Network Working Group   T. Burbridge
Internet-Draft   P. Eardley
Intended status: Standards Track   BT
Expires: August 26, 2017   M. Bagnulo
                                     Universidad Carlos III de Madrid
                                                    J. Schoenwaelder
                                             Jacobs University Bremen
                                                   February 22, 2017

      Information Model for Large-Scale Measurement Platforms (LMAP)
              draft-ietf-lmap-information-model-17

Abstract

   This Information Model applies to the Measurement Agent within a
   Large-Scale Measurement Platform. As such it outlines the
   information that is (pre-)configured on the Measurement Agent or
   exists in communications with a Controller or Collector within an
   LMAP framework. The purpose of such an Information Model is to
   provide a protocol and device independent view of the Measurement
   Agent that can be implemented via one or more Control and Report
   protocols.

Requirements Language

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
   document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

   A large-scale measurement platform is a collection of components that
   work in a coordinated fashion to perform measurements from a large
   number of vantage points. The main components of a large-scale
   measurement platform are the Measurement Agents (hereafter MAs), the
   Controller(s) and the Collector(s).

   The MAs are the elements actually performing the measurements. The
   MAs are controlled by exactly one Controller at a time and the
   Collectors gather the results generated by the MAs. In a nutshell,
   the normal operation of a large-scale measurement platform starts

with the Controller instructing a set of one or more MAs to perform a
set of one or more Measurement Tasks at a certain point in time. The
MAs execute the instructions from a Controller, and once they have
done so, they report the results of the measurements to one or more
Collectors. The overall framework for a large-scale measurement
platform as used in this document is described in detail in
[RFC7594].

A large-scale measurement platform involves basically three types of
protocols, namely, a Control protocol (or protocols) between a
Controller and the MAs, a Report protocol (or protocols) between the
MAs and the Collector(s) and several measurement protocols between
the MAs and Measurement Peers (MPs), used to actually perform the
measurements. In addition some information is required to be
configured on the MA prior to any communication with a Controller.

This document defines the information model for both Control and the
Report protocols along with pre-configuration information that is
required on the MA before communicating with the Controller, broadly
named as the LMAP Information Model. The measurement protocols are
out of the scope of this document.

As defined in [RFC3444], the LMAP Information Model defines the
concepts involved in a large-scale measurement platform at a high
level of abstraction, independent of any specific implementation or
actual protocol used to exchange the information. It is expected
that the proposed information model can be used with different
protocols in different measurement platform architectures and across
different types of MA devices (e.g., home gateway, smartphone, PC,
router). A YANG data model implementing the information model can be
found in [I-D.ietf-lmap-yang].

The definition of an Information Model serves a number of purposes:

1. To guide the standardisation of one or more Control and Report
    protocols and data models

2. To enable high-level inter-operability between different Control
    and Report protocols by facilitating translation between their
    respective data models such that a Controller could instruct sub-
    populations of MAs using different protocols

3. To form agreement of what information needs to be held by an MA
    and passed over the Control and Report interfaces and support the
    functionality described in the LMAP framework

4. To enable existing protocols and data models to be assessed for
    their suitability as part of a large-scale measurement system

2. **Notation**

   This document uses a programming language-like notation to define the
   properties of the objects of the information model. An optional
   property is enclosed by square brackets, [ ], and a list property is
   indicated by two numbers in angle brackets, <m..n>, where m indicates
   the minimal number of values, and n is the maximum. The symbol * for
   n means no upper bound.

   The object definitions use a couple of base types that are defined as
   follows:

   int   A type representing signed or unsigned integer numbers.
              This information model does not define a precision nor
              does it make a distinction between signed and unsigned
              number ranges. This type is also used to represent
              enumerations.

   boolean   A type representing a boolean value.

   string   A type representing a human-readable string consisting of
              a (possibly restricted) subset of Unicode and ISO/IEC
              10646 [ISO.10646] characters.

   datetime   A type representing a date and time using the Gregorian
              calendar. The datetime format MUST conform to RFC 3339
              [RFC3339].

   uuid   A type representing Universally Unique IDentifier (UUID(
              as defined in RFC 4122 [RFC4122]. The UUID values are
              expected to be unique within an installation of a large-
              scale measurement system.

   uri   A type representing a Uniform Resource Identifier as
              defined in STD 66 [RFC3986].

   ip-address A type representing an IP address. This type supports
              both IPv4 and IPv6 addresses.

   counter   A non-negative integer that monotonically increases.
              Counters may have discontinuities and they are not
              expected to persist across restarts.

   credentials An opaque type representing credentials needed by a
              cryptographic mechanism to secure communication. Data
              models must expand this opaque type as needed and
              required by the security protocols utilized.

   data   An opaque type representing data obtained from
               measurements.

   Names of objects are generally assumed to be unique within an
   implementation.

3. LMAP Information Model

   The information described herein relates to the information stored,
   received or transmitted by a Measurement Agent as described within
   the LMAP framework [RFC7594]. As such, some subsets of this
   information model are applicable to the measurement Controller,
   Collector and any device management system that pre-configures the
   Measurement Agent. The information described in these models will be
   transmitted by protocols using interfaces between the Measurement
   Agent and such systems according to a Data Model.

   For clarity the information model is divided into six sections:

   1. Pre-Configuration Information. Information pre-configured on the
       Measurement Agent prior to any communication with other
       components of the LMAP architecture (i.e., the Controller,
       Collector and Measurement Peers), specifically detailing how to
       communicate with a Controller and whether the device is enabled
       to participate as an MA.

   2. Configuration Information. Update of the pre-configuration
       information during the registration of the MA or subsequent
       communication with the Controller, along with the configuration
       of further parameters about the MA (rather than the Measurement
       Tasks it should perform) that were not mandatory for the initial
       communication between the MA and a Controller.

   3. Instruction Information. Information that is received by the MA
       from the Controller pertaining to the Measurement Tasks that
       should be executed. This includes the task execution Schedules
       (other than the Controller communication Schedule supplied as
       (pre)configuration information) and related information such as
       the Task Configuration, communication Channels to Collectors and
       schedule Event and Timing information. It also includes Task
       Suppression information that is used to over-ride normal Task
       execution.

   4. Logging Information. Information transmitted from the MA to the
       Controller detailing the results of any configuration operations
       along with error and status information from the operation of the
       MA.

5. Capability and Status Information. Information on the general
    status and capabilities of the MA. For example, the set of
    measurements that are supported on the device.

6. Reporting Information. Information transmitted from the MA to
    one or more Collectors including measurement results and the
    context in which they were conducted.

In addition the MA may hold further information not described herein,
and which may be optionally transferred to or from other systems
including the Controller and Collector. One example of information
in this category is subscriber or line information that may be
extracted by a task and reported by the MA in the reporting
communication to a Collector.

It should also be noted that the MA may be in communication with
other management systems which may be responsible for configuring and
retrieving information from the MA device. Such systems, where
available, can perform an important role in transferring the pre-
configuration information to the MA or enabling/disabling the
measurement functionality of the MA.

The Information Model is divided into sub-sections for a number of
reasons. Firstly the grouping of information facilitates reader
understanding. Secondly, the particular groupings chosen are
expected to map to different protocols or different transmissions
within those protocols.

The granularity of data transmitted in each operation of the Control
and Report Protocols is not dictated by the Information Model. For
example, the Instruction object may be delivered in a single
operation. Alternatively, Schedules and Task Configurations may be
separated or even each Schedule/Task Configuration may be delivered
individually. Similarly the Information Model does not dictate
whether data is read, write, or read/write. For example, some
Control Protocols may have the ability to read back Configuration and
Instruction information which have been previously set on the MA.
Lastly, while some protocols may simply overwrite information (for
example refreshing the entire Instruction Information), other
protocols may have the ability to update or delete selected items of
information.

The information in these six sections is captured by a number of
common information objects. These objects are also described later
in this document and comprise of:

a. Schedules. A set of Schedules tells the MA to execute Actions.
    An Action of a Schedule leads to the execution of a Task.

Without a Schedule no Task (including measurements or reporting
or communicating with the Controller) is ever executed.
Schedules are used within the Instruction to specify what tasks
should be performed, when, and how to direct their results. A
Schedule is also used within the pre-Configuration and
Configuration information in order to execute the Task or Tasks
required to communicate with the Controller. A specific Schedule
can only be active once. Attempts to start a Schedule while the
same Schedule is still running will fail.

b. Channels. A set of Channel objects are used to communicate with
    a number of endpoints (i.e., the Controller and Collectors).
    Each Channel object contains the information required for the
    communication with a single endpoint such as the target location
    and security details.

c. Task Configurations. A set of Task Configurations is used to
    configure the Tasks that are run by the MA. This includes the
    registry entries for the Task and any configuration parameters,
    represented as Task Options. Task Configurations are referenced
    from a Schedule in order to specify what Tasks the MA should
    execute.

d. Events. A set of Event objects that can be referenced from the
    Schedules. Each Schedule always references exactly one Event
    object that determines when the schedule is executed. An Event
    object specifies either a singleton or series of events that
    indicate when Tasks should be executed. A commonly used kind of
    Event objects are Timing objects. For Event objects specifying a
    series of events, it is generally a good idea to configure an end
    time and to refresh the end time as needed to ensure that MAs
    that loose connectivity to their controller do not continue
    executing Schedules forever.

