

# **Skel 1.0 User's Manual**

**July 2012**

## DOCUMENT AVAILABILITY

Reports produced after January 1, 1996, are generally available free via the U.S. Department of Energy (DOE) Information Bridge:

**Web site:**<http://www.osti.gov/bridge>

Reports produced before January 1, 1996, may be purchased by members of the public from the following source:

National Technical Information Service

5285 Port Royal Road

Springfield, VA 22161

**Telephone:**703-605-6000 (1-800-553-6847)

**TDD:**703-487-4639

**Fax:**703-605-6900

**E-mail:**[info@ntis.fedworld.gov](mailto:info@ntis.fedworld.gov)

**Web site:**<http://www.ntis.gov/support/ordernowabout.htm>

Reports are available to DOE employees, DOE contractors, Energy Technology Data Exchange (ETDE) representatives, and International Nuclear Information System (INIS) representatives from the following source:

Office of Scientific and Technical Information

P.O. Box 62

Oak Ridge, TN 37831

**Telephone:**865-576-8401

**Fax:**865-576-5728

**E-mail:**[reports@adonis.osti.gov](mailto:reports@adonis.osti.gov)

**Web site:**<http://www.osti.gov/contact.html>

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

# **Skel 1.0 User's Manual**

Prepared for the  
Office of Science  
U.S. Department of Energy  
July 2012

Prepared by  
Oak Ridge National Laboratory  
Oak Ridge, Tennessee 37831-6070  
managed by  
UT-Battelle, LLC  
for the  
U.S. Department of Energy  
under contract DE-AC05-00OR22725

# Contents

<b>1</b>	<b>Introduction</b>	<b>6</b>
<b>2</b>	<b>Using Skel</b>	<b>7</b>
2.1	Overview . . . . .	7
2.2	Detailed Example . . . . .	8
<b>3</b>	<b>Skel Command Reference</b>	<b>9</b>
3.1	Available Subcommands . . . . .	9
<b>4</b>	<b>The Parameters File</b>	<b>10</b>
4.1	Elements . . . . .	10
<b>5</b>	<b>Skel Settings File</b>	<b>12</b>
5.1	Available Settings . . . . .	12
<b>6</b>	<b>Low-Level Timing Mechanism</b>	<b>13</b>
6.1	Using the Low-Level Timing Mechanism . . . . .	13
6.2	Extracting Timing Information . . . . .	13
<b>7</b>	<b>Hints for Porting Skel</b>	<b>14</b>
7.1	Makefiles . . . . .	14
7.2	Submission Scripts . . . . .	14

# 1 Introduction

Skel is a tool used for the generation of I/O skeletal applications. Skel takes an I/O descriptor and a set of parameters, and generates a full I/O benchmark. Compared to a scientific application, the generated benchmark is easy to build and to run. Despite the simplicity of the generated code, it mimics quite well the I/O pattern from the target application, thus simplifying the process of understanding the I/O performance of an application.

Skel provides a simple mechanism for testing the performance of I/O operations *that are relevant to applications of interest*. This is of critical importance when evaluating new systems or new system configurations. It is equally useful for evaluating new I/O methods, or examining the effects of different parameters to existing I/O methods. Since the benchmarks produced by skel focus on the exact I/O patterns of applications, performance measurements obtained from those benchmarks will correlate highly with the performance of the actual applications.

This manual provides a detailed explanation of Skel, including the relevant file formats, documentation of skel commands, common usage of skel, and hints for porting skel to new platforms. Skel is young and still in development, so we expect there will be things that do not work perfectly. Please help us improve by letting us know when you encounter troubles. You can reach the skel developers by sending email to [lot@ornl.gov](mailto:lot@ornl.gov).

## 2 Using Skel

### 2.1 Overview

Figure 1 shows the typical workflow of using `skel` to create a skeletal I/O benchmark. The example uses the GTS application, and thus the workflow begins with `gts.xml`, the XML descriptor from GTS. The `skel xml` command is used to create a second xml file, `gts_skel.xml` which will serve as the ADIOS xml descriptor *for the skeletal application*. Next, `skel params` is used to generate a parameters file. The generated parameters file is then edited by the user to guide the subsequent generation of the skeletal application.

