. 1451.2 standard:
        i. complete and formal
        ii. still little implementation in industry, despite being out since 1997
    c. 1588 time synch standard:
        i. complete and formal
        ii. moving very well
10. Jeff Burch: think about lightweight JMS for wet side system:

    a. Publish – subscribe: best for low reliability network, normal operation
    b. Client – server: best for debug (when humans are in the loop) or high reliability network
        i. Thinks we are in the publish – subscribe case.
11. Event Response
    a. Science event response requires ability to understand the data (read the XML metadata?) on the platform
12. Discovery
    a. Need to explicitly power down a port before adding an instrument
    b. Need to explicitly issue a "new instrument" command before a new instrument is discovered
13. Security
    a. Needs review of needed functionality
    b. Looking for balance of security and operational ease
        i. Authentication
        ii. Monitoring
    c. Is firewall and VPN the answer?
14. Fault tolerance
    a. Discussed need for out-of-band communication mechanism