

# globus scheduler event generator

## 2.1

Generated by Doxygen 1.6.1

Sat Feb 13 13:47:47 2010

# Contents

|          |  |          |
|----------|--|----------|
| <b>1</b> | <b>Globus Scheduler Event Generator</b>  | <b>1</b> |
| <b>2</b> | <b>SEG Protocol</b>                      | <b>1</b> |
| 2.1      | Message Types . . . . .                  | 2        |
| 2.1.1    | 001 - Job State Change . . . . .         | 2        |
| <b>3</b> | <b>Module Index</b>                      | <b>2</b> |
| 3.1      | Modules . . . . .                        | 2        |
| <b>4</b> | <b>Module Documentation</b>              | <b>2</b> |
| 4.1      | Scheduler Implementation API . . . . .   | 2        |
| 4.1.1    | Detailed Description . . . . .           | 3        |
| 4.1.2    | Enumeration Type Documentation . . . . . | 3        |
| 4.1.3    | Function Documentation . . . . .         | 3        |

## 1 Globus Scheduler Event Generator

The Scheduler Event Generator (SEG) is a program which uses scheduler-specific monitoring modules to generate job state change events. At the SEG level, the state change events correspond to changes in any jobs which are managed by the scheduler, even if they do not correspond to jobs initiated by the Managed Job Service. These state change events are propagated to the Job State Monitor.

Depending on scheduler-specific requirements, the SEG may need to run with privileges to enable it to obtain scheduler event notifications. As such, one SEG runs per scheduler resource. For example, on a host which provides access to both PBS and fork jobs, two SEGs, running at (potentially) different privilege levels will be running.

When executed, the SEG is able to start issuing events from some time in the past. The SEG will, in general, not require any persistent state between invocations. One SEG instance exists for any particular scheduled resource instance (one for all homogeneous PBS queues, one for all fork jobs, etc).

The SEG is implemented in an executable called the globus-scheduler-event-generator, located in the Globus Toolkit's libexec directory. It is invoked with the following command line:

```
globus-scheduler-event-generator -s SCHEDULER NAME [-t TIMESTAMP]
```

It produces events in the format described in the [SEG Protocol](#) section of this document on the standard output of the process.

When begun, it loads the scheduler module for the scheduler named on the command line and then defers to it for most functionality. When it detects an error writing to stdout or reading stdin, it terminates. The scheduler specific code uses the SEG API to emit events to the JSM.

Scheduler implementations use the [SEG API](#) to send messages to the JSM.

## 2 SEG Protocol

The general form for the SEG protocol messages is

MESSAGE-TYPE;TIMESTAMP;message-type-specific content

- *MESSAGE-TYPE* is a three-digit integer. The JSM will parse the message contents based on the message type.
- *TIMESTAMP* is an unsigned value indicating seconds since the UNIX epoch.

## 2.1 Message Types

### 2.1.1 001 - Job State Change

Message Format: 001;TIMESTAMP;JOBID;STATE;EXIT\_CODE

Message Type Specific Content:

**JOBID** local scheduler-specific job id

**STATE** new job state (integer as per the GRAM protocol constants)

**EXIT\_CODE** job exit code if STATE is done or failed.

## 3 Module Index

### 3.1 Modules

Here is a list of all modules:

**Scheduler Implementation API**

[2](#)

## 4 Module Documentation

### 4.1 Scheduler Implementation API

Scheduler-specific SEG module implementations use this API to issue events to the Job State Monitor.

#### Enumerations

- enum [globus\\_scheduler\\_event\\_generator\\_error\\_t](#) {  
    [GLOBUS\\_SEG\\_ERROR\\_TYPE\\_NULL](#) = 1024,  
    [GLOBUS\\_SEG\\_ERROR\\_TYPE\\_ALREADY\\_SET](#),  
    [GLOBUS\\_SEG\\_ERROR\\_TYPE\\_INVALID\\_MODULE](#),  
    [GLOBUS\\_SEG\\_ERROR\\_TYPE\\_INVALID\\_FORMAT](#),  
    [GLOBUS\\_SEG\\_ERROR\\_TYPE\\_OUT\\_OF\\_MEMORY](#),  
    [GLOBUS\\_SEG\\_ERROR\\_TYPE\\_LOADING\\_MODULE](#) }