Figure 1 illustrates the structure in which these common information
objects are referenced. The references are achieved by each object
(Task Configuration, Event) being given a short textual name that is
used by other objects. The objects shown in parenthesis are part of
the internal object structure of a Schedule. Channels are not shown
in the diagram since they are only used as an option by selected Task
Configurations but are similarly referenced using a short text name.

```
        Schedule
           |-- triggered by --> Event
           |
           |-- executes --> Action 1
           |                    |-- using --> Task Configuration
           |                    |
           |                    `-- feeding to --> Destination Schedule
           :
           :
           `-- executes --> Action N
                                |-- using --> Task Configuration
                                |
                                `-- feeding to --> Destination Schedule
```
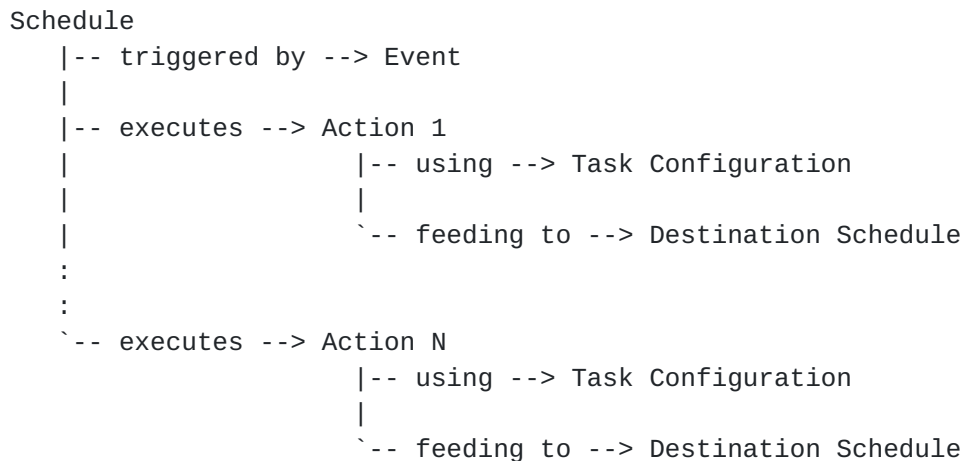
        Figure 1: Relationship between Schedules, Events, Actions, Task
                 Configurations, and Destination Schedules

   The primary function of an MA is to execute Schedules. A Schedule,
   which is triggered by an Event, executes a number of Actions. An
   Action refers to a Configured Task and it may feed results to a
   Destination Schedule. Both, Actions and Configured Tasks can provide
   parameters, represented as Action Options and Task Options.

   Tasks can implement a variety of different functions. While in terms
   of the Information Model, all Tasks have the same structure, it can
   help conceptually to think of different Task categories:

   1. Measurement Tasks measure some aspect of network performance or
       traffic. They may also capture contextual information from the
       MA device or network interfaces such as the device type or
       interface speed.

   2. Data Transfer Tasks support the communication with a Controller
       and Collectors:

       A. Reporting Tasks report the results of Measurement Tasks to
           Collectors

       B. Control Task(s) implement the Control Protocol and
           communicate with the Controller.

   3. Data Analysis Tasks can exist to analyse data from other
       Measurement Tasks locally on the MA

   4. Data Management Tasks may exist to clean-up, filter or compress
       data on the MA such as Measurement Task results

Figure 1 indicates that Actions can produce data that is fed into
Destination Schedules. This can by used by Actions implementing
Measurement Tasks to feed measurement results to a Schedule that
triggers Actions implementing Reporting Tasks. Data fed to a
Destination Schedule is consumed by the first Action of the
Destination Schedule if the Destination Schedule is using sequential
or pipelined execution mode and it is consumed by all Actions of the
Destination Schedule if the Destination Schedule is using parallel
execution mode.

3.1. Pre-Configuration Information

This information is the minimal information that needs to be pre-
configured to the MA in order for it to successfully communicate with
a Controller during the registration process. Some of the Pre-
Configuration Information elements are repeated in the Configuration
Information in order to allow an LMAP Controller to update these
items. The pre-configuration information also contains some elements
that are not under the control of the LMAP framework (such as the
device identifier and device security credentials).

This Pre-Configuration Information needs to include a URL of the
initial Controller from where configuration information can be
communicated along with the security information required for the
communication including the certificate of the Controller (or the
certificate of the Certification Authority which was used to issue
the certificate for the Controller). All this is expressed as a
Channel. While multiple Channels may be provided in the Pre-
Configuration Information they must all be associated with a single
Controller (e.g., over different interfaces or network protocols).

Where the MA pulls information from the Controller, the Pre-
Configuration Information also needs to contain the timing of the
communication with the Controller as well as the nature of the
communication itself (such as the protocol and data to be
transferred). The timing is represented as an Event that invokes a
Schedule that executes the Task(s) responsible for communication with
the Controller. It is this Task (or Tasks) that implement the
Control protocol between the MA and the Controller and utilises the
Channel information. The Task(s) may take additional parameters, as
defined by a Task Configuration.

Even where information is pushed to the MA from the Controller
(rather than pulled by the MA), a Schedule still needs to be
supplied. In this case the Schedule will simply execute a Controller
listener Task when the MA is started. A Channel is still required
for the MA to establish secure communication with the Controller.

It can be seen that these Channels, Schedules and Task Configurations
for the initial MA-Controller communication are no different in terms
of the Information Model to any other Channel, Schedule or Task
Configuration that might execute a Measurement Task or report the
measurement results (as described later).

The MA may be pre-configured with an MA ID, or may use a Device ID in
the first Controller contact before it is assigned an MA ID. The
Device ID may be a MAC address or some other device identifier
expressed as a URI. If the MA ID is not provided at this stage then
it must be provided by the Controller during Configuration.

3.1.1. Definition of ma-preconfig-obj

```
object {
    [uuid                ma-preconfig-agent-id;]
     ma-task-obj         ma-preconfig-control-tasks<1..*>;
     ma-channel-obj      ma-preconfig-control-channels<1..*>;
     ma-schedule-obj     ma-preconfig-control-schedules<1..*>;
    [uri                 ma-preconfig-device-id;]
     credentials         ma-preconfig-credentials;
} ma-preconfig-obj;
```

The ma-preconfig-obj describes information that needs to be available
to the MA in order to bootstrap communication with a Controller. The
ma-preconfig-obj consists of the following elements:

ma-preconfig-agent-id: An optional uuid uniquely identifying
                                the measurement agent.

ma-preconfig-control-tasks: An unordered set of task objects.

ma-preconfig-control-channels: An unordered set of channel objects.

ma-preconfig-control-schedules: An unordered set of scheduling
                                objects.

ma-preconfig-device-id: An optional identifier for the
                                device.

ma-preconfig-credentials: The security credentials used by the
                                measurement agent.

3.2. Configuration Information

During registration or at any later point at which the MA contacts
the Controller (or vice-versa), the choice of Controller, details for
the timing of communication with the Controller or parameters for the

communication Task(s) can be changed (as captured by the Channels,
Schedules and Task Configurations objects). For example the pre-
configured Controller (specified as a Channel or Channels) may be
over-ridden with a specific Controller that is more appropriate to
the MA device type, location or characteristics of the network (e.g.,
access technology type or broadband product). The initial
communication Schedule may be over-ridden with one more relevant to
routine communications between the MA and the Controller.

While some Control protocols may only use a single Schedule, other
protocols may use several Schedules (and related data transfer Tasks)
to update the Configuration Information, transfer the Instruction
Information, transfer Capability and Status Information and send
other information to the Controller such as log or error
notifications. Multiple Channels may be used to communicate with the
same Controller over multiple interfaces (e.g., to send logging
information over a different network).

In addition the MA will be given further items of information that
relate specifically to the MA rather than the measurements it is to
conduct or how to report results. The assignment of an ID to the MA
is mandatory. If the MA Agent ID was not optionally provided during
the pre-configuration then one must be provided by the Controller
during Configuration. Optionally a Group ID may also be given which
identifies a group of interest to which that MA belongs. For example
the group could represent an ISP, broadband product, technology,
market classification, geographic region, or a combination of
multiple such characteristics. Additional flags control whether the
MA ID or the Group ID are included in Reports. The reporting of a
Group ID without the MA ID may allow the MA to remain anonymous,
which may be particularly useful to prevent tracking of mobile MA
devices.

Optionally an MA can also be configured to stop executing any
Instruction Schedule if the Controller is unreachable. This can be
used as a fail-safe to stop Measurement and other Tasks being
conducted when there is doubt that the Instruction Information is
still valid. This is simply represented as a time window in seconds
since the last communication with the Controller after which an Event
is generated that can trigger the suspension of Instruction
Schedules. The appropriate value of the time window will depend on
the specified communication Schedule with the Controller and the
duration for which the system is willing to tolerate continued
operation with potentially stale Instruction Information.

While Pre-Configuration Information is persistent upon device reset
or power cycle, the persistency of the Configuration Information may
be device dependent. Some devices may revert back to their pre-

configuration state upon reboot or factory reset, while other devices
may store all Configuration and Instruction information in persistent
storage. A Controller can check whether an MA has the latest
Configuration and Instruction information by examining the Capability
and Status information for the MA.

3.2.1. Definition of ma-config-obj

```
object {
    uuid                ma-config-agent-id;
    ma-task-obj         ma-config-control-tasks<1..*>;
    ma-channel-obj      ma-config-control-channels<1..*>;
    ma-schedule-obj     ma-config-control-schedules<1..*>;
    credentials         ma-config-credentials;
   [string              ma-config-group-id;]
   [string              ma-config-measurement-point;]
   [boolean             ma-config-report-agent-id;]
   [boolean             ma-config-report-group-id;]
   [boolean             ma-config-report-measurement-point;]
   [int                 ma-config-controller-timeout;]
} ma-config-obj;
```

The ma-config-obj consists of the following elements:

ma-config-agent-id: A uuid uniquely identifying the
                                 measurement agent.

ma-config-control-tasks: An unordered set of task objects.

ma-config-control-channels: An unordered set of channel
                                 objects.

ma-config-control-schedules: An unordered set of scheduling
                                 objects.

ma-config-credentials: The security credentials used by
                                 the measurement agent.

ma-config-group-id: An optional identifier of the
                                 group of measurement agents this
                                 measurement agent belongs to.

ma-config-measurement-point: An optional identifier for the
                                 measurement point indicating
                                 where the measurement agent is
                                 located on a path (see [RFC7398]
                                 for further details).

   ma-config-report-agent-id: An optional flag indicating
                                       whether the agent identifier (ma-
                                       config-agent-id) is included in
                                       reports. The default value is
                                       true.

   ma-config-report-group-id: An optional flag indicating
                                       whether the group identifier (ma-
                                       config-group-id) is included in
                                       reports. The default value is
                                       false.

   ma-config-report-measurement-point: An optional flag indicating
                                       whether the measurement point
                                       (ma-config-measurement-point)
                                       should be included in reports.
                                       The default value is false.

   ma-config-controller-timeout: A timer is started after each
                                       successful contact with a
                                       controller. When the timer
                                       reaches the controller-timeout
                                       (measured in seconds), an event
                                       is raised indicating that
                                       connectivity to the controller
                                       has been lost (see ma-controller-
                                       lost-obj).

3.3. Instruction Information

   The Instruction information model has four sub-elements:

   1. Instruction Task Configurations

   2. Report Channels

   3. Instruction Schedules

   4. Suppression

   The Instruction supports the execution of all Tasks on the MA except
   those that deal with communication with the Controller (specified in
   (pre-)configuration information). The Tasks are configured in
   Instruction Task Configurations and included by reference in the
   Actions of Instruction Schedules that specify when to execute them.
   The results can be communicated to other Schedules or a Task may
   implement a Reporting Protocol and communicate results over Report
   Channels. Suppression is used to temporarily stop the execution of

new Tasks as specified by the Instruction Schedules (and optionally
to stop ongoing Tasks).