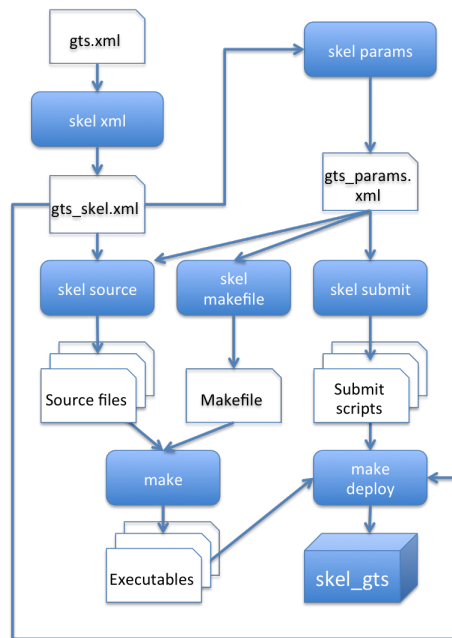


Figure 1: Skel Workflow

At this point, the commands `skel source`, `skel makefile`, and `skel submit` may be used to generate the source files, Makefile, and submission scripts that comprise the skeletal application. With all of the components of the skeletal application created, it is now time to build the application, using `make`, and finally deploy the application to a directory from which it may be launched, using `make deploy`, which copies the `gts_skel.xml` file, executable files, and submission script. The user now has a ready to run I/O benchmark without having written any source code at all.

## 2.2 Detailed Example

In this section we will describe the steps used to create an I/O Kernel based on the GTS application. The ADIOS config file for GTS can be found in the examples directory.

Coming Soon.

## 3 Skel Command Reference

Most skel commands are of the form:

```
skel subcommand project
```

### 3.1 Available Subcommands

**skel install** Copies skel configuration files to the user's home directory.

Usage: **skel install**

**skel makefile** Generates a Makefile fit for building and deploying the skeletal application.

Requires `<project>_skel.xml` and `<project>_params.xml`

Usage: **skel makefile** `<project>`

**skel params** Generates a parameters file that can be customized by the user. Note that this command creates a file called `<project>_params.xml.default` so as to avoid overwriting a customized parameters file. This means that the user should copy this file to `<project>_params.xml` and edit before proceeding with code generation.

Requires `<project>_skel.xml`

Usage: **skel params** `<project>`

**skel source** Generates a C or Fortran code that performs the I/O operations described by the XML descriptor and the parameters file.

Requires `<project>_skel.xml` and `<project>_params.xml`

Usage: **skel source** `<project>`

**skel submit** Generates a submission script for the skeletal application

Requires `<project>_skel.xml` and `<project>_params.xml`

Usage: **skel submit** `<project>`

**skel xml** Generates the `<project>_skel.xml` file.

Requires `<project>.xml`

Usage: **skel xml** `<project>`



## 4 The Parameters File

Generation of a skeletal application requires some more information that is found in the ADIOS configuration file. To specify this additional information, the user must supply a Skel parameters file. The parameters file is an XML file which contains the elements described in the following section.

Although it not overly difficult to construct your own parameters files, users may find it more convenient to use the `skel params` command to automatically generate a parameters file with default values, then simply edit the file to provide the desired configuration.

### 4.1 Elements

**<skel-config>** Each parameters file must contain exactly one **<skel-config>** element as the only element at the root level of the document. The **<skel-config>** element should contain one **<adios-group>** element and one **<batch>** element, as described below.

*Supported Attributes:*

**application** The name of the application described by the document. This should correspond to the *project* used for the `skel` calls, and the name of the original XML file given to `skel xml`.

**<adios-group>** The **<adios-group>** element is a child of the root **<skel-config>** element. It corresponds to the adios-group that will be written by the I/O skeletal application being generated. The **<adios-group>** element contains a collection of **<scalar>** and **<array>** elements corresponding to the variables described for the group in the ADIOS config file.

*Supported Attributes:*

**name** The name of the ADIOS group that this element describes

**<scalar>** Represents a scalar variable that is described in the ADIOS descriptor. Often integer valued scalars will be used to determine the dimensions of arrays, thus providing appropriate values is essential for creating meaningful benchmarks.

*Supported Attributes:*

**name** The name of the scalar variable

**type** (Optional) The type of this variable in the generated code. Supplied for convenience by `skel params`.

**value** A value, of the proper type, that will be assigned to this scalar variable in the generated code.

**<array>** Represents an array variable that is described in the ADIOS descriptor.

*Supported Attributes:*

**name** The name of the array variable

**type** (Optional) The type of this variable in the generated code. Supplied for convenience by `skel params`.

**dimensions** (Optional) A list of the scalar values that will be used to determine the dimensions of this array

**fill-method** Determines how the array memory will be initialized in the generate code

<**batch**> Describes a set of tests to be performed by the generated I/O kernel. The <batch> element contains one or more <test> elements.

*Supported Attributes:*

**name** The name of the batch. Used to name the submission script and other elements of the batch job.

**cores** The number of MPI tasks that will be used for the skeletal application

**walltime** Amount of runtime requested by the generated submission script

<**test**> Describes a single test to be performed.

*Supported Attributes:*

**type** The type of test to be performed. Currently only `write` is supported.

**group** The ADIOS group to be written

**method** The ADIOS write method to use for writing (i.e. POSIX). A full listing of available methods can be found in the ADIOS manual.

