GWD-R
Network Measurement Control Working Group
https://forge.gridforum.org/projects/nmc-wg

Jason Zurawski, Internet2
Martin Swany, UDel
March 16, 2010

NMC Measurement Archive Messages
DRAFT

Status of This Document

This document provides information to the Grid community regarding the design of protocols to control software engaged in the creation, storage, and exchange of network measurements. Distribution is unlimited.

Copyright Notice

Contents

1 Introduction

Extensions to the the *NMC Protocol* [7] have the advantage of further specifying interactions on a message by message basis for use in multiple service types. This eliminates the need to exhaustively document key exchanges for each service, and allows the protocols to evolve over time more efficiently.

The *Measurement Archive* protocol features three messages used to query services identifying themselves as long or short term storage locations of measurement data:

- **MetadataKey** - Message set that supplies metadata information in return for a re-playable key, see Section 2
- **SetupData** - Message set that supplies metadata information or a key in return for a measurement data, see Section 3
- **DataInfo** - Message set that supplies metadata information in return for a well defined key, see Section 4
- **MeasurementArchiveStore** - Message set that attempts to store information into the database, see Section 5

2 Metadata Key

The *MetadataKey* message exchange offers a way to exchange the often complex metadata description of a measurement set for more easily consumed and exchange key. The key structure was first introduced in the *NM Protocol* [4], and can be customized for any particular service implementation; the overall goal being a shortcut to an underlying data set, similar to keys used in relational databases.

Based on initial work in the *NMC Protocol*, this exchange will be described in terms of both the *MetadataKeyRequest* message and *MetadataKeyResponse* message. Each explanation will include a schematic definition and analysis, followed by example instances. Note that no one service will implement this protocol exactly as written; services are expected to offer an explanation of how they implement the protocol as well as any divergence in the service documentation.

2.1 MetadataKeyRequest Message

The *MetadataKeyRequest* message is a container for submitting metadata of a given type to a capable **Measurement Archive**. Enclosed in this envelope will be a series of metadata and data pairs containing various instructions to act on. We first present a very simple schema in Section 2.1.1 along with an analysis of the elements in Section 2.1.2. We conclude with examples in Section 2.1.3.

2.1.1 MetadataKeyRequest Message Schema

The following schema is a native description of the request schema as in the RELAX-NG[3] language. Through the use of tools such as Trang[5] and MSV[2] it is possible to convert this to other widely accepted formats such as XSD[6].

```
# Begin Schema

namespace nmwg = "http://ggf.org/ns/nmwg/base/2.0/"
```

```
start =
  element nmwg:message {
    Identifier? &
    attribute type { "MetadataKeyRequest" } &
    Parameters? &
    (
      Metadata |
      Data
    )+
  }

Parameters =
  element nmwg:parameters {
    Identifier &
    Parameter+
  }

Parameter =
  element nmwg:parameter {
    attribute name { xsd:string } &
    (
      attribute value { xsd:string } |
      (
        anyElement |
        text
      )
    )
  }

Metadata =
  element nmwg:metadata {
    Identifier &
    MetadataIdentifierRef? &
    anyElement*
  }

Data =
  element nmwg:data {
    Identifier &
    MetadataIdentifierRef &
  }

Identifier =
  attribute id { xsd:string }

MetadataIdentifierRef =
  attribute metadataIdRef { xsd:string }

anyElement =
  element * {
    anyThing
  }

anyAttribute =
  attribute * { text }

anyThing =
  (
    anyElement |
    anyAttribute |
    text
  )*


# End Schema
```

2.1.2 MetadataKeyRequest Message Analysis

There are view deviations from the *NMC Protocol* for this particular message type. The following analysis will describe each element, noting when there are changes.

2.1.2.1 Message

```
<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              id="message1"
              type="MetadataKeyRequest">
```

```
  <nmwg:parameters />
  <nmwg:metadata />
  <nmwg:data />
</nmwg:message>
```

Table 1: Message Element Specifics

Message Element	
localname	xmessage
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, type
nested elements	parameters, metadata, data
required	yes

There is no difference in the structure of this element. The only notable difference is the value of *type* is restricted to be *MetadataKeyRequest*.

