



CrossGrid User Guide

Task 3.3.1: OCM-G

Task 3.3 – Grid Monitoring

Document Filename:	cg-wp3.3-ocmg-userguide
Workpackage:	Task 3.3 – Grid Monitoring
Partner(s):	CYF, (TUM)
Lead Partner:	CYF
Config ID:	cg-wp3.3-ocmg-userguide-v0.1
Document classification:	PUBLIC

Abstract: This is the user guide for the Grid application monitoring system OCM-G, developed within the CrossGrid project.



Delivery Slip

	Name	Partner	Date	Signature
From	Bartosz Baliś	CYFRONET AGH	Aug 2004	
Verified By				
Approved By				

Document Log

Version	Date	Summary of changes	Author
0-9	Oct 21st, 2004	First draft version	Bartosz Baliś
1-0	Nov 15th, 2004	First official release	Bartosz Baliś, Roland Wismüller

Contents

CopyrightNotice	4
1 Introduction	5
2 Product Usage	6
2.1 OCM-G Overview	6
2.2 Preparing the Application for Monitoring	8
2.3 Starting the OCM-G	9
2.4 Submitting the application	10
2.5 Starting the G-PM tool	10
2.6 Using the OCM-G with Migrating Desktop	10
3 Support for performance measurement	13
3.1 Application-specific events: using probes	13
4 Auxiliary Tools	15
4.1 OCM-G Compiler Wrapper	15
4.2 OCM-G Console Tool	15
4.3 OCM-G Tracer	16
4.4 Instrumentation Tool	17
4.5 Checking Application for Monitoring Support	18
5 Contact Information and Credits	19
6 GNU General Public License	20

Copyright Notice

Copyright (c) 2005 by **ACC CYFRONET AGH, Krakow, Poland; AGH University of Science and Technology, Krakow, Poland; Technische Universität München, Germany; Universität Siegen, Germany**. All rights reserved.

Use of this product is subject to the terms and licenses stated in the GPL license agreement. Please refer to Chapter 5 for details.

This research is partly funded by the European Commission IST-2001-32243 Project CrossGrid.

Use of this product is subject to the terms and licenses stated in the GPL license agreement. Please refer to attached license for details.

This research is partly funded by the European Commission IST-2001-32243 Project CrossGrid.

1 Introduction

The OCM-G (OMIS-Compliant Monitoring system for the Grid) is an application monitoring system which supports both cluster and Grid environments based on Globus 2.4, in particular the CrossGrid (and DataGrid) testbeds.

The purpose of the OCM-G is to provide on-line information about a running parallel / distributed application to application-development-support tools, specifically performance analysis tools, like the G-PM tool. Using the information from the OCM-G, tools are enabled to obtain performance measurements of the monitored application, related to, for example, delay and volume of communication, CPU usage, etc.

This document is intended for application developers who would like to monitor their applications using the OCM-G monitoring system and an OCM-G-compliant tool, for example the G-PM performance measurement tool.

For information concerning installation and running of the OCM-G, refer to **OCM-G Installation Guide**. Tool developers who would like to learn how to construct performance metrics based on the OMIS monitoring services provided by the OCM-G, please refer to the **OCM-G Developer's Guide**.

2 Product Usage

2.1 OCM-G Overview

Fig. 2.1 shows the run-time components of OCM-G, their distribution across the different kind of machines in the Grid, and their communication topology.

