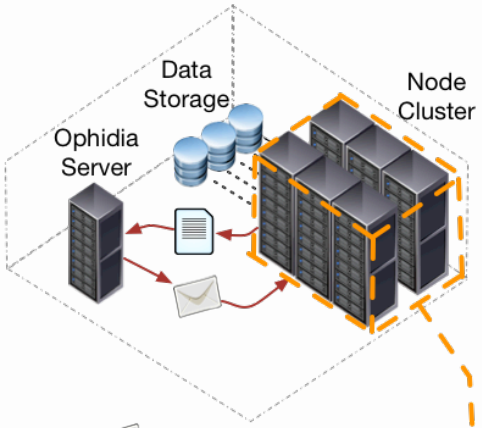
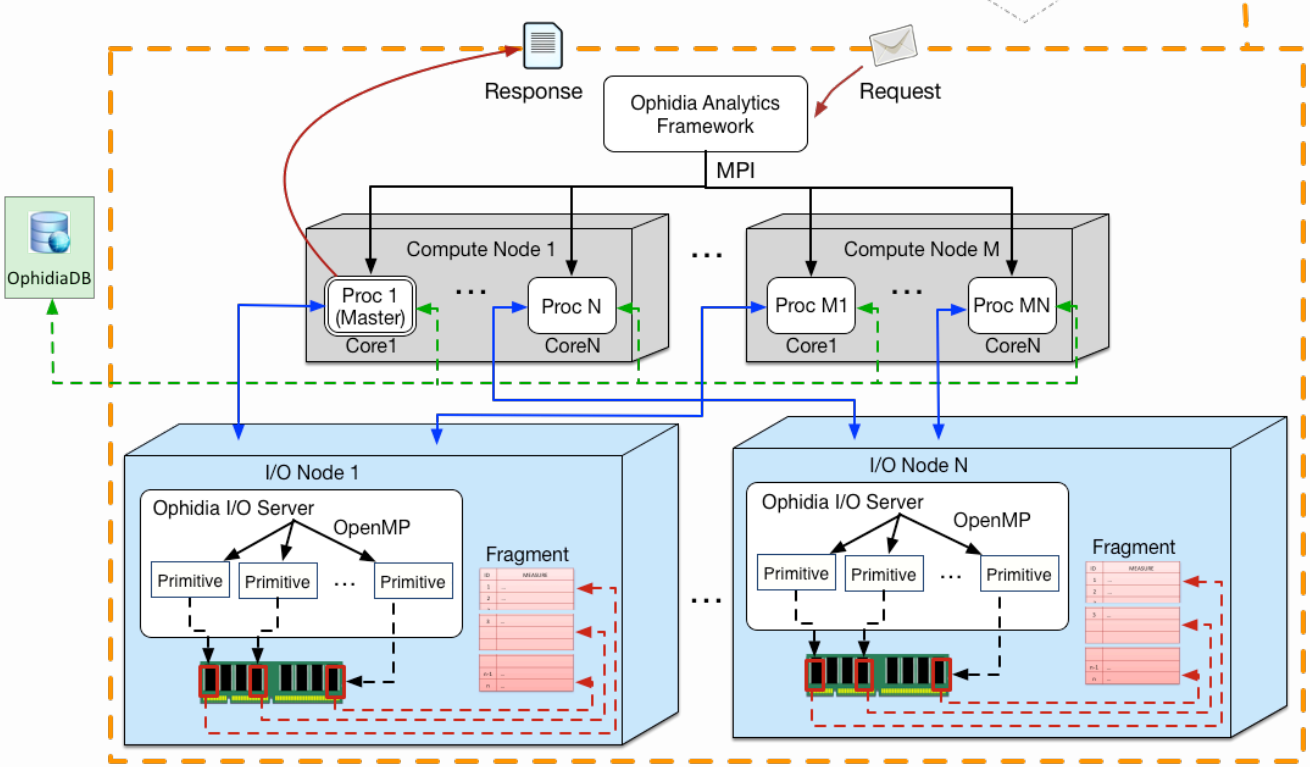
Oph_Term
The Ophidia shell

