

# AstroGrid-D Data Management

## Tutorial<sup>1</sup>

The AstroGrid-D Data Management (ADM), designed and written by Thomas Brüsemeister in the late summer of 2007, provides a virtual filesystem and simultaneously cares for the placement of the corresponding physical files on one or more storage facilities. ADM is a tool for distributed data-management [1]. The term "AstroGrid-D Data Management" is equivalent to the notion of the "virtual filesystem" and both are used interchangeably in this text.

ADM uses a relational database, namely PostgreSQL, to store a unique descriptor, i.e. a Logical File Identifier (LFID) for each file, as well as meaningful or typical meta data for each file or directory, e.g. the owner and a timestamp to log when the entry has been registered with the filesystem. Whereas file ownership and creation timestamp are mandatory meta data and ADM transparently cares for their maintenance, individual files can be endowed with custom (user-defined) properties. ADM provides the command line client `adm`, including a C-library, which offers an easy-to-use access to the stored files. Furthermore, ADM ships with a web interface which permits to browse the virtual filesystem graphically. The following pages are dedicated to guide AstroGrid-D users through the usage of the command line interface to the AstroGrid-D Data Management.

## 1 Introduction and Basic Usage

The AstroGrid-D Data Management (ADM) provides a set of commands that allow to add, delete or move/rename files and directories. In the style of the Subversion syntax, an ADM *command* is composed of two "words" and one or more arguments:

```
# Basic adm command structure
adm <subcommand> argument(s)
```

The first word is always `adm`, the ADM client program, or tersely speaking, the *client*. The second word is the actual instruction, alias *subcommand*, according to the client messages, that indicates the operation to be carried out on the virtual filesystem, e.g. `list` in order to display the contents of a directory. Usually, an `adm` subcommand has one or two arguments, except for `adm help` which has either none or a single argument as well as the always no-argument command `adm info`. First off, if invoked without arguments `adm help` displays a list of all `adm` commands available:

```
# Display available adm commands
agrid064@alnitak:~$ adm help

adm command-line client, version 0.2.0
  compiled on Mar 10 2008, 08:55:58

adm <subcommand> [options] [args]
Type 'adm help <subcommand>' for help on a specific subcommand.
```

---

<sup>1</sup>*Editorial note:* Please send e-mail concerning this text to [rwahner@ari.uni-heidelberg.de](mailto:rwahner@ari.uni-heidelberg.de) (documentation) and e-mail about technical aspects of ADM to [tbruese@ari.uni-heidelberg.de](mailto:tbruese@ari.uni-heidelberg.de) (development).

```
Available subcommands:
  add (put)
  edit (ed)
  find
  get
  info
  link (copy, cp, ln)
  list (ls)
  mkdir
  move (mv, ren, rename)
  propdel (pdel, pd)
  propget (pget, pg)
  proplist (plist, pl)
  propset (pset, ps)
  remove (rm, del, delete)
  replicate (rep)
  resolve (res)
  rmdir
```

ADM is a tool for distributed data-management.  
Copyright (c) 2007-2008 Thomas Bruesemeister, ZAH.

*Remark:* `adm help` can have an optional argument, namely a `subcommand`, in which case it shows a more detailed documentation for `subcommand`; see below. When the requested subcommand is misspelled or does not exist, `adm help` falls back to the above no-argument output.

At the time of this writing (05/2008), there is just one subcommand left, that has not yet been included in the current version of the client, namely `adm locate`, to *quickly* look up files and directories in the virtual filesystem. For the time being, the present client provides the slower but more reliable `find` subcommand. The behavior of `adm find` and `adm locate` generally reflects the corresponding Unix or Linux commands, where `locate` relies on a regularly refreshed database, while `find` examines the given subtree of the filesystem in its current state each time it is invoked. Since `locate` can "see" only what the filesystem contained when its database was recently refreshed, newly created files and directories are invisible to `locate` until the next database update happens. On the other hand, `locate` is quite fast, because it merely queries its database instead of browsing the whole filesystem. Since `find` analyzes a complete subtree of the filesystem entry-by-entry, i.e. `find` is recursive by default, it unearths all files matching the search criterion right at the moment of its invocation and therefore usually consumes more time.