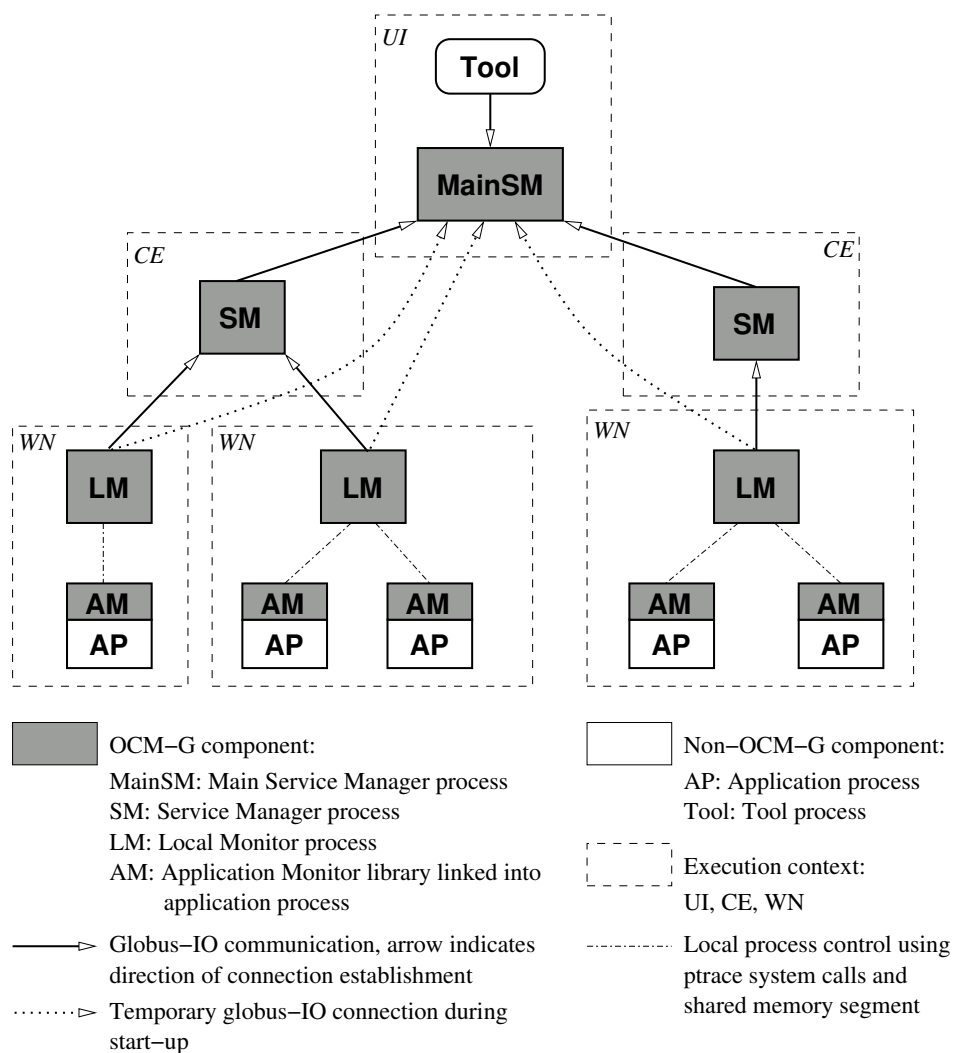


Figure 2.1: Components of OCM-G at run-time

The OCM-G is a distributed system composed of three types of components.

1. Service Managers (SM) – one for each site of the target system. Started automatically on site's computing element (CE) machine.
2. Local Monitors (LM) – one for each host of the target system. Started automatically on worker nodes (WN) where there are application processes running.

3. Main Service Manager (MainSM) – one for each grid user in the system. Currently started by the user manually on the machine of his choice, usually the User Interface (UI) machine.

Fig. 2.1 shows the distribution of these components in a sample grid environment. Note that Local Monitors handle application processes running on Worker Nodes. The user can launch a tool, such as the G-PM performance analyzer for grid applications, which connects to the OCM-G via the MainSM.