2.1.2.2 Parameters

```
<nmwg:parameters xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/" id="parameters1">
  <nmwg:parameter />
</nmwg:parameters>
```

Table 2: Parameters Element Specifics

Parameters Element	
localname	parameters
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id
nested elements	parameter
required	no

There is no difference in the structure of this element.

2.1.2.3 Parameter

```
<nmwg:parameter xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
                name="NAME">VALUE</nmwg:parameter>

<!-- OR -->

<nmwg:parameter xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
                name="NAME" value="VALUE" />
```

There is no difference in the structure of this element.

Table 3: Parameter Element Specifics

Parameter Element	
localname	parameter
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	value, name
nested elements	text, undefined
required	yes

2.1.2.4 Metadata

```
<nmwg:metadata xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
               id="metadata2" metadataIdRef="metadata1" />
```

Table 4: Metadata Element Specifics

Metadata Element	
localname	metadata
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, metadataIdRef
nested elements	undefined
required	yes

There is no difference in the structure of this element.

2.1.2.5 Data

```
<nmwg:data xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
           id="data2" metadataIdRef="metadata2" />
```

Table 5: Data Element Specifics

Data Element	
localname	data
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, metadataIdRef
nested elements	N/A
required	varies

There is no difference in the structure of this element. The only notable difference is that this element **SHOULD NOT** contain any nested elements. This element should only be used as a *data trigger* for services to act on particular metadata elements.

2.1.3 MetadataKeyRequest Message Example

The following examples demonstrate some of the possible uses and layouts of request messages in the MA Protocol. These examples are not an attempt to be exhaustive, but rather some examples of ways to perform common tasks. Note that these messages are **NOT** indicative of a particular service.

The first example demonstrates the most common use case: a single metadata and data pair.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="MetadataKeyRequest" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1" />

</nmwg:message>

<!-- End XML -->
```

The second example is similar, but incorporates a parameters block that may be populated with optional behaviors for a service.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="MetadataKeyRequest" id="message1">

  <nmwg:parameters id="paraemters1">
    <nmwg:parameter name="something">something else</nmwg:parameter>
  </nmwg:parameters>

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1" />

</nmwg:message>

<!-- End XML -->
```

The third example is also similar to the first, but shows it is possible to ask for multiple pairs of metadata and data in a single message. Note that there are two empty *data trigger*s to signify that each message be acted upon.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="MetadataKeyRequest" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1" />

  <nmwg:metadata id="m2">
    <!-- another metadata -->
  </nmwg:metadata>

  <nmwg:data id="d2" metadatIdRef="m2" />

</nmwg:message>

<!-- End XML -->
```

This example features merge chaining. Note there is only one *data trigger*, and it is at the tail of the *chain*. A service would perform the necessary chaining first, then act on the result of this operation.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="MetadataKeyRequest" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:metadata id="m2" metadataIdRef="m1">
    <!-- more metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m2" />

</nmwg:message>

<!-- End XML -->
```

The final example is an invalid case where the metadata does not have an appropriate *data trigger*.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="MetadataKeyRequest" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m2" />

</nmwg:message>

<!-- End XML -->
```

2.2 MetadataKeyResponse Message

The *MetadataKeyResponse* message is a container filled with the results of a *MetadataKeyRequest* essage from a capable Measurement Archive service. Enclosed in this simple envelope will be a series of metadata and data pairs containing the results of actions performed by a service. We first present a very simple schema in Section 2.2.1 along with an analysis of the elements in Section 2.2.2. We conclude with examples in Section 2.2.3.

2.2.1 MetadataKeyResponse Message Schema

The following schema is a native description of the request schema as in the RELAX-NG[3] language. Through the use of tools such as Trang[5] and MSV[2] it is possible to convert this to other widely accepted formats such as XSD[6].

