NMC Result Codes

NMC Working Group – September 20 2011, OGF 33

Last Modified September 27, 2011

# Background

Implementations of the perfSONAR protocols (perfSONAR-MDM, perfSONAR-PS) have independently used result and status codes to convey information about the system for many years. These codes are not standard, and do not follow a prescribed format. The current landscape features “developer chosen” messages and formats, and may be returned in situations that do or do not warrant the return of some form of status message.

The NMC-WG (Network Measurement Control Working group), charged with standardizing the protocols used by servers and clients of the perfSONAR framework, has been charged with defining a common standard for error messages and result codes. This document will describe:

* A hierarchy of result codes based loosely on similar efforts in other protocols
* Guidelines for the use of these codes within software
* Guidelines for the types of textual messages that will accompany these codes

These recommendations come from the meeting at OGF 32, were refined at OGF 33, and are subject to the will of the working group.

# Proposed Result Codes

## Syntax

A status code is defined by the following pattern:

STATUS\_CODE =

 “http//schemas.ogf.org/nmc/status/” STATUS\_CATEGORY ”/” STATUS\_NAME ”/” VERSION

STATUS\_CATEGORY =

“informational”

| “successful”

| “redirection”

| “clienterror”

| “servererror”

The following list presents acceptable status names for certain category:

* informational

STATUS\_NAME =

“protocol\_version”

| “data\_limitation”

| “service\_contact”

* clienterror

STATUS\_NAME =

“bad\_message”

| “bad\_request”

| “authentication\_failed”

| “unauthorized”

| ”message\_not\_allowed”

| “event\_type\_not\_allowed”

| “request\_too\_large”

| “request\_timeout”

| “protocol\_not\_allowed”

| “chaining\_not\_understood”

* servererror

STATUS\_NAME =

data\_fetch\_error

 | “too\_busy”

| “administrative\_down”

Two categories, successfulandredirection,  do not need to have certain status names.

VERSION is a string presenting information about the version of protocol, e.g. 201109 or 20110925. In case of early testing version an optional part after "/" may be added (e.g. 201109/beta or 20110925/beta) .

## Status Categories

The following categories were chosen to classify perfSONAR errors. Note that these recommendations form the basis of the standard, extension by implementations may be more specific as required. E.g. we anticipate that an implementation may choose to offer a specific category that further extends something in this spec. For example “http://schemas.ogf.org/nmc/status/servererror/data\_fetch\_error/database\_down/2011/09/” may be an extension of “http://schemas.ogf.org/nmc/status/servererror/data\_fetch\_error/2011/09/”. This does not go against the spirit of this specification. Client applications should be advised that parsing an error code may result in seeing this ‘unexpected’ last part, and could terminate parsing up to this point to avoid a complete failure.

A final note relates to the location of the ‘version’ string, e.g. the date the status was last modified by NMC. Convetional wistom (e.g. GFD 84) notes that namespaces should be constructed with a date in the middle of the string. Experience in perfSONAR has found that this date is more effective at the end, and allows for updates to certain parts of the schematic standard, without re-writing all of them.

### Informational

This represents valid responses for informational requests. Using just the top level, e.g. “http://schemas.ogf.org/nmc/status/informational/2011/09/” is considered to be acceptable. The following subclasses were identified:

* ***protocol\_version***: Returns the version of the NMC protocol in use
* ***data\_limitation***: Returns a message indicating that responses will be limited to a pre-set range or size
* ***service\_contact***: Returns the contact information (e.g. administrative contacts, etc.) for the service

### Successful

This represents valid responses for any form of successful interaction. Using just the top level, e.g. “http://schemas.ogf.org/nmc/status/successful2011/09/” is considered to be the only acceptable response.

### Redirection

This represents valid responses for any form of redirection that the service deems acceptable. Using just the top level, e.g. “http://schemas.ogf.org/nmc/status/redirection/2011/09/” is considered to be the only acceptable response. This redirection activity is assumed to be “temporary”, e.g. clients should not cache/store this redirection for any reason.

### Clienterror

This represents an error issued to a client based on the request. Use of the top level, e.g. “http://schemas.ogf.org/nmc/status/clienterror/2011/09/” may be possible, but is not recommended. The following subclasses were identified:

* ***bad\_message***: Returns a message indicating there is a syntactic (XML based) or semantic (logical structure of request) error. Context will be given in human readable text.
* ***bad\_request***: Request was send to non-existent endpoint on the node/service in question
* ***authentication\_failed***: The service could not determine who the user really was
* ***unauthorized***: The user is not allowed to request the content/resource
* ***message\_not\_allowed***: The wrong type of message was sent to the service (indicates a deeper level of semantic checking beyond ‘bad\_message’)
* ***event\_type\_not\_allowed***: The eventType is not allowed or unsupported by this service (indicates a deeper level of semantic checking beyond ‘bad\_message’)
* ***request\_too\_large***: The request message was too large to process
* ***request\_timeout***: The request has taken too long to service
* ***protocol\_not\_allowed***: Version of NMC protocol was not understood between the client and server. We choose to draw the line at NMC protocol in this case, and not “higher” into NM, or “lower” into SOAP, HTTP, TCP, etc. In the event that we cannot handle schematic nuances beyond NMC (or lower layer issues from something else) we should fall back to the general error, or ‘bad message’
* ***chaining\_not\_understood:*** The chaining used in the message, either merge or operation, was not understood by the parser. This is a specific use case of ‘bad\_message’