Tools connect to the OCM-G and communicate with it via a standard interface OMIS (On-line Monitoring Interface Specification), which is documented in detail in [?]. In short, OMIS defines a set of monitoring services which are categorized into three classes:

1. information services – used to obtain specific information concerning the target system or monitored application. Examples are:
 - return information about target node (OS info, host name, etc.),
 - return information about process(es) (pid, name of the executable, etc.),
 - return the list of processes for the given application.
2. manipulation services – allow for manipulation on the target system or monitored application. These services are useful, among others, for debuggers, for example:
 - suspend/continue target process(es)
 - write process' memory
 - create/increment a counter
3. event services – used to monitor events of interest inside the application, for example:
 - process has started a function call (e.g. `MPI_Send`)
 - process has terminated

Information and manipulation services are sometimes referred to as actions, since they specify some actions to be performed, while event services only define some events we would like to detect.

The mentioned services are basic building blocks to specify **monitoring requests** for the OCM-G. A request is simply a service or a combination of services parameterized with concrete objects to operate on and perhaps additional parameters to specify the exact behaviour of the service(s).

Additional parameters may specify the concrete piece of information to be returned about a node, the concrete sequence of bytes to write to the process' memory and the address of the memory, the number by which a counter should be incremented, etc.

There two important types of requests we can issue: unconditional and conditional ones.

- The unconditional services are composed of one or more information and/or manipulation services. Thus, they specify a set of actions to be performed immediately (unconditionally). This can be described as the **query** monitoring mode: we query for a piece of information (or request some manipulations) and obtain the result immediately.
- The conditional requests are composed of an event service and a set of actions. In this case, the request tells the monitoring system to execute the actions not immediately, but conditionally: each time the specified event occurs subsequently. This, in turn, is the **subscribe** monitoring mode: we subscribe for some information (related to an event) and get it whenever it occurs.

To monitor an application with the OCM-G the following steps are needed:

- the application has to be properly prepared

- the Main Service Manager must be running
- the application has to be submitted for execution, supplied with additional command-line parameters

The following sections describe these issues.

2.2 Preparing the Application for Monitoring

2.2.1 Adding code for registering the application in the OCM-G

To enable registering of the application processes in the OCM-G, the following function call must be inserted at the very beginning of the code (but after `MPI_init` in case of MPI applications):

```
ocmg_register(global_id, &argc, &argv)
```

Where `global_id` is a user-chosen global identification number, while `argc` and `argv` are pointers to parameters passed to the `main` function. The prototype of `ocmg_register` is

```
int ocmg_register(int global_id, int *argc, char ***argv);
```

The function returns 0 if and only if registration was successful.

The purpose of `global_id` is the subsequent convenient identification of the process in a tool. For this reason, the `global_id` should be unique for each process, although OCM-G itself does not require this. In case of MPI applications it makes most sense to pass the global rank of the process here.

NOTE: In the current version of the OCM-G there is no need to insert the call to `ocmg_register` in MPI applications, provided that the instrumented version of the MPI library is linked to the application's executable (see section 4.5 how to check it). In this case, registration is done automatically when `MPI_Init` is called.

2.2.2 Linking the application

In order to monitor the application with OCM-G, it must be linked in a special way. Linking is supported by a utility program (`cg-ocmg-cc`) which is part of the OCM-G distribution. In order to use the OCM-G with an application, the user should issue the usual link command (i.e. the compiler call which creates the final executable) preceded by `cg-ocmg-cc`. For example, for a program usually compiled and linked with `mpicc simple.c -o simple`, the user should invoke:

```
cg-ocmg-cc mpicc simple.c -o simple
```

This will result in linking the application with the monitoring libraries and (in the example) an instrumented version of the MPI library.

2.2.3 Additional command-line parameters

When an application has been linked as explained above, it accepts a few additional, OCM-G specific command-line parameters. Some of them must be passed to the application in order to enable monitoring, while some additional ones are optional. Currently these parameters are as follows.