Part I : First Steps

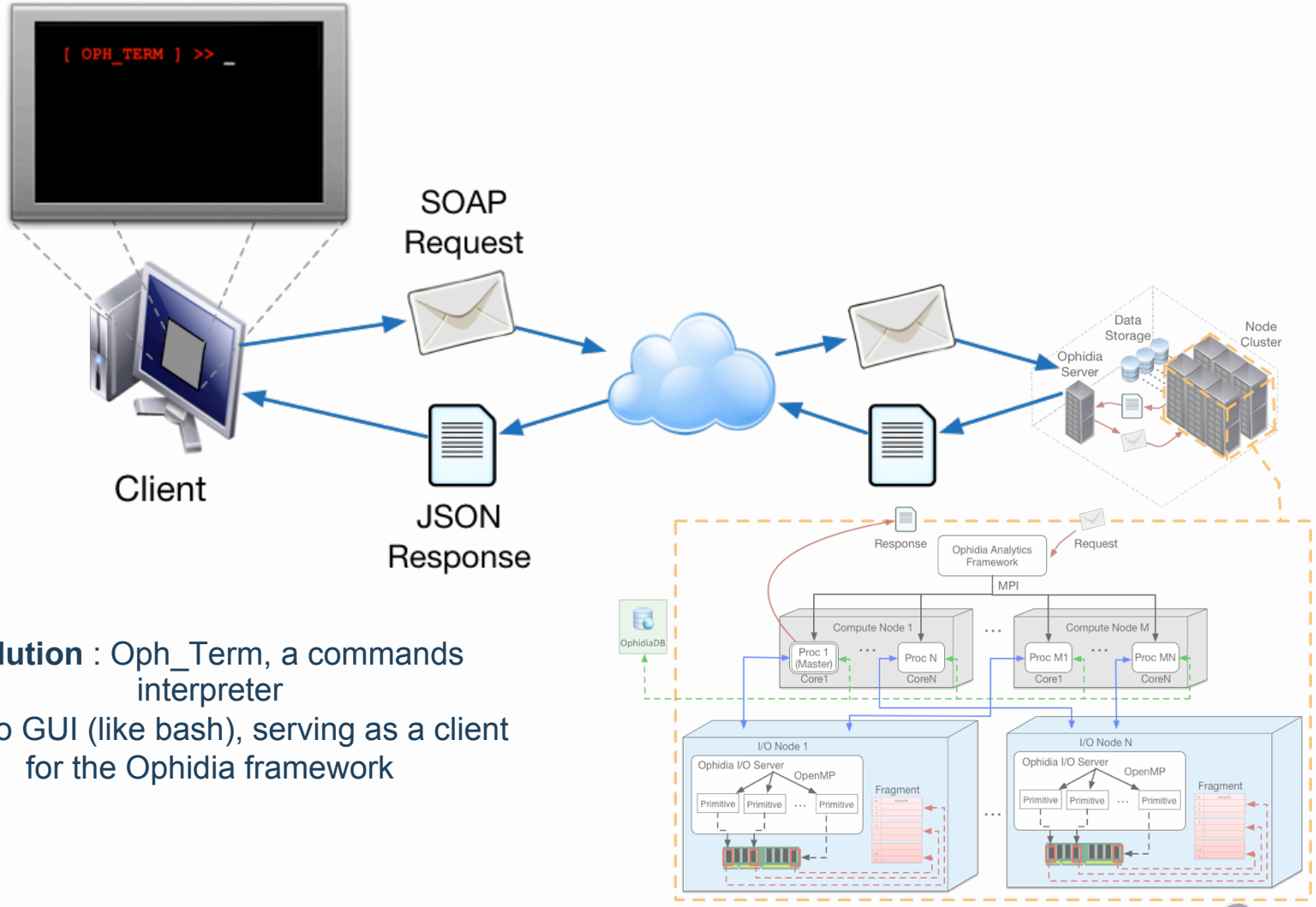
The Ophidia Architecture

Problem : find an effective way to interact with Ophidia without making users scared of its complexity





The Ophidia Architecture



Solution : Oph_Term, a commands interpreter
with no GUI (like bash), serving as a client
for the Ophidia framework

Oph_Term : the Ophidia shell

WHAT ?

- Commands interpreter with no GUI like bash
- A client for the Ophidia framework

WHY ?