According to the above output most ADM subcommands provide an abbreviated two-character-version, e.g. `adm ls vfs-path` abbreviates `adm list vfs-path`, where "vfs" is short hand for "virtual filesystem". The `list (ls)` subcommand displays the content of the directory specified by `vfs-path`:

```
# Show root directory
agrid064@alnitak:~$ adm ls /

adm-tutorial/
astrogrid/
home/
incoming/
lost+found/
```



Figure 1: ADM-Webinterface on *http://alnitak.ari.uni-heidelberg.de:12000*.

adm

Even though it means more typing, the code examples in this text use the more distinct non-abbreviated notation. Table 1 shows all subcommands at hand with the current client together with the short hand syntax, if available. Note, that following Unix and Linux habits, the root directory of the virtual filesystem is `/`. Moreover, all ADM commands need absolute paths, except for `adm info` which is the only no-argument command. Like its shell counterpart the `ls` command has an `-l` command line flag in order to show more elaborate information compared to the sole file- or directory name:

```
# Show root directory (verbose output)
agrid064@alnitak:~$ adm list -l /

d nGrid/OU=ZAH/CN=Ralf Wahner      0 2008-01-08 10:25 adm-tutorial/
d ZAH/CN=Thomas Bruesemeister     0 2007-12-07 17:32 astrogrid/
d ZAH/CN=Thomas Bruesemeister     0 2007-12-07 17:31 home/
d ZAH/CN=Thomas Bruesemeister     0 2007-12-07 17:32 incoming/
d ZAH/CN=Thomas Bruesemeister     0 2008-03-05 09:51 lost+found/
d nGrid/OU=ZAH/CN=Ralf Wahner     0 2008-01-08 13:14 performance-scalability/
s                                ADM 0 2007-12-07 17:18 adm
7 entries
```

The leftmost column indicates the file type of the entries, where lowercase `d` is assigned to directories whereas lowercase `f` denotes a file. `ADM` is a particular directory internally used by ADM for administrative purposes and therefore has type `s` in order to be distinguishable from normal directories. The second column displays the file owner, compiled from the attribute mnemonics found in the users proxy certificate and truncated for the sake of readability. The two-character

keywords are defined in the Lightweight Directory Access Protocol (LDAP) specification and mean: common name (CN), organization (O), organizational unit (OU) and country (C). The third column shows the file size in bytes and intentionally vanishes for directories. Finally, the fourth column indicates date and time when the entries were created.

Due to its presetting, `adm list` displays all entries in the specified directory, regardless of the ownership. The `-u` flag suppresses all files and directories other than those owned by the user invoking the `adm ls` command:

```
# Show root directory (verbose output)
agrid064@alnitak:~$ adm list -l /

d nGrid/OU=ZAH/CN=Ralf Wahner    0 2008-01-08 10:25 adm-tutorial/
d nGrid/OU=ZAH/CN=Ralf Wahner    0 2008-01-08 13:14 performance-scalability/
7 entries (5 filtered)
```

The output of `adm help list` demonstrates how to access the built-in documentation for an ADM command and summarizes the previous two examples:

```
# Show build-in help for "list"
agrid064@alnitak:~$ adm help list

Lists files and directories in the virtual filesystem.

usage:  list vfs-path

Valid Options:
  -l [--long]           : use a long listing format
  -u [--userdn-matches] : shows only entries matching your userdn

Example:  adm ls -l /home
```

By the way, as the above output shows, each `adm help <subcommand>` contains an example how to use this subcommand.