## Functions

- globus\_result\_t [globus\\_scheduler\\_event](#) (const char \*format,...)
- globus\_result\_t [globus\\_scheduler\\_event\\_pending](#) (time\_t timestamp, const char \*jobid)
- globus\_result\_t [globus\\_scheduler\\_event\\_active](#) (time\_t timestamp, const char \*jobid)
- globus\_result\_t [globus\\_scheduler\\_event\\_failed](#) (time\_t timestamp, const char \*jobid, int failure\_code)
- globus\_result\_t [globus\\_scheduler\\_event\\_done](#) (time\_t timestamp, const char \*jobid, int exit\_code)
- globus\_result\_t [globus\\_scheduler\\_event\\_generator\\_get\\_timestamp](#) (time\_t \*timestamp)

### 4.1.1 Detailed Description

Scheduler-specific SEG module implementations use this API to issue events to the Job State Monitor. Events occur whenever a job is placed in the scheduler's queue (PENDING), begins execution (ACTIVE), terminates successfully (DONE), or ends abnormally (FAILED).

A SEG module should register an event with the Globus event driver (most likely using either the Globus Callback or Globus XIO interfaces) in its activation function and then return. All events should be triggered from callbacks. When the SEG detects that it should terminate, it will deactivate the SEG module it started. The SEG module should wait for any outstanding callbacks to subside and before returning from its deactivation function to ensure that all events will be properly dispatched. After deactivation is complete, the SEG will unload the shared module and terminate.

### 4.1.2 Enumeration Type Documentation

#### 4.1.2.1 enum globus\_scheduler\_event\_generator\_error\_t

Error types used by the SEG.

##### Enumerator:

**GLOBUS\_SEG\_ERROR\_TYPE\_NULL** NULL Parameter.

**GLOBUS\_SEG\_ERROR\_TYPE\_ALREADY\_SET** Already called a one-time function.

**GLOBUS\_SEG\_ERROR\_TYPE\_INVALID\_MODULE** Shared module missing descriptor.

**GLOBUS\_SEG\_ERROR\_TYPE\_INVALID\_FORMAT** Invalid printf format for SEG protocol message.

**GLOBUS\_SEG\_ERROR\_TYPE\_OUT\_OF\_MEMORY** Out of memory.

**GLOBUS\_SEG\_ERROR\_TYPE\_LOADING\_MODULE** Unable to load scheduler module.

### 4.1.3 Function Documentation

#### 4.1.3.1 globus\_result\_t globus\_scheduler\_event (const char \**format*, ...)

Send an arbitrary SEG notification.

##### Parameters:

*format* Printf-style format of the SEG notification message

... Varargs which will be interpreted as per format.

##### Return values:

**GLOBUS\_SUCCESS** Scheduler message sent or queued.

***GLOBUS\_SEG\_ERROR\_NULL*** Null format.

***GLOBUS\_SEG\_ERROR\_INVALID\_FORMAT*** Unable to determine length of formatted string.

#### 4.1.3.2 **globus\_result\_t globus\_scheduler\_event\_pending (time\_t *timestamp*, const char \* *jobid*)**

Send a job pending event to the JobSchedulerMonitor implementation.

**Parameters:**

***timestamp*** Timestamp to use for the event. If set to 0, the time which this function was called is used.

***jobid*** String indicating the scheduler-specific name of the job.

**Return values:**

***GLOBUS\_SUCCESS*** Scheduler message sent or queued.

***GLOBUS\_SEG\_ERROR\_NULL*** Null jobid.

***GLOBUS\_SEG\_ERROR\_INVALID\_FORMAT*** Unable to determine length of formatted string.

#### 4.1.3.3 **globus\_result\_t globus\_scheduler\_event\_active (time\_t *timestamp*, const char \* *jobid*)**

Send a job active event to the JobSchedulerMonitor implementation.

**Parameters:**

***timestamp*** Timestamp to use for the event. If set to 0, the time which this function was called is used.

***jobid*** String indicating the scheduler-specific name of the job.

**Return values:**

***GLOBUS\_SUCCESS*** Scheduler message sent or queued.

***GLOBUS\_SEG\_ERROR\_NULL*** Null jobid.

***GLOBUS\_SEG\_ERROR\_INVALID\_FORMAT*** Unable to determine length of formatted string.

#### 4.1.3.4 **globus\_result\_t globus\_scheduler\_event\_failed (time\_t *timestamp*, const char \* *jobid*, int *failure\_code*)**

Send a job failed event to the JobSchedulerMonitor implementation.