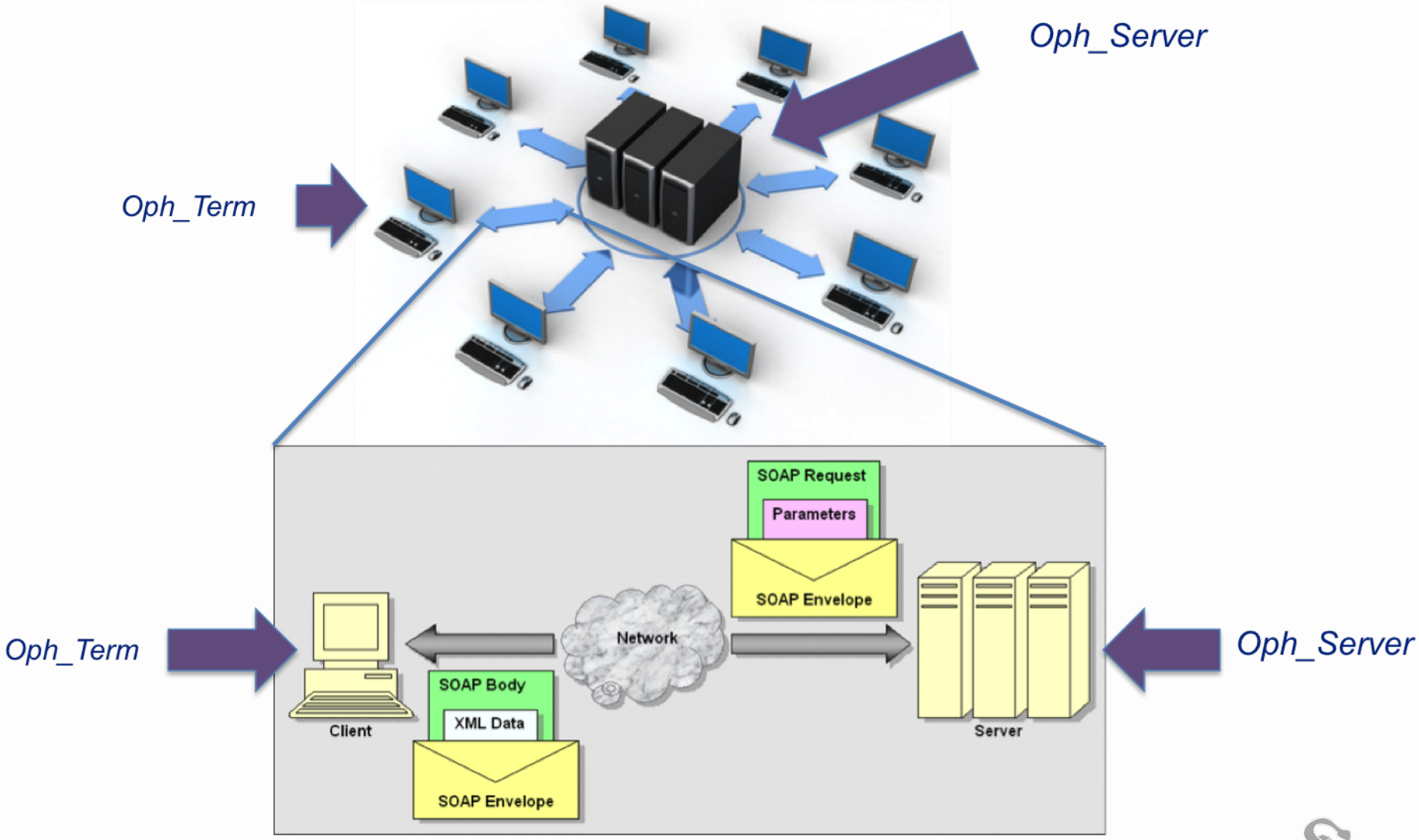
- It simplifies client-server interaction within the Ophidia framework
- It makes it easier to take advantage of all Ophidia operators

HOW ?

- Terminal with history management, auto-completion, specific environment variables and commands with integrated help…
- Easy installation as an only one executable using a small number of well-known and open-source libraries
- More than 15 KLOC
- Simple enough for a novice and at the same time powerful enough for an expert

Oph_Term : basic use case

Oph_Term - Oph_Server interaction



Execute Oph_Term

```
[...][~] -> oph_term -u user -p passwd -H host -P port
```

Start Oph_Term with one or more login options:

- -u: remote user;
- -p: remote user password;
- -H: server address;
- -P: server port number.