A Task Configuration is used to configure the mandatory and optional
parameters of a Task. It also serves to instruct the MA about the
Task including the ability to resolve the Task to an executable and
specifying the schema for the Task parameters.

A Report Channel defines how to communicate with a single remote
system specified by a URL. A Report Channel is used to send results
to a single Collector but is no different in terms of the Information
Model to the Control Channel used to transfer information between the
MA and the Controller. Several Report Channels can be defined to
enable results to be split or duplicated across different
destinations. A single Channel can be used by multiple (reporting)
Task Configurations to transfer data to the same Collector. A single
Reporting Task Configuration can also be included in multiple
Schedules. E.g., a single Collector may receive data at three
different cycle rates, one Schedule reporting hourly, another
reporting daily and a third specifying that results should be sent
immediately for on-demand measurement tasks. Alternatively multiple
Report Channels can be used to send Measurement Task results to
different Collectors. The details of the Channel element is
described later as it is common to several objects.

Instruction Schedules specify which Actions to execute according to a
given triggering Event. An Action extends a Configured Task with
additional specific parameters. An Event can trigger the execution
of a single Action or it can trigger a repeated series of Actions.
The Schedule also specifies how to link Tasks output data to other
Schedules.

Measurement Suppression information is used to over-ride the
Instruction Schedule and temporarily stop measurements or other Tasks
from running on the MA for a defined or indefinite period. While
conceptually measurements can be stopped by simply removing them from
the Measurement Schedule, splitting out separate information on
Measurement Suppression allows this information to be updated on the
MA on a different timing cycle or protocol implementation to the
Measurement Schedule. It is also considered that it will be easier
for a human operator to implement a temporary explicit suppression
rather than having to move to a reduced Schedule and then roll-back
at a later time.

It should be noted that control schedules and tasks cannot be
suppressed as evidenced by the lack of suppression information in the
Configuration. The control schedule must only reference tasks listed
as control tasks (i.e., within the Configuration information).

A single Suppression object is able to enable/disable a set of
Instruction Tasks that are tagged for suppression. This enables fine
grained control on which Tasks are suppressed. Suppression of both
matching Actions and Measurement Schedules is supported. Support for
disabling specific Actions allows malfunctioning or mis-configured
Tasks or Actions that have an impact on a particular part of the
network infrastructure (e.g., a particular Measurement Peer) to be
targeted. Support for disabling specific Schedules allows for
particularly heavy cycles or sets of less essential Measurement Tasks
to be suppressed quickly and effectively. Note that Suppression has
no effect on either Controller Tasks or Controller Schedules.

Suppression stops new Tasks from executing. In addition, the
Suppression information also supports an additional Boolean that is
used to select whether on-going tasks are also to be terminated.

Unsuppression is achieved through either overwriting the Measurement
Suppression information (e.g., changing 'enabled' to False) or
through the use of an End time such that the Measurement Suppression
will no longer be in effect beyond this time.

The goal when defining these four different elements is to allow each
part of the information model to change without affecting the other
three elements. For example it is envisaged that the Report Channels
and the set of Task Configurations will be relatively static. The
Instruction Schedule, on the other hand, is likely to be more
dynamic, as the measurement panel and test frequency are changed for
various business goals. Another example is that measurements can be
suppressed with a Suppression command without removing the existing
Instruction Schedules that would continue to apply after the
Suppression expires or is removed. In terms of the Controller-MA
communication this can reduce the data overhead. It also encourages
the re-use of the same standard Task Configurations and Reporting
Channels to help ensure consistency and reduce errors.

3.3.1. Definition of ma-instruction-obj

```
object {
    ma-task-obj        ma-instruction-tasks<0..*>;
    ma-channel-obj     ma-instruction-channels<0..*>;
    ma-schedule-obj    ma-instruction-schedules<0..*>;
   [ma-suppression-obj  ma-instruction-suppressions<0..*>;]
} ma-instruction-obj;
```

An ma-instruction-obj consists of the following elements:

ma-instruction-tasks: A possibly empty unordered set of task
                             objects.

      ma-instruction-channels: A possibly empty unordered set of
                                    channel objects.

      ma-instruction-schedules: A possibly empty unordered set of
                                    schedule objects.

   ma-instruction-suppressions: An optional possibly empty unordered
                                 set of suppression objects.

3.3.2. Definition of ma-suppression-obj

```
   object {
        string              ma-suppression-name;
       [ma-event-obj        ma-suppression-start;]
       [ma-event-obj        ma-suppression-end;]
       [string              ma-suppression-match<0..*>;]
       [boolean             ma-suppression-stop-running;]
   } ma-suppression-obj;
```

   The ma-suppression-obj controls the suppression of schedules or
   actions and consists of the following elements:

   ma-suppression-name: A name uniquely identifying a
                                 suppression.

   ma-suppression-start: The optional event indicating when
                                 suppression starts. If not present,
                                 the suppression starts immediately,
                                 i.e., as if the value would be
                                 'immediate'.

   ma-suppression-end: The optional event indicating when
                                 suppression ends. If not present, the
                                 suppression does not have a defined
                                 end, i.e., the suppression remains for
                                 an indefinite period of time.

   ma-suppression-match: An optional and possibly empty
                                 unordered set of match patterns. The
                                 suppression will apply to all schedules
                                 (and their actions) that have a
                                 matching value in their ma-schedule-
                                 suppression-tags and all actions that
                                 have a matching value in their ma-
                                 action-suppression-tags. Pattern
                                 matching is done using glob style
                                 pattern (see below).

   ma-suppression-stop-running: An optional boolean indicating whether
                                 suppression will stop any running
                                 matching schedules or actions. The
                                 default value for this boolean is
                                 false.

   Glob style pattern matching is following POSIX.2 fnmatch() [POSIX.2]
   without special treatment of file paths:

              *   matches a sequence of characters
              ?   matches a single character
              [seq]   matches any character in seq
              [!seq]   matches any character not in seq

   A backslash followed by a character matches the following character.
   In particular:

              *   matches *
              \?   matches ?
              \\   matches \

   A sequence seq may be a sequence of characters (e.g., [abc] or a
   range of characters (e.g., [a-c]).

3.4. Logging Information

   The MA may report on the success or failure of Configuration or
   Instruction communications from the Controller. In addition further
   operational logs may be produced during the operation of the MA and
   updates to capabilities may also be reported. Reporting this
   information is achieved in exactly the same manner as scheduling any
   other Task. We make no distinction between a Measurement Task
   conducting an active or passive network measurement and one which
   solely retrieves static or dynamic information from the MA such as
   capabilities or logging information. One or more logging tasks can
   be programmed or configured to capture subsets of the Logging
   Information. These logging tasks are then executed by Schedules
   which also specify that the resultant data is to be transferred over
   the Controller Channels.

   The type of Logging Information will fall into three different
   categories:

   1. Success/failure/warning messages in response to information
       updates from the Controller. Failure messages could be produced
       due to some inability to receive or parse the Controller
       communication, or if the MA is not able to act as instructed.
       For example:

   * "Measurement Schedules updated OK"

   * "Unable to parse JSON"

   * "Missing mandatory element: Measurement Timing"

   * "'Start' does not conform to schema - expected datetime"

   * "Date specified is in the past"

   * "'Hour' must be in the range 1..24"

   * "Schedule A refers to non-existent Measurement Task
      Configuration"

   * "Measurement Task Configuration X registry entry Y not found"

   * "Updated Measurement Task Configurations do not include M used
      by Measurement Schedule N"

2. Operational updates from the MA. For example:

   * "Out of memory: cannot record result"

   * "Collector 'collector.example.com' not responding"

   * "Unexpected restart"

   * "Suppression timeout"

   * "Failed to execute Measurement Task Configuration H"

3. Status updates from the MA. For example:

   * "Device interface added: eth3"

   * "Supported measurements updated"

   * "New IP address on eth0: xxx.xxx.xxx.xxx"

   This Information Model document does not detail the precise format of
   logging information since it is to a large extent protocol and MA
   specific. However, some common information can be identified.

3.4.1. Definition of ma-log-obj

```
   object {
        uuid                ma-log-agent-id;
        datetime            ma-log-event-time;
        int                 ma-log-code;
        string              ma-log-description;
   } ma-log-obj;
```

   The ma-log-obj models the generic aspects of a logging object and
   consists of the following elements:

   ma-log-agent-id: A uuid uniquely identifying the measurement
                              agent.

   ma-log-event-time: The date and time of the event reported in
                              the logging object.

   ma-log-code: A machine readable code describing the
                              event.

   ma-log-description: A human readable description of the event.

3.5. Capability and Status Information

   The MA will hold Capability Information that can be retrieved by a
   Controller. Capabilities include the device interface details
   available to Measurement Tasks as well as the set of Measurement
   Tasks/Roles (specified by registry entries) that are actually
   installed or available on the MA. Status information includes the
   times that operations were last performed such as contacting the
   Controller or producing Reports.

3.5.1. Definition of ma-capability-obj

```
   object {
        string                    ma-capability-hardware;
        string                    ma-capability-firmware;
        string                    ma-capability-version;
       [string                    ma-capability-tags<0..*>;]
       [ma-capability-task-obj  ma-capability-tasks<0..*>;]
   } ma-capability-obj;
```

   The ma-capability-obj provides information about the capabilities of
   the measurement agent and consists of the following elements:

   ma-capability-hardware: A description of the hardware of the device
                             the measurement agent is running on.

    ma-capability-firmware: A description of the firmware of the device
                              the measurement agent is running on.

    ma-capability-version: The version of the measurement agent.

    ma-capability-tags: An optional unordered set of tags that
                              provide additional information about the
                              capabilities of the measurement agent.

    ma-capability-tasks: An optional unordered set of capability
                              objects for each supported task.

3.5.2. Definition of ma-capability-task-obj

```
object {
    string              ma-capability-task-name;
    ma-registry-obj     ma-capability-task-functions<0..*>;
    string              ma-capability-task-version;
} ma-capability-task-obj;
```

The ma-capability-task-obj provides information about the capability
of a task and consists of the following elements:

    ma-capability-task-name: A name uniquely identifying a task.

    ma-capability-task-functions: A possibly empty unordered set of
                                    registry entries identifying
                                    functions this task implements.

    ma-capability-task-version: The version of the measurement task.

