Asynchronous Management Architecture (AMA)

Edward Birrane
Edward.Birrane@jhuapl.edu
443-778-7423
We cannot deploy challenged internetworks until we can manage them.

- **Use cases for DTNs are emerging:**
  - Handle signal propagation delay (space and some underwater).
    - *Mostly for space and underwater scenarios*
  - Handle frequent link disruptions.
    - *Mostly for disaster and some vehicular scenarios.*
  - Handle frequent link-access disruptions.
    - *Mostly for oversubscribed/congested links.*
    - *Link removed as matter of policy/administration and not physics.*
  - **All preclude human-in-the-loop network management**
    - Nodes operate on “far side” of delayed/disrupted links.
    - Disruptions occur from attenuation, tasking, power, and pointing
    - **Network management** starts looking more like **fault management**.
      - Maintain ability to relay information from critical assets both on-board and remotely without access to direct operator intervention.
Examined uniqueness of the problem, 2011-2013

Some early pubs defining the problem as related to DTN
- E. Birrane, S. Burleigh, V. Cerf, “Defining Tolerance: Impacts of Delay and Disruption when Managing Challenged Networks,”
- E. Birrane, H. Kruse, “Delay-Tolerant Network Management: The Definition and Exchange of Infrastructure Information in High Delay Environments”

Reviewed popular engineering approaches
- Autonomous fault protection schemes
- Mobile code and scripting schemes
- Spacecraft telemetry schedules
- Deterministic rule-based expert systems

Delay-Tolerant Network Management Protocol (DTNMP) 2013
- Published to DTNRG, Initial implementation by NASA
- Utility outside of NASA network management

Renamed as Asynchronous Management Protocol (AMP) 2015
- Submitted as set of IDs to DTNWG.
- Extracted AMA as set of requirements/properties.
How Do We Manage Networks Today? Do we need a new thing?

Low-latency approaches to network management fail to scale with increasing delays and disruptions.

- Rich set of evolving capabilities
  - Simple Network Management Protocol (SNMP).
    - *Pull model of information from managed devices.*
      - Support for “traps” to push unreliable notifications of pre-defined events.
  - Network Configuration Protocol (NETCONF).
    - *XML-based, session-based remote-procedure call (RPC) interface for node configuration.*
  - Remote Network Monitoring MIB (RMON).
    - *Mechanisms for exchanging network monitoring data.*

- Poor scaling with delays, disruptions, or commanding
  - Focus on getting data to operators.
  - Less focus on in-situ response options.
  - Reliance on scripting and mobile code which is not always a deployment option.
Performance monitoring (reporting) is one of many network management requirements.

- **Parameterized Control**
  - Detection, diagnosis, reporting, correcting failures.
  - Example: Monitor-Response Autonomy
- **Configuration.**
  - Update the behavior of functions within the network remotely.
  - Update/reconfigure systems based on local state and time.
- **Administration.**
  - Apply access control lists, security settings, and other methods filtering management function by role.
- **Reporting**
  - Report network conditions to operators and other nodes in an internetwork based on local state and time.
Desirable Properties

- Intelligent Information Push
  - Round-trip pull requests difficult in some deployments

- Minimize Message Size, not Node Processing
  - Smaller messages work for everybody

- Specific Data Identification
  - Do not waste transmissions exchanging synchronizing data. For example, support associative lookups, not table key dumps followed by index queries.

- Tactical Data Definition
  - Define reports with high fidelity.

- Autonomous Operation
  - Deterministic, monitor-response systems
  - Avoid reliance on mobile code which can be problematic
Push, don’t Pull.

(extreme) Example: Collect A at high rate, Collect B, C at lower rate.

SNMP (PULL)

AMP (PUSH)
**Keep Message Sizes As Small As Practical**

Currently recommend pre-shared schemes and binary encoding.