1. `--ocmg-appname name` (required)
This parameter is necessary to pass to the OCM-G the application name **name**, which is an arbitrary name given by the user to identify the application. Later the user provides the same name to a tool to specify which application he wants to monitor.
If this parameter is not passed to the application, it will not register in the OCM-G. This means that if the application is started in its usual way, it will execute without OCM-G.
2. `--ocmg-mainsm connstring` (required)
This parameter is necessary to pass to the OCM-G the connection string **connstring** returned by the Main Service Manager after it has been executed (see Section 2.3).
3. `--ocmg-regcont` (optional)
This optional parameter changes the default start-up behaviour of the application. By default, the application processes are initially put in a stopped state. They only continue running when the monitoring system gets an explicit request from a tool to start the program (`thread_continue` OMIS service request). However, if the user wants the processes to continue running immediately, he should pass this parameter.
4. `--ocmg-dontreg` (optional)
If this option is passed **after** `--ocmg-appname`, the application will not register in the OCM-G. The monitoring is not possible in this case. The user can use this option if he linked his application with OCM-G support, but does not want to monitor it.
Note that an easier way to achieve the same behavior is not to provide the `--ocmg-appname` option.
5. `--ocmg-debug` (optional)
This optional parameter turns on some debugging messages. It may be helpful if some problems occur.

2.3 Starting the OCM-G

With the exception of the Main Service Manager, the OCM-G components start up automatically. This section explains how to start the Main SM manually. If you are using CrossGrid Migrating Desktop to submit your application, please skip to section 2.6 **Using the OCM-G with Migrating Desktop**.

First we have to choose where to start the Main SM. Usually we choose the local User Interface machine, which we also use to submit jobs, run the G-PM tool, etc. To start the Main SM one should type:

```
cg-ocmg-monitor
```

Note that you must have a valid Globus proxy certificate before you run `cg-ocmg-monitor`. You can create the proxy certificate by invoking `grid-proxy-init`.

Also note that the Main SM opens a port for incoming connections, thus the machine it is running on requires inbound connectivity for the ports within the range defined by the environment variable `GLOBUS_TCP_PORT_RANGE`. If this is not the case for our UI machine, we might run the Main SM on some Computing Element, for example `ce01.cyf-kr.edu.pl`. This can be done via the `globus-job-run` command:

```
globus-job-run ce01.cyf-kr.edu.pl cg-ocmg-monitor
```

Once the Main SM started successfully, it returns a connection string, for example:

```
MainSM Connection String: 7f000001:8026
```

The Main SM connection string must be passed as a parameter to the application (section 2.2.3).

2.4 Submitting the application

Once the application is prepared as described in Sections 2.2.1 and 2.2.2 and the Main SM is running, the application can be submitted to run on the Grid. There are several ways to do that, for example:

- with the `globusrun` command, which requires an `rs1` file with job description,
- with the `edg-job-submit` command, which requires an `jd1` file with job description.

NOTE: if you are using the Migrating Desktop to submit the application, please skip to section 2.6: **Using the OCM-G with Migrating Desktop.**

In any case, we should pass the special OCM-G-specific parameters to the application. The parameters are described in section 2.2.3. In our example, the parameters could be:

```
--ocmg-appname myapp – arbitrary name to identify the application,  
--ocmg-mainism 7f000001:8026 – MainSM connection string, essential for the correct start-up of  
the OCM-G.
```

2.5 Starting the G-PM tool

To attach the G-PM tool to the OCM-G, we must pass the Main SM connection string, so that the tool can connect to the OCM-G, and the application name, so that G-PM knows which application to monitor:

```
cg-gpm myapp --ocmg-mainism 7f000001:8026
```

See the G-PM User’s Manual for more details.

2.6 Using the OCM-G with Migrating Desktop

The CrossGrid Migrating Desktop is a user front-end to submit jobs, copy files between storage elements, run tools, etc. Fig. 2.2 shows the OCM-G plug-in to the Migrating Desktop which is part of the application job submission interface.

Using the plug-in the user does not need to manually start the Main SM, pass parameters, or even start the G-PM tool.