## 2 Comit and Retrieve Files

*Remark:* All operations in this text are supposed to take place in the `/adm-tutorial` directory; see above output of `adm list -l /` on page 2. A valid file or directory name, with respect to the virtual filesystem implemented by the AstroGrid-D Data Management, may contain uppercase and lowercase latin letters, underscores, plus (+) and minus (-) signs as well as dots (.), in other words any string matching the regular expression `^[a-zA-Z_+-.]+/`.

At the beginning of the brief round tour about file management with ADM, a new directory `/adm-tutorial/vfs_tour` is created by means of `adm mkdir`:

```
# How to create a new directory
agrid064@alnitak:~$ adm mkdir /adm-tutorial/vfs_tour
```

According to Unix or Linux habits, a successful ADM command normally does not display a message. Again, `adm list` verifies, that the new directory now exists:

```
# Show directory /tutorial (verbose output)
agrid064@alnitak:~$ adm list -l /tutorial

d nGrid/OU=ZAH/CN=Ralf Wahner 0 2008-01-08 10:16:42 vfs_tour/
1 entries
```

Files are committed to ADM by means of `adm add`, which is equivalent to `adm put`. Assume that the current working directory contains the file `my_jobdescription.jsdl`. This file is then delegated to ADM by

```
# How to register a file with ADM
agrid064@alnitak:~$ adm add -v my_jobdescription.rsl /adm-tutorial

Source: file:///home/Aggrid/agrid064/
Dest:   gsiftp://alnitak.ari.uni-heidelberg.de/opt/d-grid/adm/fs01/
        my_jobdescription.rsl -> aab3c89633c6af44407ecedeb98f4fb5
        1743 bytes                0.01 MB/sec avg                0.01 MB/sec inst
```

Usually, `adm add` operates quiet. The above detailed output is due to providing the `-v` flag with the command invocation. The target location with respect to the virtual filesystem can be a directory path without trailing file name or a fully qualified file name. Note, that the client accepts absolute paths, only, without exception, i.e. for all subcommands. If invoked without trailing file name, `adm add` implicitly appends the basename of the physical file to the path with respect to the virtual filesystem, whereas in the latter case, the file can in one step be put under ADM control and renamed. The cryptic string `aab3c89633c6af44407ecedeb98f4fb5` is generated by applying the "Message-Digest Algorithm 5" on the file content, i.e. `md5sum my_jobdescription.rsl`, and represents the name ADM uses internally to unequivocally identify `my_jobdescription.jsdl` among the other files registered with ADM.

By default, `add` behaves "non-recursive", i.e. it handles just one file and no directories at a time. In order to enable handling of whole filesystem subtrees, `add` and several other subcommands (see table 2) own the `-r` flag. The following command was used to register the  $\text{\LaTeX}$  source of this tutorial text with ADM:

```
# How to register a filesystem subtree with ADM (ignore invisible entries)
agrid064@alnitak:~$ adm add -r adm-tutorial /adm-tutorial/latex-source
```

Note, that the `/adm-tutorial/latex-source` directory is supposed to exist before invoking the above command. If invoked with the `-r` flag, according to its presetting, `add` ignores files and directories with leading dot (`.`) in their name, i.e. "invisible" files and directories, e.g. `.bashrc` or the `.svn` directories when the sourcecode underlies version control via Subversion. This default behavior can be overridden by means of additionally providing the `-a` flag, which simply tells the client to consider the dotted filesystem entries as well:

```
# How to register a filesystem subtree with ADM (include invisible entries)
agrid064@alnitak:~$ adm add -r -a adm-tutorial /adm-tutorial/latex-source
```