**iterations** The number of times to repeat this test

**rm** determines whether and when to remove the written output files. One of {pre, post, both, none}

## 5 Skel Settings File

Skel provides a collection of configurable options which are exposed in the Skel settings file. The settings file is located in the user's home directory at `~/.skel/settings`.

The settings file consists of single line entries of the form `<name>=<value>`. Blank lines and lines starting with `#` are ignored. Entries in the settings files are case sensitive, and lower case is typically used for all names and yes/no values.

### 5.1 Available Settings

**deploy\_dir** This is the directory into which the compiled applications will be copied to be executed. This directory should be visible to the compute nodes.

**submit\_target** This determines which submission script template will be used by `skel submit`. Templates for `jaguar` and `sith` are included in this release.

**sleep\_before\_open (yes/no)** Determines whether a short `sleep` statement is inserted in the code before the `adios_open` call. Default is `no`.

**barrier\_before\_open (yes/no)** Determines whether an `MPI_Barrier` call is inserted into the generated code before `adios_open`. Default is `yes`.

**barrier\_before\_access (yes/no)** Determines whether an `MPI_Barrier` call is inserted into the generated code before the sequence of `adios_write` calls. Default is `no`.

**barrier\_before\_close (yes/no)** Determines whether an `MPI_Barrier` call is inserted into the generated code before `adios_close`. Default is `no`.

**barrier\_after\_close (yes/no)** Determines whether an `MPI_Barrier` call is inserted into the generated code after `adios_close`. Default is `no`.

**barrier\_after\_steps (yes/no)** Determines whether a single `MPI_Barrier` call is inserted into the generated code after the `finaladios_close`. Default is `no`.

**use\_adios\_timing (yes/no)** Determines whether a call is inserted to output detailed timing information collected by ADIOS. For this to work, your adios distribution must have been configured using `--enable-skel-timing`. Default is `no`.

## 6 Low-Level Timing Mechanism

By default, applications generated by skel will produce a summary timing report, sending it to standard out. On most platforms, this will be captured in the output file produced by the job script. The summary report contains textual information about the overall time taken by the various I/O operations. If more detail is desired, ADIOS has a mechanism for gathering low-level timing information for various events that occur within the ADIOS I/O calls.

### 6.1 Using the Low-Level Timing Mechanism

To use the low-level timing mechanism, you must use an ADIOS library that has been built with this low-level timing mechanism enabled. Simply build ADIOS as described in the ADIOS manual, inserting `--enable-skel-timing` in the `configure` command. We do not recommend enabling the low-level timing mechanism while running production codes.

Once you have enabled low-level timing in ADIOS, you only need to enable generation of low-level timing calls in your Skel settings file. This is done by including the line: `use_adios_timing=yes` in your Skel settings file. This will cause the `skel source` command to include an additional call near the end of the generated skeletal application to output the detailed timing information that has been collected. The detailed timing information is written to a separate file using XML. This will work best if used to write a single iteration of a single group.

The low-level timing mechanism provides detailed timing information for only some of the available write methods. As of this release, the supported methods are `POSIX`, `MPI_LUSTRE`, and `MPI_AMR`.

### 6.2 Extracting Timing Information

The XML file that is produced by ADIOS contains a large amount of measurement data, but it is somewhat unwieldy to work with directly. So, we have included an additional utility, `skel_extract.py`, which allows data from the XML file to be exported as a CSV file.

## 7 Hints for Porting Skel

Skel has been developed and tested on only a small handful of platforms. While we expect most functionality will be portable to a wider range of machines, there are likely to be some issues arising when running skel on your system. There are a few hints in the following sections that may help you to get started. If further assistance is needed to get skel working, please contact [lot@ornl.gov](mailto:lot@ornl.gov).

### 7.1 Makefiles

Assuming that they work properly, Makefiles generated by skel are quite convenient, as the skel user need not think about how to compile the code, but can simply type `make`. All of the details are taken care of by skel. We have tested skel on only a few systems at this point, and thus it is quite possible that the Makefile generated by skel may fail on some systems. Users familiar with `make` with a need to adjust some aspect of the generated makefiles should investigate `~/.skel/templates/Makefile.default.tpl`. This template file is used by Skel to generate Makefiles, and can be adjusted to the needs of the user. The template syntax is simple, with `$$VAR_NAME$$` used to indicate template substitutions to be made by `skel makefile`. Again, if you run into trouble with this, please contact us as described above.

### 7.2 Submission Scripts

Similar to the Makefile generation, skel uses templates to generate submission scripts for the generated applications. The submission templates are also located in `~/.skel/templates/`, and are named `submit_<target>.tpl`, where `<target>` corresponds to the `submit_target` defined in the user's settings file. To create a new `submit_target`, simply copy one of the existing template files, and rename it with the desired `submit_target` name. Then, adjust the submission syntax so that the generated files work properly with the submission mechanism on your system. Once again, if you run into trouble with this, please contact us as described above.