Asynchronous Mgmt Architecture (AMA)
From draft-birrane-dtn-ama-05

Edward Birrane
Edward.Birrane@jhuapl.edu
443-778-7423
AMA: Updates

From -04 to -05

- **Minor Terminology and Definitions Updates**
  - Mostly wordsmithing based on feedback.

- **No significant issues or limitations with the architecture.**
  - No “structural” changes to the architecture.
AMA: Introduction & Motivation

- **Document purpose**
  - Motivation, Service Definitions, Desirable Properties, Roles/Responsibilities, System Model, Logical Data Model
  - Not a prescriptive standard, informative guidance.

- **Scope**
  - Challenged networks where asynchronous operation is required.
  - Assumes naming, addressing, integrity, confidentiality, authentication, fragmentation, security, etc… already provided.
  - Does not address interface with synchronous network management.

- **Motivation**
  - Asynchronous management requires:
    - *Autonomy model, less reliance on sessions and per-operation state sync.*
  - SNMP/NETCONF don’t provide these capabilities.
    - *Work in RESTCONF might*
  - Provide a standard model to converge efforts in this area.
AMA: Service Definitions

- **Configuration**
  - Create new datum as function of other data \((C = A + B)\)
  - Ex: Create new reports \((RPT = \{A, B, C\})\)
  - Ex: Store pre-defined actions \((\text{IF } (X > 3) \text{ THEN } \text{Cmd}(\text{Params}))\)

- **Reporting**
  - Push data, don’t pull.
  - Ex: Push as a function of time \((\text{Generate report every hour})\)
  - Ex: Push as function of state \((\text{Generate report if } (X > 3))\)

- **Autonomous Parameterized Procedure Calls**
  - Manage” agent asynchronously by coding response options.
  - Allow for behavior to be customized through parameterization.
  - Ex: Update local route info based on local link analysis
  - Ex: Manage storage to enforce quotas
  - Ex: Apply or modify local security policy

- **Administration**
  - Finer grained access control for operations.
AMA: Desirable Properties

- **Intelligent Push**
  - Asynchronous operation doesn’t support round-trip pull requests.

- **Absolute Data Identification**
  - Data must be atomically identifiable
    - *Should not need multiple rounds of synchronization to figure out where data lives in an array, for example.*
    - *E.g., support associative looks-ups*

- **Custom Data Definition**
  - Define new data (variables) local to an agent.

- **Autonomous Operation**
  - Automation of pre-defined tasks, Autonomy to self-configure same.
  - Distributed operation allows for decentralized control/execution.
  - Deterministic Behavior – Ability to forensically reconstruct events.
  - Engine-Based Behavior – Ability to avoid mobile code where needed.
AMA: System Model

- **Agents**
  - Run on Managed Devices
  - Configure/Report on devices
  - Heavy autonomy and parameterized control

- **Manager(s)**
  - Collect/Fuse data from Agents
  - Configure Agent behavior
  - Open-loop control

- **ADM(s)**
  - Well-named Data and Controls
  - Superset of MIB
  - Move to describe them in YANG
  - Preconfiguration reduces msg size
AMA: Roles/Responsibilities

**Agent Responsibilities**
- **Application Support** – Manage local applications/protocols.
- **Local Data Collection** – Collect and/or calculate new values.
- **Autonomy Control** – Apply time/state based response options.
- **User Data Definition** – Store/remember user-defined data variables.
- **Autonomous Reporting** – Push reports based on time/state.
- **Consolidate Messages** – Where possible, reduce overhead.
- **Regional Proxy** – Collect from other nodes in a region.

**Manager Responsibilities**
- **Agent Capabilities Mapping** – Common picture of agent abilities.
- **Data Collection** – Receive data from multiple agents.
- **Custom Definitions** – Send user-defined data to agents.
- **Data Translation** – Interface with other network management systems.
- **Data Fusion** – Generate new data from received data.
AMA: Logical Data Model (ADM)

**“Atomic” Elements**
- Solely defined by their ADM.
- **EDDs**: collected by agents.
- **Literals**: useful constants.
- **Ops**: opcodes for math functions.
- **Ctrls**: opcodes for agent behavior.

**“Variable” Elements**
- Defined by ADM or by User
- ADM definitions are immutable.
- **Vars**: strong-typed variables, including a type for “expression”.
- **Macro**: Ordered set ofCtrls.
- **Rpts**: Ordered sets of data
- **Rules**: Time or State based autonomy.

An ADM defines 8 types of data for each application/protocol managed in the AMA.

![Diagram of Application Data Model](attachment://diagram.png)
AMA Control and Data Flows

Figure 1
AMA Serialized Management Control
Flow

Manager

Agent A

Agent B

PROD(1s, EDD1) --> PROD(1s, EDD1) -->

RPT(EDD1) --> RPT(EDD1)

RPT(EDD1) --> RPT(EDD1)

RPT(EDD1) --> RPT(EDD1)

RPT(EDD1) --> RPT(EDD1)

(1)

(2)
Multiplexed Management Control

Flow

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

---DEF(A,V1,EDD1*2) --> <---DEF(B, V2, EDD2*2) -- (1)

---PROD(1s, V1) -----> <---PROD(1s, V2) -----> (2)

<-------RPT(V1) -----> <-------RPT(V2) ----->

<-------RPT(V1) -----> <-------RPT(V2) ----->

<-------RPT(V1) -----> <-------RPT(V2) ----->

<---PROD(1s, V1)----- (4)

---ERR(V1 no perm.)-->

---DEF(*,V3,EDD3*3)--->

---PROD(1s, V3)----- (5)

<-------RPT(V3) -----> <-------RPT(V3) -----> (7)

<-------RPT(V1) -----> <-------RPT(V2) ----->

<-------RPT(V3) -----> <-------RPT(V3) ----->

<-------RPT(V1) -----> <-------RPT(V2) ----->
```

---ERR(V2 no perm.)-->

---DEF(A,V1,EDD1*2) --> <---DEF(B, V2, EDD2*2) -- (1)

---PROD(1s, V1) -----> <---PROD(1s, V2) -----> (2)

<-------RPT(V1) -----> <-------RPT(V2) ----->

<-------RPT(V1) -----> <-------RPT(V2) ----->

<---PROD(1s, V1)----- (4)

---ERR(V1 no perm.)-->

---DEF(*,V3,EDD3*3)--->

---PROD(1s, V3)----- (5)

<-------RPT(V3) -----> <-------RPT(V3) -----> (7)

<-------RPT(V1) -----> <-------RPT(V2) ----->

<-------RPT(V3) -----> <-------RPT(V3) ----->

<-------RPT(V1) -----> <-------RPT(V2) ----->

<-------RPT(V2) ----->
Data Fusion Control Flow

Manager A

--DEF(A, V0, EDD1+AD2)-->
--PROD(EDD1&AD2, V0)-->

Manager B

--PROD(1s, EDD1)-->
--PROD(1s, EDD2)-->

Agent B

--RPT(EDD1)----
--RPT(EDD2)----

Agent C

--RPT(A, V0)------
Thank you!

Questions?