**Parameters:**

***timestamp*** Timestamp to use for the event. If set to 0, the time which this function was called is used.

***jobid*** String indicating the scheduler-specific name of the job.

***failure\_code*** Failure code of the process if known.

**Return values:**

***GLOBUS\_SUCCESS*** Scheduler message sent or queued.

***GLOBUS\_SEG\_ERROR\_NULL*** Null jobid.

***GLOBUS\_SEG\_ERROR\_INVALID\_FORMAT*** Unable to determine length of formatted string.

#### 4.1.3.5 globus\_result\_t globus\_scheduler\_event\_done (time\_t *timestamp*, const char \**jobid*, int *exit\_code*)

Send a job done event to the JobSchedulerMonitor implementation.

##### Parameters:

*timestamp* Timestamp to use for the event. If set to 0, the time which this function was called is used.

*jobid* String indicating the scheduler-specific name of the job.

*exit\_code* Exit code of the process if known.

##### Return values:

**GLOBUS\_SUCCESS** Scheduler message sent or queued.

**GLOBUS\_SEG\_ERROR\_NULL** Null jobid.

**GLOBUS\_SEG\_ERROR\_INVALID\_FORMAT** Unable to determine length of formatted string.

#### 4.1.3.6 globus\_result\_t globus\_scheduler\_event\_generator\_get\_timestamp (time\_t \* *timestamp*)

Get the timestamp for the earliest event an SEG module should send.

##### Parameters:

*timestamp* Pointer to a time\_t which will be set to the timestamp passed to the SEG executable. The module should not send any events which occur prior to this timestamp.

##### Return values:

**GLOBUS\_SEG\_ERROR\_NULL** Null timestamp.

**GLOBUS\_SUCCESS** Timestamp value updated. If the timestamp was not set on the SEG command-line, then the value pointed to by *timestamp* will be set to 0.

## Index

GLOBUS\_SEG\_ERROR\_TYPE\_ALREADY\_SET  
    [seg\\_api, 3](#)  
GLOBUS\_SEG\_ERROR\_TYPE\_INVALID\_-  
    FORMAT  
    [seg\\_api, 3](#)  
GLOBUS\_SEG\_ERROR\_TYPE\_INVALID\_-  
    MODULE  
    [seg\\_api, 3](#)  
GLOBUS\_SEG\_ERROR\_TYPE\_LOADING\_-  
    MODULE  
    [seg\\_api, 3](#)  
GLOBUS\_SEG\_ERROR\_TYPE\_NULL  
    [seg\\_api, 3](#)  
GLOBUS\_SEG\_ERROR\_TYPE\_OUT\_OF\_-  
    MEMORY  
    [seg\\_api, 3](#)  
globus\_scheduler\_event  
    [seg\\_api, 3](#)  
globus\_scheduler\_event\_active  
    [seg\\_api, 3](#)  
globus\_scheduler\_event\_done  
    [seg\\_api, 4](#)  
globus\_scheduler\_event\_failed  
    [seg\\_api, 4](#)  
globus\_scheduler\_event\_generator\_error\_t  
    [seg\\_api, 3](#)  
globus\_scheduler\_event\_generator\_get\_timestamp  
    [seg\\_api, 4](#)  
globus\_scheduler\_event\_pending  
    [seg\\_api, 3](#)

### Scheduler Implementation API, [2](#)

[seg\\_api](#)  
    GLOBUS\_SEG\_ERROR\_TYPE\_ALREADY\_-  
        SET, [3](#)  
    GLOBUS\_SEG\_ERROR\_TYPE\_INVALID\_-  
        FORMAT, [3](#)  
    GLOBUS\_SEG\_ERROR\_TYPE\_INVALID\_-  
        MODULE, [3](#)  
    GLOBUS\_SEG\_ERROR\_TYPE\_LOADING\_-  
        MODULE, [3](#)  
    GLOBUS\_SEG\_ERROR\_TYPE\_NULL, [3](#)  
    GLOBUS\_SEG\_ERROR\_TYPE\_OUT\_OF\_-  
        MEMORY, [3](#)  
[seg\\_api](#)  
    globus\_scheduler\_event, [3](#)  
    globus\_scheduler\_event\_active, [3](#)  
    globus\_scheduler\_event\_done, [4](#)  
    globus\_scheduler\_event\_failed, [4](#)  
    globus\_scheduler\_event\_generator\_error\_t, [3](#)  
    globus\_scheduler\_event\_generator\_get\_-  
        timestamp, [4](#)  
    globus\_scheduler\_event\_pending, [3](#)