3.5.3. Definition of ma-status-obj

```
object {
     uuid                     ma-status-agent-id;
    [uri                      ma-status-device-id;]
     datetime                 ma-status-last-started;
     ma-status-interface-obj  ma-status-interfaces<0..*>;
    [ma-status-schedule-obj    ma-status-schedules<0..*>;]
    [ma-status-suppression-obj  ma-status-suppressions<0..*>;]
} ma-status-obj;
```

The ma-status-obj provides status information about the measurement
agent and consists of the following elements:

    ma-status-agent-id: A uuid uniquely identifying the measurement
                             agent.

ma-status-device-id: A URI identifying the device.

ma-status-last-started: The date and time the measurement agent
                          last started.

ma-status-interfaces: An unordered set of network interfaces
                          available on the device.

ma-status-schedules: An optional unordered set of status objects
                          for each schedule.

ma-status-suppressions: An optional unordered set of status objects
                          for each suppression.

3.5.4. Definition of ma-status-schedule-obj

```
object {
    string                ma-status-schedule-name;
    string                ma-status-schedule-state;
    int                   ma-status-schedule-storage;
    counter               ma-status-schedule-invocations;
    counter               ma-status-schedule-suppressions;
    counter               ma-status-schedule-overlaps;
    counter               ma-status-schedule-failures;
    datetime              ma-status-schedule-last-invocation;
   [ma-status-action-obj  ma-status-schedule-actions<0..*>;]
} ma-status-schedule-obj;
```

The ma-status-schedule-obj provides status information about the
status of a schedule and consists of the following elements:

ma-status-schedule-name: The name of the schedule this
                               status object refers to.

ma-status-schedule-state: The state of the schedule. The
                               value 'enabled' indicates that
                               the schedule is currently
                               enabled. The value 'suppressed'
                               indicates that the schedule is
                               currently suppressed. The value
                               'disabled' indicates that the
                               schedule is currently disabled.
                               The value 'running' indicates
                               that the schedule is currently
                               running.

ma-status-schedule-storage: The amount of secondary storage
                               (e.g., allocated in a file

                                      system) holding temporary data
                                      allocated to the schedule in
                                      bytes. This object reports the
                                      amount of allocated physical
                                      storage and not the storage used
                                      by logical data records. Data
                                      models should use a 64-bit
                                      integer type.

   ma-status-schedule-invocations   Number of invocations of this
                                      schedule. This counter does not
                                      include suppressed invocations or
                                      invocations that were prevented
                                      due to an overlap with a previous
                                      invocation of this schedule.

   ma-status-schedule-suppressions   Number of suppressed executions
                                      of this schedule.

   ma-status-schedule-overlaps   Number of executions prevented
                                      due to overlaps with a previous
                                      invocation of this schedule.

   ma-status-schedule-failures   Number of failed executions of
                                      this schedule. A failed
                                      execution is an execution where
                                      at least one action failed.

   ma-status-schedule-last-invocation: The date and time of the last
                                      invocation of this schedule.

   ma-status-schedule-actions: An optional ordered list of
                                      status objects for each action of
                                      the schedule.

3.5.5. **Definition of ma-status-action-obj**

```
   object {
       string              ma-status-action-name;
       string              ma-status-action-state;
       int                 ma-status-action-storage;
       counter             ma-status-action-invocations;
       counter             ma-status-action-suppressions;
       counter             ma-status-action-overlaps;
       counter             ma-status-action-failures;
       datetime            ma-status-action-last-invocation;
       datetime            ma-status-action-last-completion;
       int                 ma-status-action-last-status;
       string              ma-status-action-last-message;
       datetime            ma-status-action-last-failed-completion;
       int                 ma-status-action-last-failed-status;
       string              ma-status-action-last-failed-message;
   } ma-status-action-obj;
```

The ma-status-action-obj provides status information about an action
of a schedule and consists of the following elements:

ma-status-action-name: The name of the action of a
                                         schedule this status object
                                         refers to.

ma-status-action-state: The state of the action.
                                         The value 'enabled'
                                         indicates that the action is
                                         currently enabled. The
                                         value 'suppressed' indicates
                                         that the action is currently
                                         suppressed. The value
                                         'disabled' indicates that
                                         the action is currently
                                         disabled. The value
                                         'running' indicates that the
                                         action is currently running.

ma-status-action-storage: The amount of secondary
                                         storage (e.g., allocated in
                                         a file system) holding
                                         temporary data allocated to
                                         the action in bytes. This
                                         object reports the amount of
                                         allocated physical storage
                                         and not the storage used by
                                         logical data records. Data
                                         models should use a 64-bit
                                         integer type.

   ma-status-action-invocations   Number of invocations of
                                       this action. This counter
                                       does not include suppressed
                                       invocations or invocations
                                       that were prevented due to
                                       an overlap with a previous
                                       invocation of this action.

   ma-status-action-suppressions   Number of suppressed
                                       executions of this action.

   ma-status-action-overlaps   Number of executions
                                       prevented due to overlaps
                                       with a previous invocation
                                       of this action.

   ma-status-action-failures   Number of failed executions
                                       of this action.

   ma-status-action-last-invocation: The date and time of the
                                       last invocation of this
                                       action.

   ma-status-action-last-completion: The date and time of the
                                       last completion of this
                                       action.

   ma-status-action-last-status: The status code returned by
                                       the last execution of this
                                       action.

   ma-status-action-last-message: The status message produced
                                       by the last execution of
                                       this action.

   ma-status-action-last-failed-completion: The date and time of the
                                       last failed completion of
                                       this action.

   ma-status-action-last-failed-status: The status code returned by
                                       the last failed execution of
                                       this action.

   ma-status-action-last-failed-message: The status message produced
                                       by the last failed execution
                                       of this action.

3.5.6. Definition of ma-status-suppression-obj

```
object {
    string                ma-status-suppression-name;
    string                ma-status-suppression-state;
} ma-status-suppression-obj;
```

The ma-status-suppression-obj provides status information about that
status of a suppression and consists of the following elements:

ma-status-suppression-name: The name of the suppression this status
                              object refers to.

ma-status-suppression-state: The state of the suppression. The
                              value 'enabled' indicates that the
                              suppression is currently enabled. The
                              value 'active indicates that the
                              suppression is currently active. The
                              value 'disabled' indicates that the
                              suppression is currently disabled.

3.5.7. Definition of ma-status-interface-obj

```
object {
     string            ma-status-interface-name;
     string            ma-status-interface-type;
    [int               ma-status-interface-speed;]
    [string            ma-status-interface-link-layer-address;]
    [ip-address        ma-status-interface-ip-addresses<0..*>;]
    [ip-address        ma-status-interface-gateways<0..*>;]
    [ip-address        ma-status-interface-dns-servers<0..*>;]
} ma-status-interface-obj;
```

The ma-status-interface-obj provides status information about network
interfaces and consists of the following elements:

ma-status-interface-name: A name uniquely identifying a
                                    network interface.

ma-status-interface-type: The type of the network
                                    interface.

ma-status-interface-speed: An optional indication of the
                                    speed of the interface
                                    (measured in bits-per-
                                    second).

ma-status-interface-link-layer-address: An optional link-layer
                                        address of the interface.

ma-status-interface-ip-addresses: An optional ordered list of
                                        IP addresses assigned to the
                                        interface.

ma-status-interface-gateways: An optional ordered list of
                                        gateways assigned to the
                                        interface.

ma-status-interface-dns-servers: An optional ordered list of
                                        DNS servers assigned to the
                                        interface.

3.6. Reporting Information

   At a point in time specified by a Schedule, the MA will execute tasks
   that communicate a set of measurement results to the Collector.
   These Reporting Tasks will be configured to transmit task results
   over a specified Report Channel to a Collector.

   It should be noted that the output from Tasks does not need to be
   sent to communication Channels. It can alternatively, or
   additionally, be sent to other Tasks on the MA. This facilitates
   using a first Measurement Task to control the operation of a later
   Measurement Task (such as first probing available line speed and then
   adjusting the operation of a video testing measurement) and also to
   allow local processing of data to output alarms (e.g., when
   performance drops from earlier levels). Of course, subsequent Tasks
   also include Tasks that implement the reporting protocol(s) and
   transfer data to one or more Collector(s).

   The Report generated by a Reporting Task is structured hierarchically
   to avoid repetition of report header and Measurement Task
   Configuration information. The report starts with the timestamp of
   the report generation on the MA and details about the MA including
   the optional Measurement Agent ID and Group ID (controlled by the
   Configuration Information).

   Much of the report Information is optional and will depend on the
   implementation of the Reporting Task and any parameters defined in
   the Task Configuration for the Reporting Task. For example some
   Reporting Tasks may choose not to include the Measurement Task
   Configuration or Action parameters, while others may do so dependent
   on the Controller setting a configurable parameter in the Task
   Configuration.

It is possible for a Reporting Task to send just the Report header
(datetime and optional agent ID and/or Group ID) if no measurement
data is available. Whether to send such empty reports again is
dependent on the implementation of the Reporting Task and potential
Task Configuration parameter.

The handling of measurement data on the MA before generating a Report
and transfer from the MA to the Collector is dependent on the
implementation of the device, MA and/or scheduled Tasks and not
defined by the LMAP standards. Such decisions may include limits to
the measurement data storage and what to do when such available
storage becomes depleted. It is generally suggested that
implementations running out of storage stop executing new measurement
tasks and retain old measurement data.

No context information, such as line speed or broadband product are
included within the report header information as this data is
reported by individual tasks at the time they execute. Either a
Measurement Task can report contextual parameters that are relevant
to that particular measurement, or specific tasks can be used to
gather a set of contextual and environmental data at certain times
independent of the reporting schedule.

After the report header information the results are reported grouped
according to different Measurement Task Configurations. Each Task
section optionally starts with replicating the Measurement Task
Configuration information before the result headers (titles for data
columns) and the result data rows. The Options reported are those
used for the scheduled execution of the Measurement Task and
therefore include the Options specified in the Task Configuration as
well as additional Options specified in the Action. The Action
Options are appended to the Task Configuration Options in exactly the
same order as they were provided to the Task during execution.

The result row data includes a time for the start of the measurement
and optionally an end time where the duration also needs to be
considered in the data analysis.

Some Measurement Tasks may optionally include an indication of the
cross-traffic although the definition of cross-traffic is left up to
each individual Measurement Task. Some Measurement Tasks may also
output other environmental measures in addition to cross-traffic such
as CPU utlilisation or interface speed.

Whereas the Configuration and Instruction information represent
information transmitted via the Control Protocol, the Report
represents the information that is transmitted via the Report
Protocol. It is constructed at the time of sending a report and

represents the inherent structure of the information that is sent to
the Collector.