```
# Begin Schema

namespace nmwg = "http://ggf.org/ns/nmwg/base/2.0/"

start =
  element nmwg:message {
    Identifier? &
    attribute messageIdRef { xsd:string }? &
    attribute type { "MetadataKeyResponse" } &
    Parameters? &
    (
```

```
      Metadata |
      Data
    )+
  }

Parameters =
  element nmwg:parameters {
    Identifier &
    Parameter+
  }

Parameter =
  element nmwg:parameter {
    attribute name { xsd:string } &
    (
      attribute value { xsd:string } |
      (
        anyElement |
        text
      )
    )
  }

Metadata =
  element nmwg:metadata {
    Identifier &
    MetadataIdentifierRef? &
    anyElement*
  }

Data =
  element nmwg:data {
    Identifier &
    MetadataIdentifierRef &
    (
      Key |
      Datum+
    )
  }

Identifier =
  attribute id { xsd:string }

MetadataIdentifierRef =
  attribute metadataIdRef { xsd:string }

anyElement =
  element * {
    anyThing
  }

anyAttribute =
  attribute * { text }

anyThing =
  (
    anyElement |
    anyAttribute |
    text
  )*

Key =
  element nmwg:key {
    Identifier? &
    Parameters
  }

Datum =
  element nmwg:datum {
    anyThing*
  }

# End Schema
```

2.2.2 MetadataKeyResponse Message Analysis

The following is a breakdown of the elements featured in the schema.

2.2.2.1 Message

```
<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              id="message1"
              type="MetadataKeyResponse">

  <nmwg:parameters />

  <nmwg:metadata />

  <nmwg:data />

</nmwg:message>
```

Table 6: Message Element Specifics

Message Element	
localname	message
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, messageIdRef, type
nested elements	parameters, metadata, data
required	yes

There is no difference in the structure of this element. The only notable difference is the value of *type* is restricted to be *MetadataKeyResponse*.

2.2.2.2 Parameters

```
<nmwg:parameters xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/" id="parameters1">

  <nmwg:parameter />

</nmwg:parameters>
```

Table 7: Parameters Element Specifics

Parameters Element	
localname	parameters
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id
nested elements	parameter
required	no

There is no difference in the structure of this element.

2.2.2.3 Parameter

```
<nmwg:parameter xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
                name="NAME">VALUE</nmwg:parameter>

<!-- OR -->

<nmwg:parameter xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
                name="NAME" value="VALUE" />
```

Table 8: Parameter Element Specifics

Parameter Element	
localname	parameter
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	name, value
nested elements	text, undefined
required	yes

There is no difference in the structure of this element.

2.2.2.4 Metadata

```
<nmwg:metadata xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
               id="metadata2" metadataIdRef="metadata1" />
```

Table 9: Metadata Element Specifics

Metadata Element	
localname	metadata
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, metadataIdRef
nested elements	undefined
required	yes

There is no difference in the structure of this element.

2.2.2.5 Data

```
<nmwg:data xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
           id="data2" metadataIdRef="metadata2" />
```

Table 10: Data Element Specifics

Data Element	
localname	data
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, metadataIdRef
nested elements	key, datum
required	yes

This element features only one deviation from that of the *NMC Protocol*: the key element or a datum element must exist inside. The key element will be explained in Section 2.2.2.6 and the datum element will be explained in Section 2.2.2.7.

2.2.2.6 Key

```
<nmwg:key xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/" id="key1">
  <nmwg:parameters />
</nmwg:key>
```

Table 11: Key Element Specifics

Key Element	
localname	key
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id
nested elements	parameters
required	varies

The *key* element is used to specify particular nuances of a dataset linked to a metadata description. The key element was first described in the *NM Protocol*. This element can contain the following attributes:

- **id -** Identifying attribute that can be used to track state.

There is only one element allowed within the key:

- **Parameters -** Described in Parameters

2.2.2.7 Datum