In the text box **Application name** at the top of the window, the user must insert the arbitrarily chosen application name. The user can start the Main SM manually (as explained in section 2.3) and insert the Main SM connection string in the **OCM-G connection string** field. Alternatively, he can press the button **Launch OCM-G MainSM now!** to have the Migrating Desktop automatically launch the Main SM, obtain the returned connection string, and insert it in the **OCM-G connection string** field.

By default, the Migrating Desktop will start the Main SM on the user’s local computer. If it should be started on some other machine, the user must provide the proper user (login) name and host name in the optional box **SSH user and host for OCM-G** in the format *user@host*, e.g.:

```
someone@ce01.cyf-kr.edu.pl
```

Application name:

OCM-G connection string:

After start-up application should:

continue execution

wait for continue command

SSH user and host for OCM-G (optional):

Launch OCM-G MainSM now!

Number of process for G-PM to wait (optional):

Launch G-PM now!

OK **Cancel**

Figure 2.2: OCM-G Migrating Desktop Plug-in

Using the two radio-buttons in the middle pane, the user might like to change the behaviour of the application after its start-up. By default, the application is stopped and waits for a command from the G-PM tool to continue running. If the user wants to make the application run right away after start-up, he should choose the **continue execution** option.

The edit box **Number of processes for G-PM to wait** may be used to pass the expected number of application processes of the G-PM tool. This allows to start G-PM even before submitting the application, since G-PM will wait for the specified number of processes having registered before it starts up its user interface.

Finally, if all parameters has been set up, the user can start the G-PM tool by pushing the button **Launch G-PM now!**.

NOTE: Starting the Main SM and the G-PM tool via the Migrating Desktop currently only works if the user's local machine is using the Linux operating system. If the local computer is running Windows, the Main SM and G-PM have to be started manually on a remote machine, as described in Sections!2.3 and 2.5.

3 Support for performance measurement

The OCM-G provides the following features to enable performance measurements of applications:

1. Performance analysis objects: counters, integrators (supporting the measurement of time intervals) or trace buffers (for recording of arbitrary event information) and related services for creating and manipulating these objects.
2. The basic event-action mechanism, which allows to define events to be detected and actions to be executed after the event's occurrence. For example: *when MPI_Send is executed, increment counter 5.*
3. Probe mechanism to support detection of application-specific events.
4. G-PM extension which contains services that allow to satisfy some more sophisticated features of the G-PM tool.

Topics (1), (2) and (4) are important for tool developers who would like to implement performance metrics based on the OMIS services. Topics (1) and (2) are covered in the OMIS specification (see the basics of OMIS and the specification of Performance Analysis Extension). Topic (4) is described in the **OCM-G Developer's Guide**.

Topic (3) is important for application's developers who would like to perform advanced performance measurements on his application taking advantage of custom application-specific metrics which can be defined in the G-PM tool. We describe this topic in the following section.

3.1 Application-specific events: using probes

Probes are events (more exactly: event triggers) inserted at an arbitrary point in the source code of the application. Their purpose is to support the detection of application-specific events, which can be used in the G-PM tool to define application-specific metrics. Currently probes are just function calls.

If the user wants to use probes in his application he needs to perform the following steps:

1. Define the probe functions in a separate C file, e.g., 'probes.c'. The function names are arbitrary. The function must have at least one parameter of type `int`, to which the virtual time is passed (see G-PM User's Guide). Optionally, other parameters may follow. These parameters must either have type `int` or `double`. They can be used to pass application-specific data to the probe. The body of the functions must be empty. For example, the probe-file might contain:

```
// probes.c

void probe_start_iteration(int vt) { }
void probe_end_iteration(int vt, double residuum) { }
```

NOTE: In the current version of OCM-G, each function definition must be provided in a single line!