3.6.1. Definition of ma-report-obj

```
object {
     datetime              ma-report-date;
    [uuid                  ma-report-agent-id;]
    [string                ma-report-group-id;]
    [string                ma-report-measurement-point;]
    [ma-report-result-obj  ma-report-results<0..*>;]
} ma-report-obj;
```

The ma-report-obj provides the meta-data of a single report and
consists of the following elements:

ma-report-date: The date and time when the report was
                           sent to a collector.

ma-report-agent-id: An optional uuid uniquely identifying
                           the measurement agent.

ma-report-group-id: An optional identifier of the group of
                           measurement agents this measurement
                           agent belongs to.

ma-report-measurement-point: An optional identifier for the
                           measurement point indicating where the
                           measurement agent is located on a path
                           (see [RFC7398] for further details).

ma-report-results: An optional and possibly empty
                           unordered set of result objects.

3.6.2. Definition of ma-report-result-obj

```
   object {
      string                  ma-report-result-schedule-name;
      string                  ma-report-result-action-name;
      string                  ma-report-result-task-name;
     [ma-option-obj           ma-report-result-options<0..*>;]
     [string                  ma-report-result-tags<0..*>;]
      datetime                ma-report-result-event-time;
      datetime                ma-report-result-start-time;
     [datetime                ma-report-result-end-time;]
     [string                  ma-report-result-cycle-number;]
      int                     ma-report-result-status;
     [ma-report-conflict-obj  ma-report-result-conflicts<0..*>;]
     [ma-report-table-obj     ma-report-result-tables<0..*>;]
   } ma-report-result-obj;
```

The ma-report-result-obj provides the meta-data of a result report of
a single executed action. It consists of the following elements:

ma-report-result-schedule-name: The name of the schedule that
                                produced the result.

ma-report-result-action-name: The name of the action in the
                                schedule that produced the result.

ma-report-result-task-name: The name of the task that produced
                                the result.

ma-report-result-options: An optional ordered joined list of
                                options provided by the task object
                                and the action object when the action
                                was started.

ma-report-result-tags: An optional unordered set of tags.
                                This is the joined set of tags
                                provided by the task object and the
                                action object and schedule object
                                when the action was started.

ma-report-result-event-time: The date and time of the event that
                                triggered the schedule of the action
                                that produced the reported result
                                values. The date and time does not
                                include any added randomization.

ma-report-result-start-time: The date and time of the start of the
                                action that produced the reported
                                result values.

ma-report-result-end-time: An optional date and time indicating
                                 when the action finished.

ma-report-result-cycle-number: An optional cycle number derived from
                                 ma-report-result-event-time. It is
                                 the time closest to ma-report-result-
                                 event-time that is a multiple of the
                                 ma-event-cycle-interval of the event
                                 that triggered the execution of the
                                 schedule. The value is only present
                                 in an ma-report-result-obj if the
                                 event that triggered the execution of
                                 the schedule has a defined ma-event-
                                 cycle-interval. The cycle number is
                                 represented in the format
                                 YYYYMMDD.HHMMSS where YYYY represents
                                 the year, MM the month (1..12), DD
                                 the day of the months (01..31), HH
                                 the hour (00..23), MM the minute
                                 (00..59), and SS the second (00..59).
                                 The cycle number is using Coordinated
                                 Universal Time (UTC).

ma-report-result-status: The status code returned by the
                                 execution of the action.

ma-report-result-conflicts: A possibly empty set of conflict
                                 actions that might have impacted the
                                 measurement results being reported.

ma-report-result-tables: An optional and possibly empty
                                 unordered set of result tables.

3.6.3. Definition of ma-report-conflict-obj

```
object {
    string  ma-report-conflict-schedule-name;
    string  ma-report-conflict-action-name;
    string  ma-report-conflict-task-name;
} ma-report-conflict-obj;
```

The ma-report-conflict-obj provides the information about conflicting
action that might have impacted the measurement results. It consists
of the following elements:

ma-report-result-schedule-name: The name of the schedule that may
                                have impacted the result.

    ma-report-result-action-name: The name of the action in the
                                        schedule that may have impacted the
                                        result.

    ma-report-result-task-name: The name of the task that may have
                                        impacted the result.

3.6.4. Definition of ma-report-table-obj

```
object {
   [ma-registry-obj    ma-report-table-functions<0..*>;]
   [string]            ma-report-table-column-labels<0..*>;]
   [ma-report-row-obj  ma-report-table-rows<0..*>;]
} ma-report-table-obj;
```

The ma-report-table-obj represents a result table and consists of the
following elements:

    ma-report-table-functions: An optional and possibly empty
                                          unordered set of registry entries
                                        identifying the functions for which
                                          results that are reported.

    ma-report-table-column-labels: An optional and possibly empty
                                          ordered list of column labels.

    ma-report-table-rows: A possibly empty ordered list of
                                          result rows.

3.6.5. Definition of ma-report-row-obj

```
object {
    data               ma-report-row-values<0..*>;
} ma-report-row-obj;
```

The ma-report-row-obj represents a result row and consists of the
following elements:

    ma-report-row-values: A possibly empty ordered list of result
                              values. When present, it contains an
                              ordered list of values that align to the
                              set of column labels for the report.

3.7. Common Objects: Schedules

A Schedule specifies the execution of a single or repeated series of
Actions. An Action extends a Configured Task with additional
specific parameters. Each Schedule contains basically two elements:

an ordered list of Actions to be executed and an Event object
triggering the execution of the Schedule. The Schedule states what
Actions to run (with what configuration) and when to run the Actions.
A Schedule may optionally have an Event that stops the execution of
the Schedule or a maximum duration after which a schedule is stopped.

Multiple Actions contained as an ordered list of a single Measurement
Schedule will be executed according to the execution mode of the
Schedule. In sequential mode, Actions will be executed sequentially
and in parallel mode, all Actions will be executed concurrently. In
pipelined mode, data produced by one Action is passed to the
subsequent Action. Actions contained in different Schedules execute
in parallel with such conflicts being reported in the Reporting
Information where necessary. If two or more Schedules have the same
start time, then the two will execute in parallel. There is no
mechanism to prioritise one schedule over another or to mutex
scheduled tasks.

As well as specifying which Actions to execute, the Schedule also
specifies how to link the data outputs from each Action to other
Schedules. Specifying this within the Schedule allows the highest
level of flexibility since it is even possible to send the output
from different executions of the same Task Configuration to different
destinations. A single Task producing multiple different outputs is
expected to properly tag the different result. An Action receiving
the output can then filter the results based on the tag if necessary.
For example, a Measurement Task might report routine results to a
data Reporting Task in a Schedule that communicates hourly via the
Broadband PPP interface, but also outputs emergency conditions via an
alarm Reporting Task in a different Schedule communicating
immediately over a GPRS channel. Note that task-to-task data
transfer is always specified in association with the scheduled
execution of the sending task - there is no need for a corresponding
input specification for the receiving task. While it is likely that
an MA implementation will use a queue mechanism between the Schedules
or Actions, this Information Model does not mandate or define a
queue. The Information Model, however, reports the storage allocated
to Schedules and Actions so that storage usage can be monitored.
Furthermore, it is recommended that MA implementations by default
retain old data and stop the execution of new measurement tasks if
the MA runs out of storage capacity.

When specifying the task to execute within the Schedule, i.e.,
creating an Action, it is possible to add to the Action option
parameters. This allows the Task Configuration to determine the
common characteristics of a Task, while selected parameters (e.g.,
the test target URL) are defined within as option parameters of the
Action in the schedule. A single Tasks Configuration can even be

used multiple times in the same schedule with different additional
parameters. This allows for efficiency in creating and transferring
the Instruction. Note that the semantics of what happens if an
option is defined multiple times (either in the Task Configuration,
Action or in both) is not standardised and will depend upon the Task.
For example, some tasks may legitimately take multiple values for a
single parameter.

Where Options are specified in both the Action and the Task
Configuration, the Action Options are appended to those specified in
the Task Configuration.

Example: An Action of a Schedule references a single Measurement
   Task Configuration for measuring UDP latency. It specifies that
   results are to be sent to a Schedule with a Reporting Action.
   This Reporting Task of the Reporting Action is executed by a
   separate Schedule that specifies that it should run hourly at 5
   minutes past the hour. When run this Reporting Action takes the
   data generated by the UDP latency Measurement Task as well as any
   other data to be included in the hourly report and transfers it to
   the Collector over the Report Channel specified within its own
   Schedule.

Schedules and Actions may optionally also be given tags that are
included in result reports sent to a Collector. In addition,
schedules can be given suppression tags that may be used to select
Schedules and Actions for suppression.

3.7.1. Definition of ma-schedule-obj

```
object {
    string              ma-schedule-name;
    ma-event-obj        ma-schedule-start;
   [ma-event-obj        ma-schedule-end;]
   [int                 ma-schedule-duration;]
    ma-action-obj       ma-schedule-actions<0..*>;
    string              ma-schedule-execution-mode;
   [string              ma-schedule-tags<0..*>;]
   [string              ma-schedule-suppression-tags<0..*>;]
} ma-schedule-obj;
```

The ma-schedule-obj is the main scheduling object. It consists of
the following elements:

ma-schedule-name: A name uniquely identifying a
                             scheduling object.

   ma-schedule-start: An event object indicating when the
                                schedule starts.

   ma-schedule-end: An optional event object controlling
                                the forceful termination of scheduled
                                actions. When the event occurs, all
                                actions of the schedule will be forced
                                to terminate gracefully.

   ma-schedule-duration: An optional duration in seconds for the
                                schedule. All actions of the schedule
                                will be forced to terminate gracefully
                                after the duration number of seconds
                                past the start of the schedule.

   ma-schedule-actions: A possibly empty ordered list of
                                actions to invoke when the schedule
                                starts.

   ma-schedule-execution-mode: Indicates whether the actions should be
                                executed sequentially, in parallel, or
                                in a pipelined mode (where data
                                produced by one action is passed to the
                                subsequent action). The default
                                execution mode is pipelined.

   ma-schedule-tags: An optional unordered set of tags that
                                are reported together with the
                                measurement results to a collector.

   ma-schedule-suppression-tags: An optional unordered set of
                                suppression tags that are used to
                                select schedules to be suppressed.