```
<nmwg:datum xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/" />
```

Table 12: Datum Element Specifics

Datum Element	
localname	datum
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	value, undefined
nested elements	undefined
required	varies

The *datum* element may appear instead of a *key* when anomalous behaviour in the service prevents the return of a key. Legacy or alternate services may also choose to issue the key *inside* of a datum element (see individual service documentation for details). This document will not specific attributes or elements for the datum element.

2.2.3 MetadataKeyResponse Message Example

The following examples demonstrate some of the possible uses and layouts of response messages in the *MA Protocol*.

The first example is the most common form of *MetadataKeyResponse* message containing a single metadata and data pair. This would be indicative of success due to the presence of the key.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="MetadataKeyResponse" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1">

    <nmwg:key id="key1">
      <nmwg:parameters id="parameters1">
        <nmwg:parameter name="something" value="something" />
      </nmwg:parameters>
    </nmwg:key>

  </nmwg:data>

</nmwg:message>

<!-- End XML -->
```

The second example is similar, although it features two pairs.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="MetadataKeyResponse" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1">

    <nmwg:key id="key1">
      <nmwg:parameters id="parameters1">
        <nmwg:parameter name="something" value="something" />
      </nmwg:parameters>
    </nmwg:key>

  </nmwg:data>

  <nmwg:metadata id="m2">
    <!-- another metadata -->
  </nmwg:metadata>

  <nmwg:data id="d2" metadatIdRef="m2">

    <nmwg:key id="key2">
      <nmwg:parameters id="parameters2">
        <nmwg:parameter name="something" value="something" />
      </nmwg:parameters>
    </nmwg:key>

  </nmwg:data>

</nmwg:message>

<!-- End XML -->
```

The final example demonstrates an error condition. Note that this may contain multiple pairs if sent, and it may be possible to have success for some, and errors for others.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="MetadataKeyResponse" id="message1">

  <nmwg:metadata id="m1">
    <!-- some error -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1">

    <nmwg:datum>
      <!-- some error message -->
    </nmwg:datum>

  </nmwg:data>

</nmwg:message>

<!-- End XML -->
```

3 Setup Data

The *SetupData* message exchange offers a way to retrieve measurement data for a given *metadata* description or *key* (collected via the *MetadataKey* message exchange, see Section 2). This message represents the true crux of measurement archive services and allows the retrieval of actual measurement data.

Based on initial work in the *NMC Protocol*, this exchange will be described in terms of both the *SetupDataRequest* message and *SetupDataResponse* message. Each explanation will include a schematic definition and analysis followed by example instances. Note that no one service will implement this protocol *exactly* as presented; services are expected to offer an explanation of how they implement the protocol as well as any divergence in the service documentation.

3.1 SetupDataRequest Message

The *SetupDataRequest* message is a container for submitting metadata of a given type to a capable Measurement Archive. Enclosed in this envelope will be a series of metadata and data pairs containing various instructions to act on. We first present a very simple schema in Section 3.1.1 along with an analysis of the elements in Section 3.1.2. We conclude with examples in Section 3.1.3.

3.1.1 SetupDataRequest Message Schema

The following schema is a native description of the request schema as in the RELAX-NG[3] language. Through the use of tools such as Trang[5] and MSV[2] it is possible to convert this to other widely accepted formats such as XSD[6].

```
# Begin Schema

namespace nmwg = "http://ggf.org/ns/nmwg/base/2.0/"

start =
  element nmwg:message {
    Identifier? &
    attribute type { "SetupDataRequest" } &
    Parameters? &
    (
      Metadata |
      Data
    )+
  }

Parameters =
```

```
  element nmwg:parameters {
    Identifier &
    Parameter+
  }

Parameter =
  element nmwg:parameter {
    attribute name { xsd:string } &
    (
      attribute value { xsd:string } |
      (
        anyElement |
        text
      )
    )
  }

Metadata =
  element nmwg:metadata {
    Identifier &
    MetadataIdentifierRef? &
    anyElement*
  }

Data =
  element nmwg:data {
    Identifier &
    MetadataIdentifierRef &
  }

Identifier =
  attribute id { xsd:string }

MetadataIdentifierRef =
  attribute metadataIdRef { xsd:string }

anyElement =
  element * {
    anyThing
  }

anyAttribute =
  attribute * { text }

anyThing =
  (
    anyElement |
    anyAttribute |
    text
  )*


# End Schema
```

3.1.2 SetupDataRequest Message Analysis

There are view deviations from the *NMC Protocol* for this particular message type. The following analysis will describe each element, noting when there are changes.