The `-p` flag for `add` allows to specify the number of so-called parallel streams to use for the file transfer and is directly handed over to the homonymous flag of the underlying `globus-url-copy` command. It is recommended to keep the presetting of four streams unchanged, i.e. to not use

adm-Subcommand	Options	Description
add	-a, -b, -p, -r, -s, -v	Register a file with ADM. See <code>mkdir</code> for directories.
copy (cp)	no options	Create a new link to an already registered file. (Files only.)
delete (del)	-r, -v	Unregister a file from ADM. See <code>rmdir</code> for directories.
edit (ed)	no options	Allows in-situ editing on a registered file without download.
find	no options	Search for files and directories matching a pattern.
get	-b, -p, -r, -s, -v	Download a file or directory registered with ADM.
info	no options	Print status and properties of the client ( <code>adm</code> ).
link (ln)	no options	See <code>copy (cp)</code>
list (ls)	-l, -u	Print the contents of a directory. (Directories only.)
<del>locate</del>		<del>Not yet implemented; see page 2</del>
mkdir	no options	Register a new directory with ADM.
move (mv)	no options	Change the location or name of a file or directory.
propdel (pdel, pd)	no options	Unregister a property from a file registered with ADM.
propget (pget, pg)	no options	Retrieve a property value.
proplist (plist, pl)	no options	Show the properties registered for a file.
propset (pset, ps)	no options	Register a property, i.e. a name-value pair, for a file registered with ADM. (Files only)
put	-a, -b, -p, -r, -s, -v	See <code>add</code>
remove (rm)	-r, -v	See <code>delete (del)</code>
rename (ren)	no options	See <code>move (mv)</code>
replicate (rep)	-b, -s	Creates a replica for a file registered with ADM. (Files only.)
resolve (res)	-a, -f, -l	Lücke
rmdir	no options	Unregister an <i>empty</i> directory from the virtual filesystem.

**Table 1:** Overview of the ADM-subcommands.

-p, because experience has proven that four streams care for the best transfer capacity across the internet.<sup>2</sup> The `add` subcommand owns two more flags, namely `-s` and `-b`. Understanding these flags which also appear serveral other subcommands (again, see table 2), requires understanding the notion of the file-space. Therefore, introducing `-s` and `-b` is deferred to the next subsection *File-space Concept*.

Files and directories in the virtual filesystem can be moved from one place to another by means of `adm move`, abbreviated by `adm mv`. This command has always two arguments, namely the entry to be moved and the target file or directory. Given, that `/adm-tutorial/vfs_tour` has a subdirectory `jsdl` the following command will change the location of `my_jobdescription.jsdl` from `/adm-tutorial/vfs_tour` to the new subdirectory:

<sup>2</sup>See section *Performance Options, "How do I pick a value?"* under [3] for a short discussion on data transmission with parallel streams and how to choose the number of connections.

Option	Occurrence	Description
-a [--all]	add, put	Include files and directories when their names has a leading dot (invisible files/directories).
-a [--all]	resolve	Show also replicas on inactive file-spaces.
-b [--fallback]	add, get, put, replicate	Try to access an alternative file-space if available and give up otherwise.
-f [--file]	resolve	Name of the file where all occurrences of <code>adm://</code> are supposed to be substituted by physical file names.
-p [--parallel-streams]	add, get, put	Specify how many parallel streams to use for the data transfer. Default is 4 streams.
-r [--recursive]	add, delete, get, put, remove	Apply command to the subtree of the filesystem given by the command argument.
-s [--file-space]	add, get, put	Access the specified file-space only and give up immediately if the file-space is unavailable.
-v [--verbose]	add, delete, get, put, remove	Show verbose output for the command at hand.
-l [--long]	list	Show verbose information about files and directories, e.g. file owner and file size.
-u [--userdn-matches]	list	Show only files and directories owned by the user who invoked the <code>list</code> command.