3.7.2. Definition of ma-action-obj

```
   object {
        string               ma-action-name;
        string               ma-action-config-task-name;
       [ma-option-obj        ma-action-task-options<0..*>;]
       [string               ma-action-destinations<0..*>;]
       [string               ma-action-tags<0..*>;]
       [string               ma-action-suppression-tags<0..*>;]
     } ma-action-obj;
```

   The ma-action-obj models a task together with its schedule specific
   task options and destination schedules. It consists of the following
   elements:

   ma-action-name: A name uniquely identifying an action
                                  of a scheduling object.

   ma-action-config-task-name: A name identifying the configured task
                                  to be invoked by the action.

   ma-action-task-options: An optional and possibly empty ordered
                                  list of options (name-value pairs) that
                                  are passed to the task by appending
                                  them to the options configured for the
                                  task object.

   ma-action-destinations: An optional and possibly empty
                                  unordered set of names of destination
                                  schedules that consume output produced
                                  by this action.

   ma-action-tags: An optional unordered set of tags that
                                  are reported together with the
                                  measurement results to a collector.

   ma-action-suppression-tags: An optional unordered set of
                                  suppression tags that are used to
                                  select actions to be suppressed.

3.8. Common Objects: Channels

   A Channel defines a bi-directional communication channel between the
   MA and a Controller or Collector. Multiple Channels can be defined
   to enable results to be split or duplicated across different
   Collectors.

   Each Channel contains the details of the remote endpoint (including
   location and security credential information such as the
   certificate). The timing of when to communicate over a Channel is
   specified by the Schedule which executes the corresponding Control or
   Reporting Task. The certificate can be the digital certificate
   associated to the FQDN in the URL or it can be the certificate of the
   Certification Authority that was used to issue the certificate for
   the FQDN (Fully Qualified Domain Name) of the target URL (which will
   be retrieved later on using a communication protocol such as TLS).
   In order to establish a secure channel, the MA will use its own
   security credentials (in the Configuration Information) and the given
   credentials for the individual Channel end-point.

   As with the Task Configurations, each Channel is also given a text
   name by which it can be referenced as a Task Option.

   Although the same in terms of information, Channels used for
   communication with the Controller are referred to as Control Channels
   whereas Channels to Collectors are referred to as Report Channels.
   Hence Control Channels will be referenced from Control Tasks executed
   by a Control Schedule, whereas Report Channels will be referenced
   from within Reporting Tasks executed by an Instruction Schedule.

   Multiple interfaces are also supported. For example the Reporting
   Task could be configured to send some results over GPRS. This is
   especially useful when such results indicate the loss of connectivity
   on a different network interface.

   Example: A Channel used for reporting results may specify that
      results are to be sent to the URL (https://collector.example.org/
      report/), using the appropriate digital certificate to establish a
      secure channel.

3.8.1. Definition of ma-channel-obj

```
object {
    string              ma-channel-name;
    url                 ma-channel-target;
    credentials         ma-channel-credentials;
    [string             ma-channel-interface-name;]
} ma-channel-obj;
```

   The ma-channel-obj consists of the following elements:

   ma-channel-name: A unique name identifying the channel
                               object.

   ma-channel-target: A URL identifying the target channel
                               endpoint.

   ma-channel-credentials: The security credentials needed to
                               establish a secure channel.

   ma-channel-interface-name: An optional name of the network interface
                               to be used. If not present, the IP
                               protocol stack will select a suitable
                               interface.

3.9. Common Objects: Task Configurations

   Conceptually each Task Configuration defines the parameters of a Task
   that the Measurement Agent (MA) may perform at some point in time.
   It does not by itself actually instruct the MA to perform them at any
   particular time (this is done by a Schedule). Tasks can be

Measurement Tasks (i.e., those Tasks actually performing some type of
passive or active measurement) or any other scheduled activity
performed by the MA such as transferring information to or from the
Controller and Collectors. Other examples of Tasks may include data
manipulation or processing Tasks conducted on the MA.

A Measurement Task Configuration is the same in information terms to
any other Task Configuration. Both measurement and non-measurement
Tasks may have registry entries to enable the MA to uniquely identify
the Task it should execute and retrieve the schema for any parameters
that may be passed to the Task. Registry entries are specified as a
URI and can therefore be used to identify the Task within a namespace
or point to a web or local file location for the Task information.
As mentioned previously, these URIs may be used to identify the
Measurement Task in a public namespace
[I-D.ietf-ippm-metric-registry].

Example: A Measurement Task Configuration may configure a single
   Measurement Task for measuring UDP latency. The Measurement Task
   Configuration could define the destination port and address for
   the measurement as well as the duration, internal packet timing
   strategy and other parameters (for example a stream for one hour
   and sending one packet every 500 ms). It may also define the
   output type and possible parameters (for example the output type
   can be the 95th percentile mean) where the measurement task
   accepts such parameters. It does not define when the task starts
   (this is defined by the Schedule element), so it does not by
   itself instruct the MA to actually perform this Measurement Task.

The Task Configuration will include a local short name for reference
by a Schedule. Task Configurations may also refer to registry
entries as described above. In addition the Task can be configured
through a set of configuration Options. The nature and number of
these Options will depend upon the Task. These options are expressed
as name-value pairs although the 'value' may be a structured object
instead of a simple string or numeric value. The implementation of
these name-value pairs will vary between data models.

An Option that must be present for Reporting Tasks is the Channel
reference specifying how to communicate with a Collector. This is
included in the task options and will have a value that matches a
channel name that has been defined in the Instruction. Similarly
Control Tasks will have a similar option with the value set to a
specified Control Channel.

A Reporting Task might also have a flag parameter, defined as an
Option, to indicate whether to send a report without measurement
results if there is no measurement result data pending to be

transferred to the Collector. In addition many tasks will also take
as a parameter which interface to operate over.

In addition the Task Configuration may optionally also be given tags
that can carry a Measurement Cycle ID. The purpose of this ID is to
easily identify a set of measurement results that have been produced
by Measurement Tasks with comparable Options. This ID could be
manually incremented or otherwise changed when an Option change is
implemented which could mean that two sets of results should not be
directly compared.

3.9.1. Definition of ma-task-obj

```
object {
    string              ma-task-name;
    ma-registry-obj     ma-task-functions<0..*>;
  [ma-option-obj        ma-task-options<0..*>;]
  [string               ma-task-tags<0..*>;]
} ma-task-obj;
```

The ma-task-obj defines a configured task that can be invoked as part
of an action. A configured task can be referenced by its name and it
contains a possibly empty set of URIs to link to registry entries.
Options allow the configuration of task parameters (in the form of
name-value pairs). The ma-task-obj consists of the following
elements:

ma-task-name: A name uniquely identifying a configured
                          task object.

ma-task-functions: A possibly empty unordered set of registry
                          entries identifying the functions of the
                          configured task.

ma-task-options: An optional and possibly empty ordered list
                          of options (name-value pairs) that are
                          passed to the configured task.

ma-task-tags: An optional unordered set of tags that are
                          reported together with the measurement
                          results to a collector.

3.9.2. Definition of ma-option-obj

```
object {
    string              ma-option-name;
  [object               ma-option-value;]
} ma-option-obj;
```

The ma-option-obj models a name-value pair and consists of the
following elements:

ma-option-name: The name of the option.

ma-option-value: The optional value of the option.

The ma-option-obj is used to define Task Configuration Options. Task
Configuration Options are generally task specific. For tasks
associated with an entry in a registry, the registry may define well-
known option names (e.g., the so-called parameters in the IPPM metric
registry [I-D.ietf-ippm-metric-registry]). Control and Reporting
Tasks need to know the Channel they are going to use. The common
option name for specifying the channel is "channel" where the
option's value refers to the name of an ma-channel-obj.

3.10. Common Objects: Registry Information

Tasks and actions can be associated with entries in a registry. A
registry object refers to an entry in a registry (identified by a
URI) and it may define a set of roles.

3.10.1. Definition of ma-registry-obj

```
  object {
     uri                   ma-registry-uri;
    [string                ma-registry-role<0..*>;]
  } ma-registry-obj;
```

The ma-registry-obj refers to an entry of a registry and it defines
the associated role(s). The ma-registry-obj consists of the
following elements:

ma-registry-uri: A URI identifying an entry in a registry.

ma-registry-role: An optional and possibly empty unordered
                           set of roles for the identified registry
                           entry.

3.11. Common Objects: Event Information

The Event information object used throughout the information models
can initially take one of several different forms. Additional forms
may be defined later in order to bind the execution of schedules to
additional events. The initially defined Event forms are:

1. Periodic Timing: Emits multiple events periodically according to
    an interval time defined in seconds

2. Calendar Timing: Emits multiple events according to a calendar
    based pattern, e.g., 22 minutes past each hour of the day on
    weekdays

3. One Off Timing: Emits one event at a specific date and time

4. Immediate: Emits one event as soon as possible

5. Startup: Emits an event whenever the MA is started (e.g., at
    device startup)

6. Controller Lost: Emits an event when connectivity to the
    controller has been lost

7. Controller Connected: Emits an event when connectivity to the
    controller has been (re-)established

Optionally each of the Event options may also specify a randomness
that should be evaluated and applied separately to each indicated
event. This randomness parameter defines a uniform interval in
seconds over which the start of the task is delayed from the starting
times specified by the event object.

Both the Periodic and Calendar timing objects allow for a series of
Actions to be executed. While both have an optional end time, it is
best practice to always configure an end time and refresh the
information periodically to ensure that lost MAs do not continue
their tasks forever.

Startup events are only created on device startup, not when a new
Instruction is transferred to the MA. If scheduled task execution is
desired both on the transfer of the Instruction and on device restart
then both the Immediate and Startup timing needs to be used in
conjunction.

The datetime format used for all elements in the information model
MUST conform to RFC 3339 [RFC3339].