3.1.2.1 Message

```
<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              id="message1"
              type="SetupDataRequest">

  <nmwg:parameters />

  <nmwg:metadata />

  <nmwg:data />

</nmwg:message>
```

There is no difference in the structure of this element. The only notable difference is the value of *type* is restricted to be *SetupDataRequest*.

Table 13: Message Element Specifics

Message Element	
localname	message
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, type
nested elements	parameters, metadata, data
required	yes

3.1.2.2 Parameters

```
<nmwg:parameters xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/" id="parameters1">

  <nmwg:parameter />

</nmwg:parameters>
```

Table 14: Parameters Element Specifics

Parameters Element	
localname	parameters
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id
nested elements	parameters
required	no

There is no difference in the structure of this element.

3.1.2.3 Parameter

```
<nmwg:parameter xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
                name="NAME">VALUE</nmwg:parameter>

<!-- OR -->

<nmwg:parameter xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
                name="NAME" value="VALUE" />
```

Table 15: Parameter Element Specifics

Parameter Element	
localname	parameter
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	name, value
nested elements	text
required	yes

There is no difference in the structure of this element.

3.1.2.4 Metadata

```
<nmwg:metadata xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
               id="metadata2" metadataIdRef="metadata1" />
```

Table 16: Metadata Element Specifics

Metadata Element	
localname	metadata
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, metadataIdRef
nested elements	undefined
required	yes

There is no difference in the structure of this element. Note that the contents will vary from service to service.

3.1.2.5 Data

```
<nmwg:data xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
           id="data2" metadataIdRef="metadata2" />
```

Table 17: Data Element Specifics

Data Element	
localname	data
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, metadataIdRef
nested elements	N/A
required	varies

There is no difference in the structure of this element. The only notable difference is that this element **SHOULD NOT** contain any nested elements. This element should only be used as a *data trigger* for services to act on particular metadata elements.

3.1.3 SetupDataRequest Message Example

The following examples demonstrate some of the possible uses and layouts of request messages in the *MA Protocol*. These examples are not an attempt to be exhaustive, but rather some examples of ways to perform common tasks. Note that these messages are **NOT** indicative of a particular service.

The first example demonstrates the most common use case: a single metadata and data pair.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="SetupDataRequest" id="message1">

  <nmwg:metadata id="m1">
```

```
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1" />

</nmwg:message>

<!-- End XML -->
```

The second example is similar, but incorporates a parameters block that may be populated with optional behaviors for a service.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="SetupDataRequest" id="message1">

  <nmwg:parameters id="paraemters1">
    <nmwg:parameter name="something">something else</nmwg:parameter>
  </nmwg:parameters>

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1" />

</nmwg:message>

<!-- End XML -->
```

The third example is also similar to the first, but shows it is possible to ask for multiple pairs of metadata and data in a single message. Note that there are two empty *data trigger*s to signify that each message be acted upon.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="SetupDataRequest" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1" />

  <nmwg:metadata id="m2">
    <!-- another metadata -->
  </nmwg:metadata>

  <nmwg:data id="d2" metadatIdRef="m2" />

</nmwg:message>

<!-- End XML -->
```

This example features *merge chaining*. Note there is only one *data trigger*, and it is at the tail of the *chain*. A service would perform the necessary chaining first, then act on the result of this operation.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="SetupDataRequest" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:metadata id="m2" metadataIdRef="m1">
```

```
    <!-- more metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m2" />

</nmwg:message>

<!-- End XML -->
```

The final example is an invalid case where the metadata does not have an appropriate data trigger.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="SetupDataRequest" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m2" />

</nmwg:message>

<!-- End XML -->
```

3.2 SetupDataResponse Message

The *SetupDataResponse* message is a container filled with the results of a *SetupDataRequest* Message from capable Measurement Archive services. Enclosed in this simple envelope will be a series of metadata and data pairs containing the results of actions performed by a service. We first present a very simple schema in Section 3.2.1 along with an analysis of the elements in Section 3.2.2. We conclude with examples in Section 3.2.3.

3.2.1 SetupDataResponse Message Schema

The following schema is a native description of the request schema as in the RELAX-NG[3] language. Through the use of tools such as Trang[5] and MSV[2] it is possible to convert this to other widely accepted formats such as XSD[6].