**Table 2:** Options of the ADM-subcommands. Each one-character option has a corresponding long version. Except for `-a` which has different meanings for `add` (`put`) and `resolve`, the meaning of the options is consistent for all subcommands.

```
# Move a file to a different directory
agrid064@alnitak:~$ adm move /adm-tutorial/vfs_tour/my_jobdescription.rsl \
                        /adm-tutorial/vfs_tour/jdsl

# Verify that the file has successfully been relocated
agrid064@alnitak:~$ adm ls -l /adm-tutorial/vfs_tour/jdsl

f nGrid/OU=ZAH/CN=Ralf Wahner 1743 2008-01-08 23:49:08 my_jobdescription.rsl
1 entries
```

The content of the new directory is listed to immediately confirm, that the move operation has occurred. The same command is used to change file and directory names within the virtual filesystem. Again, `adm help` denotes, that `adm move`, abbreviated by `adm mv`, is the same as `adm rename`, abbreviated by `adm ren`.

### 3 File-space Concept

Where does a *physical* file reside, after it has been registered with AstroGrid-D Data Management? ADM subcommands that actually transfer files in either direction between grid accounts and storage facilities, i.e. the aforementioned `adm` (`add|put`) and the below described `adm replicate` and `adm get` have two additional flags, namely `-s` and `-b`, which allow to specify a so-called *file-space*. From the users perspective, a file-space is a large amount of disk space with

a unique identifier provided by a member of the AstroGrid-D community, that can be accessed to store scientific data. Each client can individually select a *default file-space*. The client talks to a file-space by means of its Uniform Resource Locator (URL).

Currently ADM owns three file-spaces, one at the "Center for Astronomy of Heidelberg" (3.64 Terabytes) and two at the "Astrophysical Institute Potsdam" (2×1.73 Terabytes), as the always no-argument command `adm info` certifies:

```
# Show status information about ADM (including all file-spaces available)
agrid064@alnitak:~$ adm info

ADM service information, URL: http://alnitak.ari.uni-heidelberg.de:12000
      Version: 0.2.0-dev, $Revision: 278 $
      Protocol: ADM/0.9
      Service uptime: 26 days 21:18:52
      File-spaces: 3 [3 up 0 down]
      LFIDs: 2122
      Directories: 82
      Replicas: 2141
      MRU cache (size/hits/misses): 256/6834/2224
      Path lookback (hits/misses): 2178/45

User-DN:
/O=GermanGrid/OU=ZAH/CN=Ralf Wahner

File-spaces:
  ID S URL                                FREE          TOTAL
  1 a gsiftp://alnitak.ari.uni-heidelberg.de/... 3990339803400 4000000000000
  2 a gsiftp://astrodata10.gac-grid.org/... 1899768040842 1900000000000
  3 a gsiftp://astrodata07.gac-grid.org/... 1899999843231 1900000000000

Default file-space: 1
```

The URLs at the bottom end of the output are abbreviated for better readability. They are, to their full extend `gsiftp://alnitak.ari.uni-heidelberg.de/opt/d-grid/adm/fs01`, `gsiftp://astrodata10.gac-grid.org/store/05/zah` and `gsiftp://astrodata07.gac-grid.org/store/02/ADM`.