3.11.1. Definition of ma-event-obj

```
   object {
      string               ma-event-name;
      union {
          ma-periodic-obj          ma-event-periodic;
          ma-calendar-obj          ma-event-calendar;
          ma-one-off-obj           ma-event-one-off;
          ma-immediate-obj         ma-event-immediate;
          ma-startup-obj           ma-event-startup;
          ma-controller-lost-obj     ma-event-controller-lost;
          ma-controller-connected-obj ma-event-controller-connected;
      }
      [int               ma-event-random-spread;]
      [int               ma-event-cycle-interval;]
   } ma-event-obj;
```

The ma-event-obj is the main event object. Event objects are
identified by a name. A generic event object itself contains a more
specific event object. The set of specific event objects should be
extensible. The initial set of specific event objects is further
described below. The ma-event-obj also includes an optional uniform
random spread that can be used to randomize the start times of
schedules triggered by an event. The ma-event-obj consists of the
following elements:

ma-event-name: The name uniquely identifies an event
                               object. Schedules refer to event
                               objects by this name.

ma-event-periodic: The ma-event-periodic is present for
                               periodic timing objects.

ma-event-calendar: The ma-event-calendar is present for
                               calendar timing objects.

ma-event-one-off: The ma-event-one-off is present for
                               one-off timing objects.

ma-event-immediate: The ma-event-immediate is present for
                               immediate event objects.

ma-event-startup: The ma-event-startup is present for
                               startup event objects.

ma-event-controller-lost: The ma-event-controller-lost is
                               present for connectivity to
                               controller lost event objects.

   ma-event-controller-connected: The ma-event-controller-connected is
                                   present for connectivity to a
                                   controller established event objects.

   ma-event-random-spread: The optional ma-event-random-spread
                                   adds a random delay defined in
                                   seconds to the event object. No
                                   random delay is added if ma-event-
                                   random-spread does not exist.

   ma-event-cycle-interval: The optional ma-event-cycle-interval
                                   defines the duration of the time
                                   interval in seconds that is used to
                                   calculate cycle numbers. No cycle
                                   number is calculated if ma-event-
                                   cycle-interval does not exist.

3.11.2. Definition of ma-periodic-obj

```
   object {
      [datetime          ma-periodic-start;]
      [datetime          ma-periodic-end;]
       int               ma-periodic-interval;
   } ma-periodic-obj;
```

   The ma-periodic-obj timing object has an optional start and an
   optional end time plus a periodic interval. Schedules using an ma-
   periodic-obj are started periodically between the start and end time.
   The ma-periodic-obj consists of the following elements:

   ma-periodic-start: The optional date and time at which
                           Schedules using this object are first
                           started. If not present it defaults to
                           immediate.

   ma-periodic-end: The optional date and time at which
                           Schedules using this object are last
                           started. If not present it defaults to
                           indefinite.

   ma-periodic-interval: The interval defines the time in seconds
                           between two consecutive starts of tasks.

3.11.3. Definition of ma-calendar-obj

   Calendar Timing supports the routine execution of Schedules at
   specific times and/or on specific dates. It can support more
   flexible timing than Periodic Timing since the execution of Schedules

does not have to be uniformly spaced. For example a Calendar Timing
could support the execution of a Measurement Task every hour between
6pm and midnight on weekdays only.

Calendar Timing is also required to perform measurements at
meaningful times in relation to network usage (e.g., at peak times).
If the optional timezone offset is not supplied then local system
time is assumed. This is essential in some use cases to ensure
consistent peak-time measurements as well as supporting MA devices
that may be in an unknown timezone or roam between different
timezones (but know their own timezone information such as through
the mobile network).

The calendar elements within the Calendar Timing do not have defaults
in order to avoid accidental high-frequency execution of Tasks. If
all possible values for an element are desired then the wildcard * is
used.

```
object {
    [datetime            ma-calendar-start;]
    [datetime            ma-calendar-end;]
    [string              ma-calendar-months<0..*>;]
    [string              ma-calendar-days-of-week<0..*>;]
    [string              ma-calendar-days-of-month<0..*>;]
    [string              ma-calendar-hours<0..*>;]
    [string              ma-calendar-minutes<0..*>;]
    [string              ma-calendar-seconds<0..*>;]
    [int                 ma-calendar-timezone-offset;]
} ma-calendar-obj;
```

ma-calendar-start: The optional date and time at which
                             Schedules using this object are first
                             started. If not present it defaults to
                             immediate.

ma-calendar-end: The optional date and time at which
                             Schedules using this object are last
                             started. If not present it defaults to
                             indefinite.

ma-calendar-months: The optional set of months (1-12) on
                             which tasks scheduled using this object
                             are started. The wildcard * means all
                             months. If not present, it defaults to
                             no months.

ma-calendar-days-of-week: The optional set of days of a week
                             ("Mon", "Tue", "Wed", "Thu", "Fri",

                                "Sat", "Sun") on which tasks scheduled
                                using this object are started. The
                                wildcard * means all days of the week.
                                If not present, it defaults to no days.

      ma-calendar-days-of-month: The optional set of days of a months
                                (1-31) on which tasks scheduled using
                                this object are started. The wildcard
                                * means all days of a months. If not
                                present, it defaults to no days.

      ma-calendar-hours: The optional set of hours (0-23) on
                                which tasks scheduled using this object
                                are started. The wildcard * means all
                                hours of a day. If not present, it
                                defaults to no hours.

      ma-calendar-minutes: The optional set of minutes (0-59) on
                                which tasks scheduled using this object
                                are started. The wildcard * means all
                                minutes of an hour. If not present, it
                                defaults to no hours.

      ma-calendar-seconds: The optional set of seconds (0-59) on
                                which tasks scheduled using this object
                                are started. The wildcard * means all
                                seconds of an hour. If not present, it
                                defaults to no seconds.

      ma-calendar-timezone-offset: The optional timezone offest in hours.
                                If not present, it defaults to the
                                system's local timezone.

   If a day of the month is specified that does not exist in the month
   (e.g., 29th of Feburary) then those values are ignored.

3.11.4. Definition of ma-one-off-obj

      object {
          datetime   ma-one-off-time;
      } ma-one-off-obj;

   The ma-one-off-obj timing object specifies a fixed point in time.
   Schedules using an ma-one-off-obj are started once at the specified
   date and time. The ma-one-off-obj consists of the following
   elements:

    ma-one-off-time: The date and time at which Schedules using
                                  this object are started.

3.11.5. Definition of ma-immediate-obj

       object {
                                 // empty
       } ma-immediate-obj;

   The ma-immediate-obj event object has no further information
   elements. Schedules using an ma-immediate-obj are started as soon as
   possible.

3.11.6. Definition of ma-startup-obj

       object {
                                 // empty
       } ma-startup-obj;

   The ma-startup-obj event object has no further information elements.
   Schedules or suppressions using an ma-startup-obj are started at MA
   initialization time.

3.11.7. Definition of ma-controller-lost-obj

       object {
                                 // empty
       } ma-controller-lost-obj;

   The ma-controller-lost-obj event object has no further information
   elements. The ma-controller-lost-obj indicates that connectivity to
   the controller has been lost. This is determined by a timer started
   after each successful contact with a controller. When the timer
   reaches the controller-timeout (measured in seconds), an ma-
   controller-lost-obj event is generated. This event may be used to
   start a suppression.

3.11.8. Definition of ma-controller-connected-obj

       object {
                                 // empty
       } ma-controller-connected-obj;

   The ma-controller-connected-obj event object has no further
   information elements. The ma-controller-connected-obj indicates that
   connectivity to the controller has been established again after it
   was lost. This event may be used to end a suppression.

4. **Example Execution**

   The example execution has two event sources E1 and E2 and three
   schedules S1, S2, and S3. The schedule S3 is started by events of
   event source E2 while the schedules S1 and S2 are both started by
   events of the event source E1. The schedules S1 and S2 have two
   actions each and schedule S3 has a single action. The event source
   E2 has no randomization while the event source E1 has the
   randomization r.

   Figure 2 shows a possible timeline of an execution. The time T is
   progressing downwards. The dotted vertial line indicates progress of
   time while a dotted horizontal line indicates which schedule are
   triggered by an event. Tilded lines indicate data flowing from an
   action to another schedule. Actions within a schedule are named A1,
   A2, etc.

```
        E2    E1   T              S1               S2               S3
                             sequential       parallel       pipelined
                     :
             e0 +
                     :
                     :
            e0+r + .......... + .......... ++
                     :              | A1       A1 || A2
                     :              +              |+ ~~~~~~~>
                     :              | A2          |
                     :              |              + ~~~~~~~>
                     :              + ~~~~~~~~~~~~~~~~~~~~>
                     :
                     :
             e1 +
                     :
            e1+r + .......... + .......... ++
                     :              | A1       A1 ||
                     :              |              +|~~~~~~~>
                     :              |              | A2
                     :              +              +~~~~~~~>
                     :              | A2
                     :              + ~~~~~~~~~~~~~~~~~~~~>
         e0         + ................................... +
                     :                                    | A1
             e3 +                                         |
            e3+r + .......... + .......... ++             |
                     :              | A1       A1 || A2    |
                     :              +              ++ ~~~~~>  |
                     :              | A2                  +
                     :              + ~~~~~~~~~~~~~~~~~~~~>
             V
```
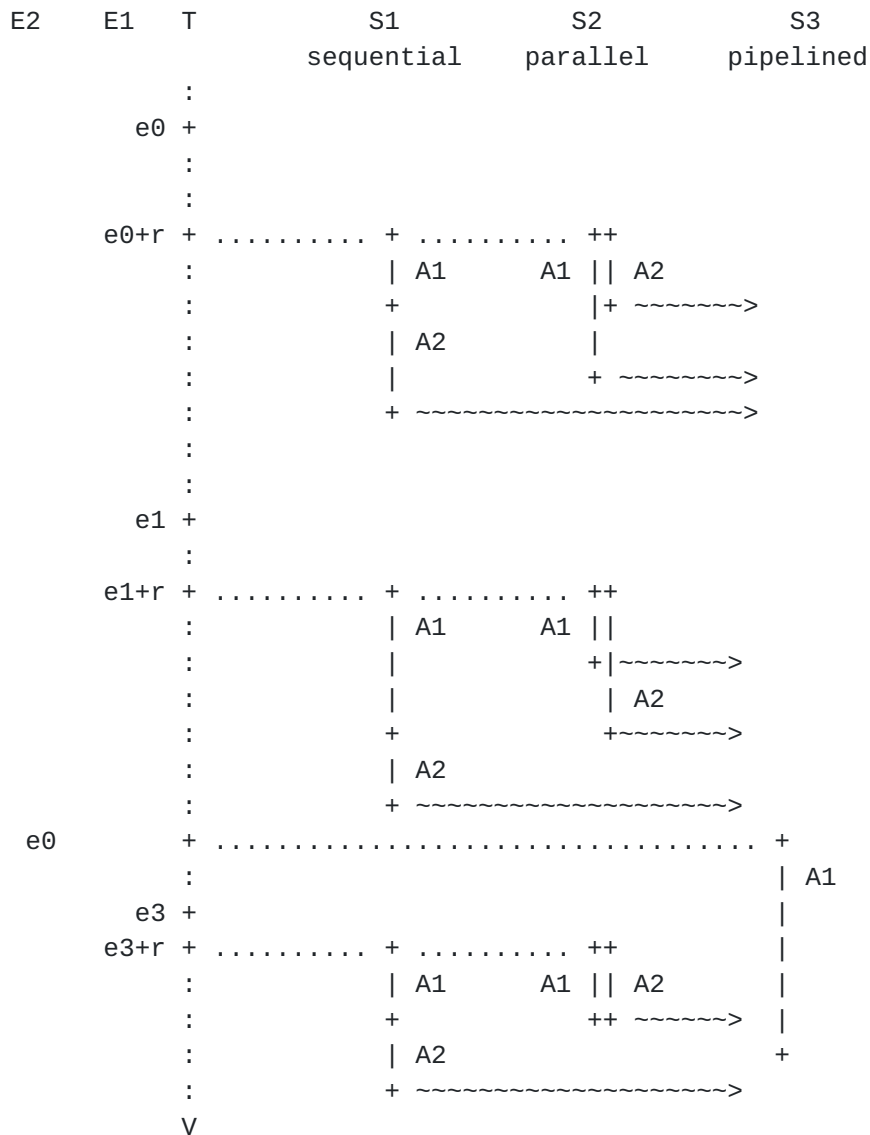
                    Figure 2: Example Execution

   Note that implementations must handle possible concurrency issues.
   In the example execution, action A1 of schedule S3 is consuming the
   data that has been forwarded to schedule S3 while additional data is
   arriving from action A2 of schedule S2.