```
# Begin Schema

namespace nmwg = "http://ggf.org/ns/nmwg/base/2.0/"

start =
  element nmwg:message {
    Identifier? &
    attribute messageIdRef { xsd:string }? &
    attribute type { "SetupDataResponse" } &
    Parameters? &
    (
      Metadata |
      Data
    )+
  }

Parameters =
  element nmwg:parameters {
    Identifier &
    Parameter+
  }

Parameter =
  element nmwg:parameter {
    attribute name { xsd:string } &
    (
      attribute value { xsd:string } |
```

```
      (
        anyElement |
        text
      )
    )
  }

Metadata =
  element nmwg:metadata {
    Identifier &
    MetadataIdentifierRef? &
    anyElement*
  }

Data =
  element nmwg:data {
    Identifier &
    MetadataIdentifierRef &
    (
      Datum*
    )
  }

Identifier =
  attribute id { xsd:string }

MetadataIdentifierRef =
  attribute metadataIdRef { xsd:string }

anyElement =
  element * {
    anyThing
  }

anyAttribute =
  attribute * { text }

anyThing =
  (
    anyElement |
    anyAttribute |
    text
  )*

Datum =
  element nmwg:datum {
    anyThing*
  }

# End Schema
```

3.2.2 SetupDataResponse Message Analysis

The following is a breakdown of the elements featured in the schema.

3.2.2.1 Message

```
<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              id="message1"
              type="SetupDataResponse">

  <nmwg:parameters />

  <nmwg:metadata />

  <nmwg:data />

</nmwg:message>
```

There is no difference in the structure of this element. The only notable difference is the value of *type* is restricted to be *SetupDataResponse*.

Table 18: Message Element Specifics

Message Element	
localname	message
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, type, messageIdRef
nested elements	parameters, metadata, data
required	yes

3.2.2.2 Parameters

```
<nmwg:parameters xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/" id="parameters1">

  <nmwg:parameter />

</nmwg:parameters>
```

Table 19: Parameters Element Specifics

Parameters Element	
localname	parameters
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id
nested elements	parameter
required	no

There is no difference in the structure of this element.

3.2.2.3 Parameter

```
<nmwg:parameter xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
                name="NAME">VALUE</nmwg:parameter>

<!-- OR -->

<nmwg:parameter xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
                name="NAME" value="VALUE" />
```

Table 20: Parameter Element Specifics

Parameter Element	
localname	parameter
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	name, value
nested elements	text, undefined
required	yes

There is no difference in the structure of this element.

3.2.2.4 Metadata

```
<nmwg:metadata xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
               id="metadata2" metadataIdRef="metadata1" />
```

Table 21: Metadata Element Specifics

Metadata Element	
localname	metadata
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, metadataIdRef
nested elements	undefined
required	yes

There is no difference in the structure of this element. As in Section 3.1.2.4 the contents of this element will vary by service.

3.2.2.5 Data

```
<nmwg:data xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
           id="data2" metadataIdRef="metadata2" />
```

Table 22: Data Element Specifics

Data Element	
localname	data
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	id, metadataIdRef
nested elements	datum
required	yes

This element features only one deviation from that of the *NMC Protocol*: datum elements are expected to reside in the data element, it will be explained in Section 3.2.2.6.

3.2.2.6 Datum

```
<nmwg:datum xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/" />
```

The datum element is the only element that should appear here. This document will not specific attributes or elements for the datum element.

3.2.3 SetupDataResponse Message Example

The following examples demonstrate some of the possible uses and layouts of response messages in the *MA Protocol*.

Table 23: Datum Element Specifics

Datum Element	
localname	datum
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	value, undefined
nested elements	undefined
required	varies

The first example is the most common form of *SetupDataResponse* message containing a single metadata and data pair. This would be indicative of success due to the presence of the *key*.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="SetupDataResponse" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1">

    <nmwg:datum />
    <nmwg:datum />
    <!-- ... -->
    <nmwg:datum />

  </nmwg:data>

</nmwg:message>

<!-- End XML -->
```