<table>
<thead>
<tr>
<th>Fully Named ASCII Data (Good)</th>
<th>~ 60 bytes</th>
<th>EXPIRED_BUNDLE_COUNT = 50505</th>
<th>CUSTODY_REJECT_BAD_EID = 10000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully Named Binary Data (Better)</td>
<td>~ 30 bytes</td>
<td>0x11092A030102017A010150</td>
<td>50505</td>
</tr>
<tr>
<td>Summary Named Binary Data (Best)</td>
<td>~ 19 bytes</td>
<td>0x11092A030102017A010150</td>
<td>50505</td>
</tr>
</tbody>
</table>
Specific, Tactical Data Definition

- **Pre-defined sets of data**
  - BP, LTP, ICI
  - Query individual items

- **Pre-defined collects per set**
  - All BP Disposition
  - All LTP Stats
  - All ICI SDR Stats

- **How to mix/match across data sets?**
  - ExpiredBundleCount + Head Used + Small Pool Size
  - Could make 3 queries (3 sets of NAME=VALUE)
    - This is wasteful from previous slide)
  - Define new report to represent 3 values
    - 1 NAME, 3 VALUES
    - More bandwidth efficient
Application Data Model

ADMs are, notionally, a superset of MIBs. Recommend they be specified in YANG.

- **Atomic Data and Controls.**
  - Well-defined data definitions.
    - \( A = <\text{firmware-sampled value}> \)
  - Well-defined, parameterized command opcodes.
    - \( \text{SetNewTemperature}(\text{float NewTemp}, \text{uint Deadline}) \)
- **Literals and Operators.**
  - Custom constants can be defined per-ADM.
    - \( PI = 3.14159 \)
  - Special operators (unary, binary, and more) can be defined.
    - \( \text{Pow}(x,y), \text{Avg}(A,B,C,D), \text{etc…} \)
- **Computed Data.**
  - Runtime-calculated, typed data: \( CD = \text{Data OP Data} \)
    - \( CD1 = A + B; CD2 = \text{pow}(A,C1); \)
- **Collections.**
  - What pre-defined collections of data values (reports) and control sequences (macros) have been created?
ADM Example

A sample ADM for an application implementing a stack.

<table>
<thead>
<tr>
<th>Atomic Controls</th>
<th>Computed Data</th>
<th>Atomic Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>- PUSH(X)</td>
<td>- Average POPs</td>
<td>- Stack Depth</td>
</tr>
<tr>
<td>- POP(X)</td>
<td></td>
<td>- Total Items</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Total # POPs</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Literals</th>
<th>Data Collections</th>
<th>Control Collections</th>
</tr>
</thead>
<tbody>
<tr>
<td>- MAX_DEPTH = 10</td>
<td>Report 1:</td>
<td>EMPTY:</td>
</tr>
<tr>
<td></td>
<td>- Cur. Stack Depth</td>
<td>Stack Depth &gt; 0 POP(X)</td>
</tr>
<tr>
<td></td>
<td>- Total Items</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Average POPs</td>
<td></td>
</tr>
</tbody>
</table>
In a simple network, a Manager interacts with multiple Agents.
Multi-Manager Flow

Manager A  |  Agent  |  Manager B
-------+-------+-------

--- DEF(A, CD1, AD1*2) ---->  <--- DEF(B, CD2, AD2*2) (Step 1)

--- PROD(1s, CD1) ---->  <--- PROD(1s, CD2) (Step 2)

<----- RPT(CD1) ---->

<----- RPT(CD1) ---->  <----- RPT(CD2) ---->

<----- RPT(CD1) ---->  <----- RPT(CD2) ---->

<----- PROD(1s, CD1) ---->

<----- ERR(CD1 no perm.) ---->

--- DEF(1s, CD3, AD3*3) ----> (Step 5)

--- PROD(1s, CD3) ----> (Step 6)

<----- PROD(1s, CD3) ---->

<----- RPT(CD3) ---->  <----- RPT(CD3) ----> (Step 7)

<----- RPT(CD1) ---->  <----- RPT(CD1) ---->

<----- RPT(CD3) ---->  <----- RPT(CD3) ---->

<----- RPT(CD1) ---->  <----- RPT(CD2) ---->
Data fusion occurs amongst Managers in the network.
Related internet drafts

- Asynchronous Management Protocol (AMP)
  - Binary encoding
  - Uses any transport layer

- ADMs

Reference Implementations (AMP)

- Reference implementation in ION open source, this summer
Thank you!

Questions?