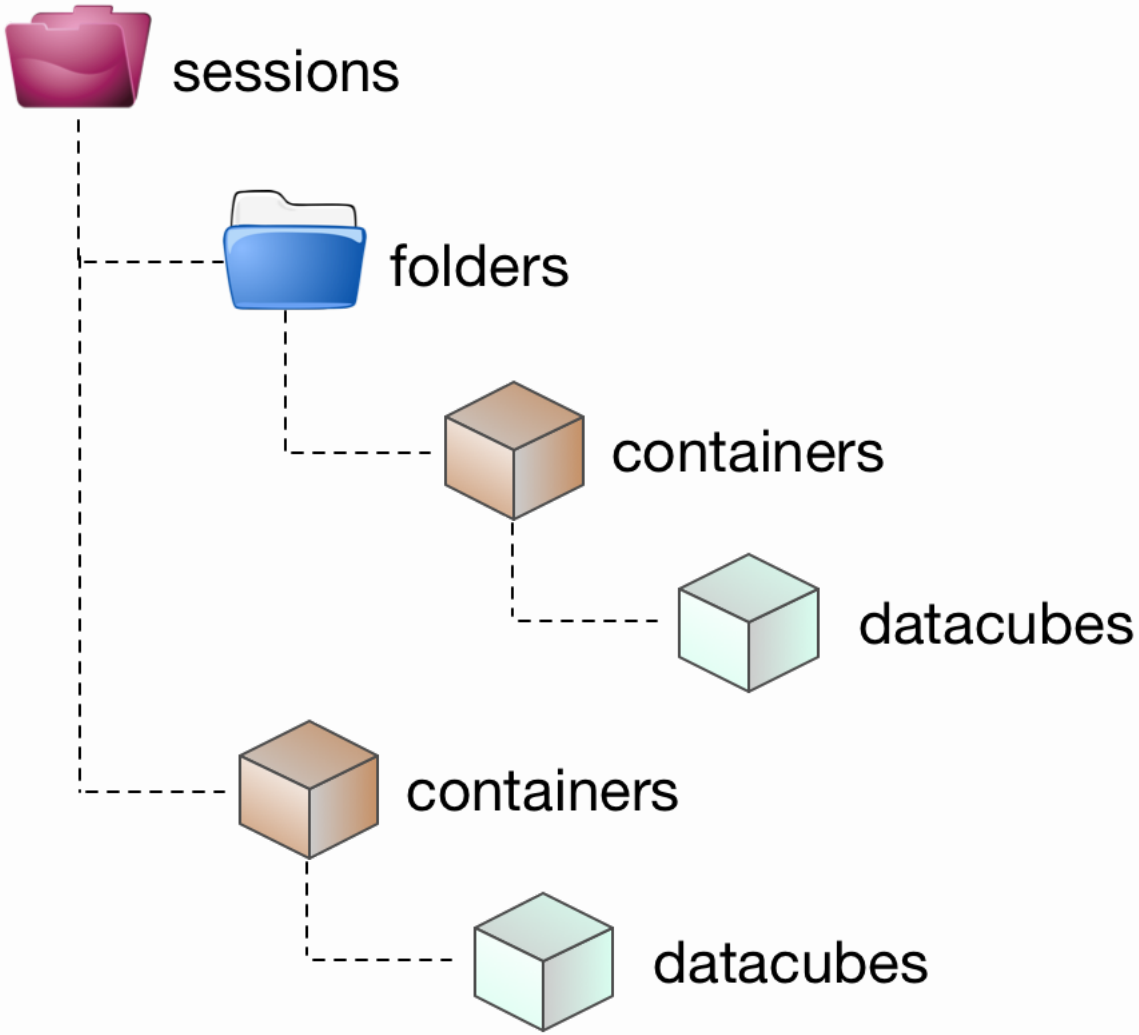
```
[...][~] -> oph_term
```

Start Oph_Term with no particular options.
Use the content of your shell variables OPH_USER, OPH_PASSWD, OPH_SERVER_HOST, OPH_SERVER_PORT, etc. if defined in your shell environment at runtime or at shell startup through files like .bashrc to set corresponding Oph_Term variables.

```
[...][~] -> oph_term -e "my command”
```

Execute a command without entering Oph_Term.
Oph_Term can always be started with one or more login options as described above.
Oph_Term will also search for not-inserted parameters in your shell environment as described above and will use environment variables as in interactive mode.