2. To use probes in the application, the user just manually inserts calls to probe functions in the source code. In the following example, probes are inserted in such a way as to generate events at the beginning and at the end of each iteration of a for loop:

```
// ... some code here ...

for (i = 0; i < n; ++i) {
    probe_start_iteration(i);

    // ... some computations here ...
    // ... residuum is some value computed in the loop ...

    probe_end_iteration(i, residuum);
}

// ... some code here ...
```

Note that the value of loop iterator variable `i` has been passed as the first parameter to both probe functions. This parameter defines a virtual time, which is used to determine event occurrences that belong together, like beginning and end of the same loop iteration. While the only restriction for this parameter is that the sequence of values is monotonically increasing, the iteration number is the most natural choice.

Note that these two events: beginning and end of iteration will allow to compute application-specific metrics such as *computation time per iteration* (difference of time stamps for the two events), *communication volume per iteration* (volume of communication measured between the two events), etc. The G-PM tool provide means to define such metrics based on the probes inside the application.

3. In the linking step outlined in Section 2.2.2, an additional parameter “`--probes probe-file`” must be passed, which specifies the name of the file where the probes are defined. For example:

```
cg-ocmg-cc --probes probes.c mpicc simple.c -o simple.o
```

See also the OMIS specification [?] for documentation of the `thread_executes_probe` event service which allows to detect probe events.

4 Auxiliary Tools

4.1 OCM-G Compiler Wrapper

One of the essential auxiliary tools provided with the OCM-G is the compile-wrapper `cg-ocmg-cc`, which is necessary to enable monitoring of any application. The usage of the tool (also discussed in sections 2.2.2 and 3.1) is very simple: the usual command issued to compile the application is preceded by `cg-ocmg-cc`, for example:

```
cg-ocmg-cc gcc -o app app.c
cg-ocmg-cc mpicc ping.c -o ping
```

The `cg-ocmg-cc` changes the command line of the compiler so that necessary monitoring libraries, and an instrumented version of the MPI library (if available) are linked to the application's executable. The `cg-ocmg-cc` has been tested to work correctly with `gcc` and `mpicc` (mpich distribution). However, the tool is expected to work correctly with many other C/C++ compilers, since in fact only the commands for the linker are affected.

If probes are used in the application (see section 3.1), the `-probes` option should be used to pass the name of the file, where the probes are defined. For example:

```
cg-ocmg-cc -probes probes.c mpicc ping.c -o ping
```

Note that in this case the application is actually recompiled (not only relinked) due to additional instructions (calls to probe functions) inserted in the source code of the application.

Please note that the `cg-ocmg-cc` tool looks for the OCM-G monitoring libraries and the instrumented MPI library in a specific location, namely under the `CG_LOCATION` directory. If they are not found, the compilation fails. The value of `CG_LOCATION` is evaluated in the following order:

1. If file `.etc.sysconfig.cg` is found in the user's home directory, it is sourced as a shell script, since it is expected to contain a command which exports the `CG_LOCATION` environment variable, for example:

```
export CG_LOCATION=$HOME/cg
```

2. If file `/etc/sysconfig/cg` is found, the same procedure as above is used for this file.
3. if 1. and 2. fail, the default value is used: `/opt/cg`.

4.2 OCM-G Console Tool

The OCM-G distribution provides a simple OMIS-compliant tool `cg-ocmg-tool`, which can be used to connect to the OCM-G and issue requests directly in OMIS. The tool, like any other OCM-G-compliant tool, requires the `--ocmg-mainism` option which should contain the connection string to Main Service Manager (see section 2.3 and 2.5). For example:

```
cg-ocmg-tool --ocmg-mainism 7f000001:8026
```

If the tool has successfully connected to the OCM-G, it displays a prompt sign 'ocm.1>', and we may issue unconditional or conditional OMIS requests, exactly as specified by the OMIS specification. The tool will display corresponding OMIS replies. An example session with the tool might look as follows:

```
ocm.1> :app_attach2("ping")
```

```
Got reply 1:
```

```
Element 0:
          |  0 |
Element 1:
  app_1 |  0 |
```

```
ocm.2> :app_get_proclist([app_1])
```

```
Got reply 2:
```

```
Element 0:
          |  0 |
Element 1:
  app_1 |  0 | 2, [n_2, p_29026_n_2, n_1, p_21598_n_1]
```

```
ocm.3> quit
```

In this session, two unconditional OMIS requests were issued: to attach to an application called “ping”, and to obtain a list of processes for this application. The `quit` command can be used to exit the tool.

