M3DA

An open-source protocol for efficient M2M communications

SIERRA WIRELESS™

Goals

Compact *… because in the wireless world, data overhead costs money*

- Efficient in the transport of binary M2M data

Interoperable *… because the M2M communication chain is heterogeneous*

- Language-independent
- Tolerant to data schema changes
- Agnostic to transport layer (TCP, HTTP, SMS, …)

Secure *… because security is #1 concern for M2M adopters*

- Ensure the confidentiality of customer data

Open *… because vendor lock-in hinders M2M adoption*

- Open specification
- Open-source, royalty-free, implementations

Layered Architecture

Application Layer – `M3DA::Message`

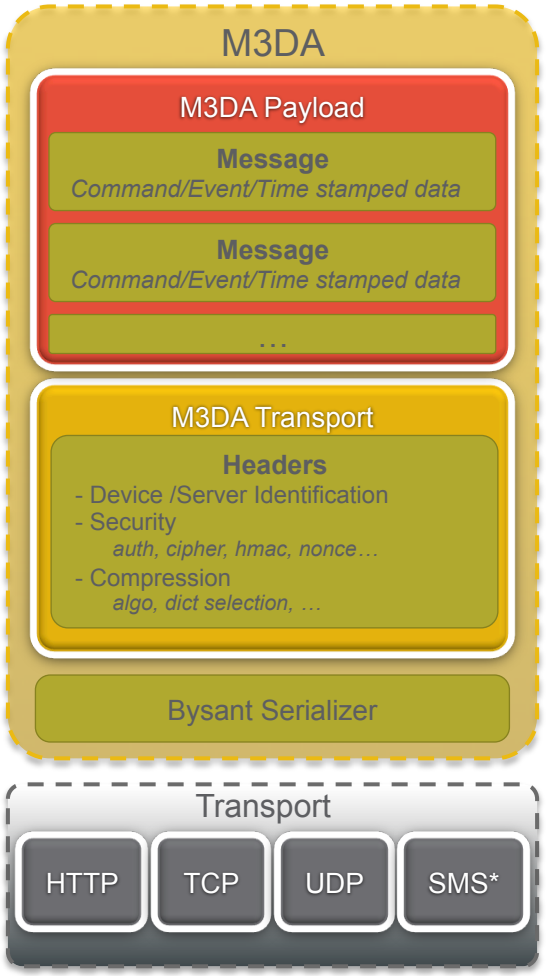
- Specify access to the tree-oriented data model

Transport Layer – `M3DA::Envelope`

- Enables authentication, encryption, compression, …

Serialization Layer – Bysant

- Provides an efficient binary serialization based on contextual information



Application Layer

The Application layer provides two objects in order to perform tree access

- `M3DA::Message`
- `M3DA::Response`

Composite structures are materialized with nodes and leaves in the data tree.

- *Example:* a GPS position can be represented with a node that has 4 leaves: latitude, longitude, speed and altitude.

M3DA::Message

`M3DA::Message` is mostly a map:

- key = path in the tree
- values = value(s) for that leaf

`M3DA::Message` has a ticket ID that can be used for acknowledging delivery

Example:

Ticket ID: `23424` ← if ≠ 0 ➔ request acknowledge

Path: `my.car.position`

Body:

latitude	10.0
longitude	32.5
speed	33
altitude	433

see Serialization section

Commands, Events, Acks

Commands and Events are higher level concepts that are instantiated by using conventions on the tree.