Ophidia Virtual Filesystem



Oph_Term environment

PRE-DEFINED VARIABLES
(user-defined variables possible)

OPH_TERM_PS1 = "color of the prompt”
OPH_USER = "username”
OPH_PASSWD = "password”
OPH_SERVER_HOST = "server address”
OPH_SERVER_PORT = "server port”
OPH_SESSION_ID = "current session”
OPH_EXEC_MODE = "execution mode”
OPH_NCORES = "number of cores”
OPH_CWD = "current working directory”
OPH_DATACUBE = "current datacube”
OPH_TERM_VIEWER = "output renderer”
OPH_TERM_IMGS = "save and/or auto-open images"

PRE-DEFINED COMMANDS
(user-defined aliases possible)

help = "get the description of a command or variable”
history = "manage the history of commands”
env = "list environment variables”
setenv = "set or change the value of a variable”
unsetenv = "unset an environment variable”
getenv = "print the value of a variable”
quit = "quit Oph_Term”
exit = "quit Oph_Term”
clear = "clear screen”
update = "update local XML files”
resume = "resume session”
view = "view jobs output”
alias = "list aliases”
setalias = "set or change the value of an alias”
unsetalias = "unset an alias”
getalias = "print the value of an alias"

Oph_Term: operator submission

[OPH_TERM] >> oph_operator param1=val1;param2=val2;...paramN=valN;

Ophidia Operator such as "oph_list"

Sequence of arguments passed to the operator

IF UNSURE USE THE MANUAL…

Get a complete description of each operator/primitive with mandatory and optional arguments with:

oph_man function=operator_name;
oph_man function=primitive_name;function_type=primitive;

In case of optional arguments, they can be omitted and then their default values are used

```
Function Name and Version
-------------------------

OPH_MERGE v1.0

Function Info
-------------

+===============+=================================================================
| INFO TYPE     | INFO VALUE
+===============+=================================================================
| Abstract      | [Type]
|               | Data Process.
|               |
|               | [Behaviour]
|               | It creates a new datacube grouping nmerge input fragments in a new o
|               | If the number of fragments of the input datacube is not a multiple o
|               | then there will be some output fragments grouping a number of fragme
|               | In addition, the number of tuples in each output fragment is general
|               | The operator allows also to change the datacube fragmentation struct
|               |
|               | [Parameters]
|               | - cube : name of the input datacube. The name must be in DOI format.
|               | - schedule : scheduling algorithm. The only possible value is 0,
|               |              for a static linear block distribution of resources.
|               | - nmerge : number of input fragments to merge in an output fragment.
|               |
```

```
Function Arguments
------------------

+================+=========+============+==========+=====+=====+====================+
| NAME           | TYPE    | MANDATORY  | DEFAULT  | MIN | MAX | VALUES             |
+================+=========+============+==========+=====+=====+====================+
| sessionid      | string  | no         | null     |     |     |                    |
|----------------|---------|------------|----------|-----|-----|--------------------|
| ncores         | int     | no         | 1        | 1   |     |                    |
|----------------|---------|------------|----------|-----|-----|--------------------|
| exec_mode      | string  | no         | batch    |     |     | batch|interactive  |
|----------------|---------|------------|----------|-----|-----|--------------------|
| cube           | string  | yes        |          |     |     |                    |
|----------------|---------|------------|----------|-----|-----|--------------------|
| schedule       | int     | no         | 0        |     |     | 0                  |
|----------------|---------|------------|----------|-----|-----|--------------------|
| nmerge         | int     | yes        |          | 2   |     |                    |
|----------------|---------|------------|----------|-----|-----|--------------------|
| objkey_filter  | string  | no         | all      |     |     | all|none|merge    |
+================+=========+============+==========+=====+=====+====================+
```

Oph_Term: operator submission

[OPH_TERM] >> | oph_operator | param1=val1;param2=val2;...paramN=valN; |

Ophidia Operator such as "oph_list"

Sequence of arguments passed to the operator

Special arguments:

- "exec_mode": it specifies if we want synchronous mode (”sync") or asynchronous mode (”async") which is the default;
- "ncores": it specifies the number of parallel processes requested for the execution of the operator (default is 1);
- "sessionid": it specifies the current session;
- "cwd": it specifies the current working directory;
- "cube": it specifies the input datacube.