Optionally, on starting the tool we may pass it the application’s name as a parameter, for example:

```
cg-ocmg-tool --ocmg-mainsm 7f000001:8026 ping
```

As a result, the two requests showed in the example above (and a few others) will be performed automatically. This is useful, since at the beginning of each session, we usually first attach to the application, then get the list of processes, and so on.

4.3 OCM-G Tracer

The OCM-G distribution includes a simple tool, `cg-ocmg-tracer`, which creates an event trace of some MPI related events in an application’s execution, using the SDDF format¹. The tool is provided primarily as an example and as a simple means to test the proper installation of OCM-G. It is not really meant as a tool for the user, however, it may nevertheless be useful.

The tool traces the calls to the following MPI routines: `MPI_Barrier`, `MPI_Bcast`, `MPI_Reduce`, `MPI_Send`, `MPI_Bsend`, `MPI_Ssend`, and `MPI_Recv`. A trace entry is generated for the beginning and the end of these functions. The event entry includes a time stamp (seconds and microseconds), the process that created the event, and some event specific parameters. The trace is printed to `stdout`, but can be stored in a file by using the shell’s output redirection.

The `cg-ocmg-tracer` tool accepts the following command line arguments:

¹See <http://vibes.cs.uiuc.edu/Project/SDDF/SDDFOverview.htm>

-
- app_name*** (required) – the application name passed to the application via the `--ocmg-appname` option (see Section 2.2.3).
 - `--ocmg_mainsm connstring` (required) – specifies the connection string to the OCM-G Main SM. This string is printed by the Main SM when it starts up (see Section 2.3).
 - `--num-procs num` (optional) – This option allows to specify the number of processes in the monitored application. When this number is provided, the `cg-ocmg-tracer` tool will wait for the specified number of processes to have registered in the OCM-G, before it starts monitoring. In this way, the tool can be started even before the application is submitted.
 - `--terminate` (optional) – terminate the OCM-G after the `cg-ocmg-tracer` tool terminates.
 - `-v` (optional) – verbose flag. When this option is provided, the `cg-ocmg-tracer` tool will print all OMIS requests it sends to OCM-G.
 - `-t read_delay` (optional) – the time in seconds between two requests for reading trace entries from OCM-G. Trace events are buffered locally in OCM-G and transferred to `cg-ocmg-tracer` only when requested. This parameter controls the frequency of these transfers. It allows advanced users to control the overhead of trace collection. The default is an interval of 5 seconds.
 - `-n num_entries` (optional) – the maximum number of trace entries which should be read from the OCM-G in one request. This parameter allows advanced users to control the overhead of trace collection. The default is that an unlimited amount of trace entries can be read with a single request to OCM-G.
 - `-s` (optional) – read the trace from OCM-G whenever a process has been stopped. This option is useful if the `cg-ocmg-tracer` tool is used in combination with a debugger.

A typical invocation of the tool looks like:

```
cg-ocmg-tracer myapp --num-procs 4 --terminate --ocmg-mainsm 7f000001:8026
```

After the `cg-ocmg-tracer` tool started up completely, it will send a `thread_continue` request to the OCM-G in order to start the application program (in case it was submitted without the `--ocmg-regcont` parameter, see Section 2.2.3). It then periodically issues requests to the OCM-G to read the event trace and prints it to `stdout`. When the application terminates, the `cg-ocmg-tracer` tool will terminate, too.

4.4 Instrumentation Tool

The OCM-G distribution provides a small tool `cg-ocmg-mpich-instr` which can be used to prepare an instrumented version of the MPI library.