The second example is similar, although it features two pairs.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="SetupDataResponse" id="message1">

  <nmwg:metadata id="m1">
    <!-- metadata -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1">

    <nmwg:datum />
    <nmwg:datum />
    <!-- ... -->
    <nmwg:datum />

  </nmwg:data>

  <nmwg:metadata id="m2">
    <!-- another metadata -->
  </nmwg:metadata>

  <nmwg:data id="d2" metadatIdRef="m2">

    <nmwg:datum />
    <nmwg:datum />
    <!-- ... -->
    <nmwg:datum />

  </nmwg:data>

</nmwg:message>
```

```
<!-- End XML -->
```

The final example demonstrates an error condition. Note that this may contain multiple pairs if sent, and it may be possible to have success for some, and errors for others.

```
<!-- Begin XML -->

<nmwg:message xmlns:nmwg="http://ggf.org/ns/nmwg/base/2.0/"
              type="SetupDataResponse" id="message1">

  <nmwg:metadata id="m1">
    <!-- some error -->
  </nmwg:metadata>

  <nmwg:data id="d1" metadatIdRef="m1">

    <nmwg:datum>
      <!-- some error message -->
    </nmwg:datum>

  </nmwg:data>

</nmwg:message>

<!-- End XML -->
```

4 Data Info

4.1 DataInfoRequest Message

4.1.1 DataInfoRequest Message Schema

The following schema is a native description of the request schema as in the RELAX-NG[3] language. Through the use of tools such as Trang[5] and MSV[2] it is possible to convert this to other widely accepted formats such as XSD[6].

```
% INLINESCHEMA="schema/datainfo_request.rnc"
```

4.1.2 DataInfoRequest Message Analysis

4.1.2.1 Message

Table 24: Message Element Specifics

Message Element	
localname	message
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

4.1.2.2 Parameters

Table 25: Parameters Element Specifics

Parameters Element	
localname	parameters
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

4.1.2.3 Parameter

Table 26: Parameter Element Specifics

Parameter Element	
localname	parameter
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

4.1.2.4 Metadata

- -

4.1.2.5 Data

- -

Table 27: Metadata Element Specifics

Metadata Element	
localname	metadata
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

Table 28: Data Element Specifics

Data Element	
localname	data
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

4.1.3 DataInfoRequest Message Example

4.2 DataInfoResponse Message

4.2.1 DataInfoResponse Message Schema

The following schema is a native description of the request schema as in the RELAX-NG[3] language. Through the use of tools such as Trang[5] and MSV[2] it is possible to convert this to other widely accepted formats such as XSD[6].

```
% INLINESCHEMA="schema/datainfo_response.rnc"
```

4.2.2 DataInfoResponse Message Analysis

4.2.2.1 Message

Table 29: Message Element Specifics

Message Element	
localname	message
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

4.2.2.2 Parameters

Table 30: Parameters Element Specifics

Parameters Element	
localname	parameters
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

4.2.2.3 Parameter

Table 31: Parameter Element Specifics

Parameter Element	
localname	parameter
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

4.2.2.4 Metadata

- -

4.2.2.5 Data

- -

Table 32: Metadata Element Specifics

Metadata Element	
localname	metadata
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

Table 33: Data Element Specifics

Data Element	
localname	data
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

4.2.2.6 Key

Table 34: Key Element Specifics

Key Element	
localname	key
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

4.2.2.7 Datum

- -

Table 35: Datum Element Specifics

Datum Element	
localname	datum
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

4.2.3 DataInfoResponse Message Example

5 Measurement Archive Store

5.1 MeasurementArchiveStoreRequest Message

5.1.1 MeasurementArchiveStoreRequest Message Schema

The following schema is a native description of the request schema as in the RELAX-NG[3] language. Through the use of tools such as Trang[5] and MSV[2] it is possible to convert this to other widely accepted formats such as XSD[6].