### Servererror

This represents an error issued to a client based on the behavior of the service that s serving the request. Using just the top level, e.g. “http://schemas.ogf.org/nmc/status/servererror/2011/09/” is considered to be acceptable, but not recommended. The following subclasses were identified:

* ***data\_fetch\_error***: The request is valid, but there is an underlying problem with the service backend.
* ***too\_busy***: The service is unable to act on the request at this time due to internal limitations on resource consumption
* ***administrative\_down***: The service has been configured to not respond.

# Message structure

## Basic structure

Two elements which characterize the basic massage structure containing a status code are as follow:

* the namespace "http://ggf.org/ns/nmwg/result/2.0/"[[1]](#footnote-1) of xml tags containing status code information,
* the merge chaining between metadata with STATUS\_CODE and data with SHORT\_DESCRIPTION of a status code.[[2]](#footnote-2)

The following xml snippet presents the structure[[3]](#footnote-3):

<nmwg:message

 xmlns:nmwg=http://ggf.org/ns/nmwg/base/2.0/

 xmlns:nmwgr="http://ggf.org/ns/nmwg/result/2.0/">

 <nmwg:metadata id="status-code">

 <nmwg:eventType>

 STATUS\_CODE

 </nmwg:eventType>

 </nmwg:metadata>

 <nmwg:data id="status-code-desc"

 metadataIdRef="status-code">

 <nmwgr:datum>

 SHORT\_DESCRIPTION

 </nmwgr:datum>

 </nmwg:data>

</nmwg:message>

## Extensions

The basic structure of status code message can be extended by introducing new namespacesc (see new-domain-name in the xml snippet below). They allow to redefine the datum element in order to contain more complex information formats.

<nmwg:message

 xmlns:nmwg=http://ggf.org/ns/nmwg/base/2.0/>

 <nmwg:metadata id="status-code">

 <nmwg:eventType>

 STATUS\_CODE

 </nmwg:eventType>

 </nmwg:metadata>

 <nmwg:data id="status-code-desc"

 metadataIdRef="status-code">

 <new-domain-name:datum >

 STATUS\_EXTENDED\_CONTENT

 </new-domain-name:datum>

 </nmwg:data>

</nmwg:message>

# Examples

Example 1:

<nmwg:message id="response"

 type="EchoResponse"

 xmlns:nmwg=http://ggf.org/ns/nmwg/base/2.0/

 xmlns:nmwgr="http://ggf.org/ns/nmwg/result/2.0/">

 <nmwg:metadata id="status-code">

 <nmwg:eventType>

 http://schemas.ogf.org/nmc/status/successful/2.0/

 </nmwg:eventType>

 </nmwg:metadata>

 <nmwg:data id="status-code-desc"

 metadataIdRef="status-code">

 <nmwgr:datum>

 This is the success echo response from the service.

 </nmwgr:datum>

 </nmwg:data>

</nmwg:message>

Example 2:

<nmwg:message id="resp1"

 type="MetadataKeyResponse"

 xmlns:nmwg=http://ggf.org/ns/nmwg/base/2.0/

 xmlns:nmwgr="http://ggf.org/ns/nmwg/result/2.0/">

 <nmwg:metadata id="status-code">

 <nmwg:eventType>

 http://schemas.ogf.org/nmc/status/servererror/data\_fetch\_error/2.0

 </nmwg:eventType>

 </nmwg:metadata>

 <nmwg:data id="status-code-desc"

 metadataIdRef="status-code">

 <nmwgr:datum>

 Requested metadata items are not available.

 </nmwgr:datum>

 </nmwg:data>

</nmwg:message>

Example 3:

<nmwg:message id="resp1"

 type="EchoResponse"

 xmlns:nmwg=http://ggf.org/ns/nmwg/base/2.0/

 xmlns:*echo*="*http://ggf.org/ns/nmwg/result/echo/2.0/*">

 <nmwg:metadata id="status-code">

 <nmwg:eventType>

 http://schemas.ogf.org/nmc/status/successful/2.0/

 </nmwg:eventType>

 </nmwg:metadata>

 <nmwg:data id="status-code-desc"

 metadataIdRef="status-code">

 <*echo*:datum>

 <!-- *the content depends on the type of this datum element*

 *which is indicated by its namespace* -->

 </*echo*:datum>

 </nmwg:data>

</nmwg:message>

# Next Steps

After ratification, these items will be included in the base protocol document.

1. Version number in the namespace may change in future. [↑](#footnote-ref-1)
2. STATUS\_CODE and SHORT\_DESCRIPTION are the variables. In real messages are replaces by strings. [↑](#footnote-ref-2)
3. The message identifier and the message type are omitted intentionally for simplicity. [↑](#footnote-ref-3)