The command line client uses the default file-space to place new files or to retrieve files that are already under ADM control, unless told otherwise or the default file-space is not available. Beyond unavailability of file-spaces, which can be caused e.g. by network failure or local administrative issues, there are reasons for overriding the default setting and manually selecting another file-space, e.g. duplicating crucial data for backup or shorter transfer distances across the internet. This is where the flags `-s` and `-b` come into play. If `-s` (`--file-space`) is present but `-b` is not, the client tries to access exactly the file-space given as the flag's argument. When the specified file-space is inaccessible, the client immediately gives up and displays an error message. However, if the `-b` (`--fallback`) flag is also present, the client will try one file-space after the other in order to access the desired file and it won't give up until the last file-space fails as well.

```

emacs@alnitak
\sourcecomment{\# Show root directory (verbose output)} \\  

\srcindent{00}\shellprompt{} \sourcecodehighlight{adm ls -l /}\ [6pt]  

\srcindent{00}d nGrid/OU=ZAH/CN=Ralf Wahner \srcindent{3}0 2008-01-08 10:25 adm-tutorial/\\  

\srcindent{00}d ZAH/CN=Thomas Bruesemeister \srcindent{3}0 2007-12-07 17:32 astrogrid/\\  

\srcindent{00}d ZAH/CN=Thomas Bruesemeister \srcindent{3}0 2007-12-07 17:31 home/\\  

\srcindent{00}d ZAH/CN=Thomas Bruesemeister \srcindent{3}0 2007-12-07 17:32 incoming/\\  

\srcindent{00}d ZAH/CN=Thomas Bruesemeister \srcindent{3}0 2008-03-05 09:51 lost+found/\\  

\srcindent{00}d nGrid/OU=ZAH/CN=Ralf Wahner \srcindent{3}0 2008-01-08 13:14 performance-scalability/\\  

\srcindent{00}s \srcindent{24}ADM \srcindent{3}0 2007-12-07 17:18 adm\  

\srcindent{00}7 entries  

\end{sourcecodeENV}  

***  

The leftmost column indicates the file type of the entries, where lowercase \sourcecode{d} is assigned to directories  

whereas lowercase \sourcecode{f} denotes a file. \filename{ADM} is a particular *** ausgezeichnet (besonders)  

directory internally used by \ADM{} for administrative purposes and therefore has type \sourcecode{s} in order to  

be distinguishable from normal directories. The second column displays the file owner, compiled from the attribute  

mnemonics found in the users proxy certificate and truncated for the sake of readability. The two-character keywords  

are defined in the Lightweight Directory Access Protocol (LDAP) specification and mean: common name (\sourcecode{CN}),  

organization (\sourcecode{O}), organizational unit (\sourcecode{OU}) and country (\sourcecode{C}). The third column  

shows the file size in bytes and intentionally vanishes for directories. Finally, the fourth column indicates date and  

time when the entries were created. The output of \sourcecode{adm help list} demonstrates how to access the built-in  

documentation for an \ADM{} command and summarizes the previous two examples:  

*** adm help list  

\begin{sourcecodeENV}  

\sourcecomment{\# Show build-in help for 'list'} \\  

\srcindent{00}\shellprompt{} \sourcecodehighlight{adm help list}\ [6pt]  

\srcindent{00}Lists files and directories in the virtual filesystem. \ [6pt]  

\srcindent{00}usage: list vfs-path \ [6pt]  

\srcindent{00}Valid Options: \\  

\srcindent{02}-l [--long] \srcindent{11}: use a long listing format\  

\srcindent{02}-u [--userdn-matches] \srcindent{1}: shows only entries matching your userdn \ [6pt]  

\srcindent{00}Example: adm ls -l /home  

\end{sourcecodeENV}  

***  

By default, \sourcecode{adm list vfs-path} displays all entries in the \filename{vfs-path} directory, where  

''vfs'' ist short hand for ''virtual filesystem'', regardless of their individual ownership.  

The optional \sourcecode{-u} (\sourcecode{-()}-user\dn-mat\ches}) switch allows to filter the  

output in order to show only those files and directories that belong to the grid user who invoked the command.  

%The \sourcecode{-b} switch is boolean, i.e. it is either present or absent and never has an argument.  

By the way, as the above output shows, each \sourcecode{adm help <subcommand>} contains an example how  

to use this \sourcecode{<subcommand>}.

```

Figure 2: In-situ editing a file by means of adm edit.