5. IANA Considerations

   This document makes no request of IANA.

   Note to the RFC Editor: this section may be removed on publication as
   an RFC.

6. Security Considerations

   This Information Model deals with information about the control and
   reporting of the Measurement Agent. There are broadly two security
   considerations for such an Information Model. Firstly the
   Information Model has to be sufficient to establish secure
   communication channels to the Controller and Collector such that
   other information can be sent and received securely. Additionally,
   any mechanisms that the Network Operator or other device
   administrator employs to pre-configure the MA must also be secure to
   protect unauthorized parties from modifying pre-configuration
   information. These mechanisms are important to ensure that the MA
   cannot be hijacked, for example to participate in a distributed
   denial of service attack.

   The second consideration is that no mandated information items should
   pose a risk to confidentiality or privacy given such secure
   communication channels. For this latter reason items such as the MA
   context and MA ID are left optional and can be excluded from some
   deployments. This may, for example, allow the MA to remain anonymous
   and for information about location or other context that might be
   used to identify or track the MA to be omitted or blurred.
   Implementations and deployments should also be careful about exposing
   device-ids when this is not strictly needed.

   An implementation of this Information Model should support wherever
   relevant, all the security and privacy requirements associated with
   the LMAP Framework. In addition, users of this Information Model are
   advised to choose identifiers for Group IDs, tags or names of
   information model objects (e.g., configured tasks, schedules or
   actions) that do not reveal any sensitive information to people
   authorized to process measurement results but who are not authorized
   to know details about the Measurement Agents that were used to
   perform the measurement.

7. Acknowledgements

   Several people contributed to this specification by reviewing early
   versions and actively participating in the LMAP working group
   (apologies to those unintentionally omitted): Vaibhav Bajpai, Michael
   Bugenhagen, Timothy Carey, Alissa Cooper, Kenneth Ko, Al Morton, Dan
   Romascanu, Henning Schulzrinne, Andrea Soppera, Barbara Stark, and
   Jason Weil.

   Trevor Burbridge, Philip Eardley, Marcelo Bagnulo and Juergen
   Schoenwaelder worked in part on the Leone research project, which
   received funding from the European Union Seventh Framework Programme
   [FP7/2007-2013] under grant agreement number 317647.

8. References

8.1. Normative References

   [ISO.10646]
              International Organization for Standardization,
              "Information Technology - Universal Multiple-Octet Coded
              Character Set (UCS)", ISO Standard 10646:2014, 2014.

   [POSIX.2] The IEEE and The Open Group, "The Open Group Base
              Specifications Issue 7", IEEE Standard 1003.1-2008, 2016.

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/
              RFC2119, March 1997,
              <http://www.rfc-editor.org/info/rfc2119>.

   [RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet:
              Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002,
              <http://www.rfc-editor.org/info/rfc3339>.

   [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform
              Resource Identifier (URI): Generic Syntax", STD 66, RFC
              3986, DOI 10.17487/RFC3986, January 2005,
              <http://www.rfc-editor.org/info/rfc3986>.

   [RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally
              Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10
              .17487/RFC4122, July 2005,
              <http://www.rfc-editor.org/info/rfc4122>.

8.2. Informative References

   [I-D.ietf-ippm-metric-registry]
              Bagnulo, M., Claise, B., Eardley, P., Morton, A., and A.
              Akhter, "Registry for Performance Metrics", draft-ietf-
              ippm-metric-registry-10 (work in progress), November 2016.

   [I-D.ietf-lmap-yang]
              Schoenwaelder, J. and V. Bajpai, "A YANG Data Model for
              LMAP Measurement Agents", draft-ietf-lmap-yang-10 (work in
              progress), January 2017.

   [RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between
              Information Models and Data Models", RFC 3444, DOI 10
              .17487/RFC3444, January 2003,
              <http://www.rfc-editor.org/info/rfc3444>.

   [RFC7398] Bagnulo, M., Burbridge, T., Crawford, S., Eardley, P., and
              A. Morton, "A Reference Path and Measurement Points for
              Large-Scale Measurement of Broadband Performance", RFC
              7398, DOI 10.17487/RFC7398, February 2015,
              <http://www.rfc-editor.org/info/rfc7398>.

   [RFC7594] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T.,
              Aitken, P., and A. Akhter, "A Framework for Large-Scale
              Measurement of Broadband Performance (LMAP)", RFC 7594,
              DOI 10.17487/RFC7594, September 2015,
              <http://www.rfc-editor.org/info/rfc7594>.

Appendix A. Change History

   Note to the RFC Editor: this section should be removed on publication
   as an RFC.

A.1. Non-editorial changes since -16

   o Addressing Alissa Cooper's review comments.

A.2. Non-editorial changes since -15

   o The reference to the framework is now informational.

A.3. Non-editorial changes since -14

   o Clarified that the cycle number is in UTC.

A.4. Non-editorial changes since -13

   o Removed the ma-config-device-id from the ma-config-obj.

   o Added ma-config-report-group-id and clarified how two flags ma-
      config-report-agent-id and ma-config-report-group-id work.

A.5. Non-editorial changes since -12

   o Renamed the ma-metrics-registry-obj to ma-registry-obj since tasks
      may refer to different registries (not just a metrics registry).

   o Clarifications and bug fixes.

A.6. **Non-editorial changes since -11**

   o Clarifications and bug fixes.

A.7. **Non-editorial changes since -10**

   o Rewrote the text concerning the well-known "channel" option name.

   o Added ma-report-result-event-time, ma-report-result-cycle-number,
      and ma-event-cycle-interval.

   o Added ma-capability-tags.

   o Added a new section showing an example execution.

   o Several clarifications and bug fixes.

A.8. **Non-editorial changes since -09**

   o Added ma-status-schedule-storage and ma-status-action-storage.

   o Removed suppress-by-default.

   o Moved ma-report-result-metrics of the ma-report-result-obj to ma-
      report-table-metrics of the ma-report-table-obj so that the
      relationship between metrics and result tables is clear.

   o Added ma-report-conflict-obj.

   o Added ma-report-result-status to ma-report-result-obj.

   o Several clarifications and bug fixes.

A.9. **Non-editorial changes since -08**

   o Refactored the ma-report-task-obj into the ma-report-result-obj.

   o Introduced the ma-report-table-obj so that a result can contain
      multiple tables.

   o Report schedule, action, and task name as part of the ma-report-
      result-obj.

   o Report conflicts per ma-report-result-obj and not per ma-report-
      row-obj.

   o Report the start/end time as part of the ma-report-result-obj.

A.10. **Non-editorial changes since -07**

   o Added ma-schedule-end and ma-schedule-duration.

   o Changed the granularity of scheduler timings to seconds.

   o Added ma-status-suppression-obj to report the status of
      suppressions as done in the YANG data model.

   o Added counters to schedule and action status objects to match the
      counters in the YANG data model.

   o Using tags to pass information such as a measurement cycle
      identifier to the collector.

   o Using suppression tags and glob-style matching to select schedules
      and actions to be suppressed.

A.11. **Non-editorial changes since -06**

   o The default execution mode is pipelined (LI12)

   o Added text to define which action consumes data in sequential,
      pipelines, and parallel execution mode (LI11)

   o Added ma-config-measurement-point, ma-report-measurement-point,
      and ma-config-report-measurement-point to configure and report the
      measurement point (LI10)

   o Turned ma-suppression-obj into a list that uses a start event and
      a stop event to define the start and end of suppression; this
      unifies the handling of suppression and loss of controller
      connectivity (LI09)

   o Added ma-controller-lost-obj and ma-controller-ok-obj event
      objects (LI09)

   o Added ma-status-schedule-obj to report the status of a schedule
      and refactored ma-task-status-obj into ma-status-action-obj to
      report the status of an action (LI07, LI08)

   o Introduced a common ma-metric-registry-obj that identifies a
      metric and a set of associated roles and added this object to
      expose metric capabilities and to support the configuration of
      metrics and to report the metrics used (LI06)

   o Introduced ma-capability-obj and ma-capability-task-obj to expose
      the capabilities of a measurement agent (LI05)

   o Use 'ordered list' or 'unordered set' instead of list, collection,
      etc. (LI02)

   o Clarification that Actions are part of a Schedule (LI03)

   o Deleted terms that are not strictly needed (LI04)

A.12. Non-editorial changes since -05

   o A task can now reference multiply registry entries.

   o Consistent usage of the term Action and Task.

   o Schedules are triggered by Events instead of Timings; Timings are
      just one of many possible event sources.

   o Actions feed into other Schedules (instead of Actions within other
      Schedules).

   o Removed the notion of multiple task outputs.

   o Support for sequential, parallel, and pipelined execution of
      Actions.

Authors' Addresses

   Trevor Burbridge
   BT
   Adastral Park, Martlesham Heath
   Ipswich IP5 3RE
   United Kingdom

   Email: trevor.burbridge@bt.com

   Philip Eardley
   BT
   Adastral Park, Martlesham Heath
   Ipswich IP5 3RE
   United Kingdom

   Email: philip.eardley@bt.com

Marcelo Bagnulo
Universidad Carlos III de Madrid
Av. Universidad 30
Leganes, Madrid 28911
Spain

Email: marcelo@it.uc3m.es

Juergen Schoenwaelder
Jacobs University Bremen
Campus Ring 1
Bremen 28759
Germany

Email: j.schoenwaelder@jacobs-university.de