To obtain useful information about running MPI application, a modified – so called instrumented – version of the MPI library must be linked to the application’s executable. The instrumentation of the MPI library enables to monitor MPI library calls which is essential, e.g., for most performance metrics supported by the G-PM tool.

Usually, the instrumented version of the MPI library should be installed with the OCM-G, and it should be automatically linked to the application using the OCM-G compilation wrapper `cg-ocmg-cc` (see section 4.1). You can also check if your application’s executable is linked with an instrumented version of the MPI library using the `cg-ocmg-check-app` tool (see section 4.5).

If it turns out that the instrumented MPI library is not available, we can create it simply calling the tool:

```
cg-ocmg-mpich-instr
```

The tool will try to find where the MPI distribution is installed in the system, and it will use the MPI library found there to create an instrumented version of this library (source codes are not necessary!) Note that the instrumented MPI library is saved under the `CG_LOCATION` directory tree. If `CG_LOCATION` is evaluated to the default directory `/opt/cg`, the administrator privileges are needed to successfully run the tool. You can also provide your own location of this directory using the `--cg-location` option, for example:

```
cg-ocmg-mpich-instr --cg-location $HOME/cg
```

Note that if the `--cg-location` option is not provided, the `CG_LOCATION` directory is evaluated in the same way as for the `cg-ocmg-cc` wrapper (see section 4.1).

4.5 Checking Application for Monitoring Support

The OCM-G distribution provides a small utility, which can be used to check whether an application executable has been properly prepared for the use with OCM-G. This tool is useful when there are problems like the application is not registering with OCM-G.

In particular, the `cg-ocmg-check-app` utility checks whether

- the executable is linked with the OCM-G monitoring library. This is the prerequisite for the application to be monitored at all.
- the executable is linked with instrumented versions of the MPICH libraries (`libmpich.a` or `libmpich2.a`). If these libraries are missing, the application can be monitored, but any monitoring related to MPI calls (esp. MPI-related measurements in the G-PM tool) will not work.
- the executable includes probes. The utility is not able to determine the names of these probes, but at least, it can verify that there are some.

The `cg-ocmg-check-app` utility just accepts one argument: the name of the executable program it should check. For example:

```
cg-ocmg-check-app BStream_1
```

This command will print something like

```
Application is linked with OCM-G
Instrumented libraries: libmpich.a
Probes files: probes.c
```

If the command prints “Application is NOT linked with OCM-G” it means that the executable is not linked properly for the use with OCM-G (or at least that the utility is not able to verify this, since e.g. the executable’s symbol tables have been stripped). The output “NO instrumented libraries” means that the instrumented MPI libraries have not been linked with the application (either due to some problem, or simply because it is not an MPI application). Finally, the message “NO probes” indicates that no probes have been found in the executable.

5 Contact Information and Credits

You may want to provide a list of people involved in the development of this software and provide a feedback channel (typically e-mail) for bug reporting and commenting on the product. This section is optional.

6 GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation’s software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author’s protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors’ reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone’s free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:
 - a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
 - b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
 - c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
 - a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions

of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.> Copyright (C)
19yy <name of author>
```

```
This program is free software; you can redistribute it and/or modify it under the terms
of the GNU General Public License as published by the Free Software Foundation; either
version 2 of the License, or (at your option) any later version.
```

```
This program is distributed in the hope that it will be useful, but WITHOUT ANY
WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS
FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License along with this
program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge,
MA 02139, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author Gnomovision comes with
ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and
you are welcome to redistribute it under certain conditions; type 'show c' for details.
```

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than 'show w' and 'show c'; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision'
(which makes passes at compilers) written by James Hacker.
```

```
<signature of Ty Coon>, 1 April 1989 Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Bibliography

[QAP] WP5, CYRFRONET; **Quality Assurance Plan**; Evolving document