```
% INLINESCHEMA="schema/measurementarchivestore_request.rnc"
```

5.1.2 MeasurementArchiveStoreRequest Message Analysis

5.1.2.1 Message

Table 36: Message Element Specifics

Message Element	
localname	message
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

5.1.2.2 Parameters

- -

Table 37: Parameters Element Specifics

Parameters Element	
localname	parameters
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

5.1.2.3 Parameter

Table 38: Parameter Element Specifics

Parameter Element	
localname	parameter
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

5.1.2.4 Metadata

Table 39: Metadata Element Specifics

Metadata Element	
localname	metadata
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

Table 40: Data Element Specifics

Data Element	
localname	data
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

5.1.2.5 Data

- -

5.1.3 MeasurementArchiveStoreRequest Message Example

5.2 MeasurementArchiveStoreResponse Message

5.2.1 MeasurementArchiveStoreResponse Message Schema

The following schema is a native description of the request schema as in the RELAX-NG[3] language. Through the use of tools such as Trang[5] and MSV[2] it is possible to convert this to other widely accepted formats such as XSD[6].

```
% INLINESCHEMA="schema/measurementarchivestore_response.rnc"
```

5.2.2 MeasurementArchiveStoreResponse Message Analysis

5.2.2.1 Message

Table 41: Message Element Specifics

Message Element	
localname	message
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

5.2.2.2 Parameters

Table 42: Parameters Element Specifics

Parameters Element	
localname	parameters
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

5.2.2.3 Parameter

Table 43: Parameter Element Specifics

Parameter Element	
localname	parameter
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

5.2.2.4 Metadata

- -

5.2.2.5 Data

- -

Table 44: Metadata Element Specifics

Metadata Element	
localname	metadata
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

Table 45: Data Element Specifics

Data Element	
localname	data
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

5.2.2.6 Key

Table 46: Key Element Specifics

Key Element	
localname	key
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

- -

5.2.2.7 Datum

- -

Table 47: Datum Element Specifics

Datum Element	
localname	datum
namespaces	http://ggf.org/ns/nmwg/base/2.0/
attributes	xxx
nested elements	xxx
required	xxx

5.2.3 MeasurementArchiveStoreResponse Message Example

6 Result Codes

The following new result codes can be incorporated into this extension based on the work in the *NMC Protocol* [7]. We will introduce these into both styles to allow for backwards compatibility. The original style is presented first:

success.
ma.

error.
ma.

We can express the same information using the new URI style:

http://schemas.perfsonar.net/status/
success/
ma/
error/
ma/

7 Notational Conventions

The key words “MUST” “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” are to be interpreted as described in RFC 2119 [1]

8 Security Considerations

There are no security considerations.

9 Contributors

Jason Zurawski
Internet2
1150 18th Street, NW
Suite 1020
Washington, DC 20036

D. Martin Swany (editor)
University of Delaware
Department of Computer and Information Sciences
Newark, DE 19716

10 Intellectual Property Statement

The OGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director.

11 Disclaimer

This document and the information contained herein is provided on an "As Is" basis and the OGF disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

12 Full Copyright Notice

organizations, except as needed for the purpose of developing Grid Recommendations in which case the procedures for copyrights defined in the OGF Document process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the OGF or its successors or assignees.

References

[1] S. Bradner. Key Words for Use in RFCs to Indicate Requirement Levels. RFC 2119, March 1997.

[2] Sun Multi-Schema XML Validator (MSV). `https://msv.dev.java.net/`.

[3] RELAX-NG Schema Language. `http://relaxng.org/`.

[4] M. Swany. An Extensible Schema for Network Measurement and Performance Data. Network measurements working group document, Open Grid Forum, May 2009. https://forge.gridforum.org/projects/nm-wg.

[5] Multi-format schema converter based on RELAX NG. `http://www.thaiopensource.com/relaxng/trang.html`.

[6] XML Schema). `http://www.w3.org/XML/Schema`.

[7] J. Zurawski and M. Swany. An Extensible Protocol for Network Measurement and Control. Network measurement control working group document, Open Grid Forum, May 2009. https://forge.gridforum.org/projects/nmc-wg.