## 4 Retrieve, Replicate and Cleanup Files

This is currently the final section of the ADM tutorial and it describes how to get a local copy of a file that resides under AMD control, how to store a copy of a file on another file-space and how files and directories can be removed from ADM. A replica of a file is a one-to-one copy of that file on a different file-space; more precisely, two replicas of a file can never reside on the same file-space, but discussing that is beyond the scope of this text. Replicas are created for several reasons, e.g. to back up significant data or to reduce the network transfer load by locating a file as near to the target arithmetic resource as possible, to mention two frequently named requirements. Unless told otherwise, ADM implicitly selects an appropriate file-space, when `adm replicate` is called:

```

# How to create replicas of files
agrid064@alnitak:~$ adm replicate /tutorial/vfs_tour/my_jobdescription.rsl

Source: gsiftp://alnitak.ari.uni-heidelberg.de/opt/d-grid/adm/fs01/
Dest:   gsiftp://astrodata10.gac-grid.org/store/05/zah/
        aab3c89633c6af44407ecedeb98f4fb5

```

While `adm list` displays the files and directories on the specified level in the filesystem hierarchy, `adm resolve` takes a filename argument and displays a list of locations of all replicas of that file, so `adm resolve` is a kind of counterpart of `adm list`:

```
# How to view available replicas and their locations
agrid064@alnitak:~$ adm resolve /tutorial/vfs_tour/my_jobdescription.rsl

gsiftp://astrodata10.gac-grid.org/store/05/zah/aab3c89633c6af44407ecedeb98f4fb5
gsiftp://alnitak.ari.uni-heidelberg.de/opt/d-grid/adm/fs01/aab3c89633c6af44407e-
cedeb98f4fb5
```

After browsing the virtual filesystem in order to find a specific file, the retrieval starts operating by means of `adm get` from the default file-space:

```
# How to get a file out of the ADM
agrid064@alnitak:~$ adm get /adm-tutorial/vfs_tour/my_jobdescription.rsl
Source: gsiftp://alnitak.ari.uni-heidelberg.de/opt/d-grid/adm/fs01/
Dest:   file:///home/Aggrid/agrid064/
        aab3c89633c6af44407ecedeb98f4fb5 -> my_jobdescription.rsl
```

The commands `adm replicate` and `adm get` can use the above documented flags `-s` and `-b` in order to manually select the file-space where the replica should be placed or where the file should be retrieved from, respectively.

Finally, files are deleted from the virtual filesystem by means of `adm remove` (abbreviated `adm rm`) whereas directories are wiped out by means of `adm rmdir`. Non-empty directories cannot be removed and file and directory removal is based on ownership, i.e. any user can delete filesystem entries that he owns, only:

```
# How to remove a file from the ADM
agrid064@alnitak:~$ adm remove /vfs_tour/jsdl/my_jobdescription.rsl
agrid064@alnitak:~$ adm rmdir /vfs_tour/jsdl/
agrid064@alnitak:~$ adm rmdir /vfs_tour
```

**Outlook:** Careful readers might have noticed, that the commands concerning the user-defined file properties, namely `adm prop(del|get|list|set)` are not yet described in this tutorial. In order to accommodate the way of thinking and the requirements of common scientific AstroGrid-D users, the revised future tutorial is supposed to guide readers along a typical scenario, i.e. an  $n$ -body or  $\varphi$ -grape run, rather than "*my\_file.txt*" and "*my\_directory*". Apart from those two items, the authoring board welcomes suggestions what should be included in this text; see e-mail addresses in footnote 1 on the cover page. Please consider, that ADM has just been released on January 21<sup>st</sup>, 2008 and development and documentation need some time to accumulate the users' experience. Thanks for you interest in the AstroGrid-D Data Management.

## References

- [1] AstroGrid-D Data Management (ADM) build-in documentation, accessible by means of `adm help` (general information) or `adm help <subcommand>` (manual page for individual subcommand).
- [2] Collins-Sussman, Ben; Fitzpatrick, Brian W. and Pilato, C. Michael: *Version Control with Subversion*, for Subversion 1.4 (Compiled from r2866), see <http://svnbook.red-bean.com> or file *svn-book.pdf* located in directory *adm\_tutorial/misc*.

- [3] The official `globus-url-copy` website: *globus-url-copy — Multi-protocol data movement* at <http://www.globus.org/toolkit/docs/4.0/data/gridftp/rn01re01.html>.