- Commands are located into ‘commands’ sub branch

The protocol uses a ticket ID field to ensure a correct acknowledgement mechanism

Example: Send a reboot command with a delay of 42 seconds

Ticket ID: `3455`

Path: `@sys.commands.Reboot`

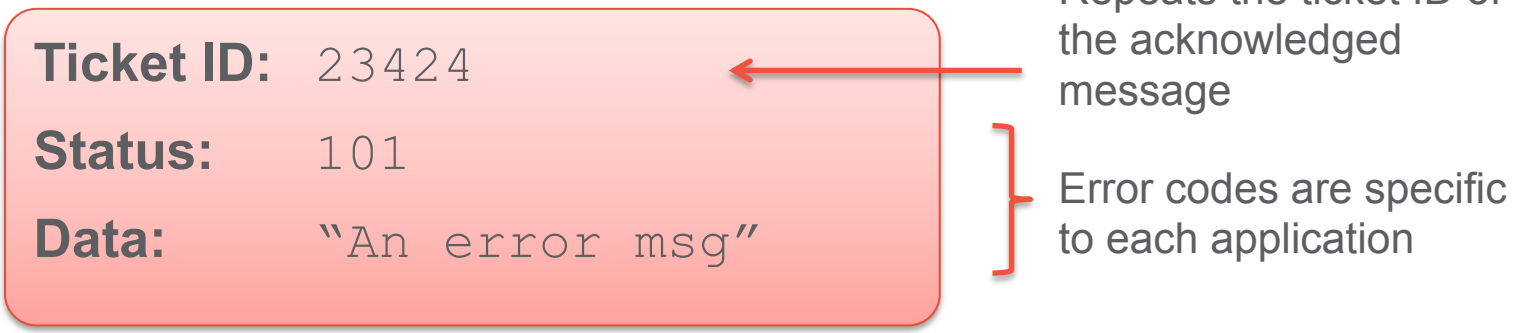
Body:

`delay`	`42`

M3DA::Response

The `M3DA::Response` repeats the message ticket ID, a status code and an optional data payload.

Example:



Transport Layer

`M3DA::Envelope` is used to transport the application messages.

This object provides header and footer

- identify the device
- add compression mechanism (zip)
- add checksums, authentication, encryption

The envelope itself can be put in any other transport layer

- HTTP
- TCP
- UDP
- SMS
- ...

Bysant Serialization

Object-oriented

Provides basic classes

- Numbers (Integers, Floating point numbers, Boolean)
- Strings (Binary string)
- Container objects (List, Map)

User-defined objects

- Allows to structure protocol objects (Message, Responses, DeltaVector, …)

Serialization is optimized for M2M data

- Small numbers take less bytes
  - *5* → 1 byte, *2040* → 2 bytes
- String size header are adaptative
  - 1 byte for a 28-byte long string, but 3 bytes for 66k-byte long string.

SIERRA WIRELESS™

Lists optimization

Specialized lists for M2M data collections

- `DeltasVector` – Data is serialized using deltas from the previous value
- `QuasiPeriodicVector` – Data is serialized as the difference to an affine function. This encode very well periodic values.

Deltas or shifts are usually small integers, hence bandwidth optimization is maximal

- A factor can be applied to minimize the deltas even more (by reducing the precision of the values)

Security

Inspired by existing security models (OMA-DM)

Reviewed and certified by a security expert group

Provides an over the air password auto registration mechanism

Is protected against replay attacks by using a randomly generated nonce for each single message.

Help reduce DoS attacks by minimizing the false authentication cost.

Security

Password auto registration

- Using Diffie-Hellman algorithm to create a temporary secure link and exchange the randomly generated device-server password.
- Single time operation in the device lifetime

Authentication

- Based on HMAC algorithm, inspired by OMA-DM authentication.

$$HMAC\ (K,m) = H\ ((K \oplus opad) \ // \ H\ ((K \oplus ipad) \ // \ m))$$

- Using a unique password for a single device-server pair

Encryption

- Based on AES cipher, configurable from 128 to 256 key size
- Using CBC or CTR block cipher modes of operation in order to prevent pattern matching attacks.

Open-source

M3DA is an open-source, royalty-free protocol

Specification to be released in open-source in Q1 2013

Reference implementation of the protocol will be available in the Eclipse Mihini project (EPL license)