They are special arguments that the user can explicitly write into the submission string or not, in which case Oph_Term will look up and use the content of the variables OPH_SESSION_ID, OPH_EXEC_MODE, OPH_NCORES, OPH_CWD or OPH_DATACUBE if existent.

Oph_Term : async mode

```
[…][~] -> oph_term -e "oph_operator exec_mode=async;"
```

```
[ OPH_TERM ] >> oph_operator exec_mode=async;
```

```
[Request]:
operator=oph_operator;exec_mode=async;cwd=/;

[JobID]:
http://oph_server/sessions/SESSIONCODE/experiment?cmdid#markerid
```

Non-blocking request, no response to render.
At job completion, the output response will be written in a file pointed by the JobID

Oph_Term : sync mode

```
[…][~] -> oph_term -e "oph_operator exec_mode=sync;"
```

```
[ OPH_TERM ] >> oph_operator exec_mode=sync;
```

```
[Request]:
operator=oph_operator;exec_mode=sync;cwd=/;

[JobID]:
http://oph_server/sessions/SESSIONCODE/experiment?cmdid#markerid

[Response]:
...
```

Blocking request, response rendered to standard output.
At job completion, the output response will be written in a file pointed by the JobID

Oph_Term Features : autocompletion

1) [OPH_TERM] >>[a-zA-Z]<*TAB*>

Perform autocompletion over Oph_Term specific commands (local commands), environment variables (pre-defined and user-defined) and aliases.
In case of "o" as first character autocompletion will return the prefix "oph_".

```
[ OPH_TERM ] >> s
```


```
[ OPH_TERM ] >> setenv
```

2) [OPH_TERM] >>$[a-zA-Z]<*TAB*>

Perform autocomplete over Oph_Term environment variables (pre-defined and user-defined).
The same kind of autocompletion occurs even if there is the following pattern:
[OPH_TERM] >>${[a-zA-Z]<TAB>
In this case a match is completed with the symbol "}".

```
[ OPH_TERM ] >> $OPH_U
```


```
[ OPH_TERM ] >> $OPH_USER
```

3) [OPH_TERM] >>[./]<*TAB*>

With a full stop or a slash perform autocompletion over local filesystem.

Oph_Term Features : autocompletion

4) [OPH_TERM] >>oph_*<TAB>*

With the "oph_" prefix perform autocompletion over Ophidia operators (remote commands).

```
[ OPH_TERM ] >> oph_ [ ] + [Tab] + [Tab] =
```

```
oph_aggregate          oph_duplicate          oph_list               oph_reduce
oph_aggregate2         oph_explorecube        oph_log_info           oph_reduce2
oph_apply              oph_exportnc           oph_loggingbk          oph_restorecontainer
oph_concatnc           oph_find               oph_man                oph_rollup
oph_createcontainer    oph_folder             oph_merge              oph_search
oph_cubeelements       oph_get_config         oph_metadata           oph_showgrid
oph_cubeio             oph_hierarchy          oph_movecontainer      oph_split
oph_cubeschema         oph_importnc           oph_operators_list     oph_subset
oph_cubesize           oph_inspectfrag        oph_permute            oph_subset2
oph_delete             oph_instances          oph_primitives_list    oph_system
oph_deletecontainer    oph_intercomparison    oph_publish2           oph_tasks
oph_drilldown          oph_intercube          oph_randcube2          oph_unpublish
[ OPH_TERM ] >> oph_ [ ]
```

Oph_Term Features : autocompletion

5) [OPH_TERM] >>oph_*<operator>* [a-z]*<TAB>* or [OPH_TERM] >>oph_*<operator>* arg1=val1;[a-z]*<TAB>*

In case of an operator specified at the beginning of the line perform autocompletion over its arguments.
In case of 1 possible match the line is automatically updated with the symbol "=".
In case of more than 1 possible matches they are all printed with useful info:

**: mandatory argument;
(x): default value for optional argument;
[x|y|...]: list of admitted values for an argument.

```
[ OPH_TERM ] >> oph_list
```


```
** cwd
container_filter (all)
cube (all)
db_filter (all)
dbms_filter (all)
exec_mode [async|sync (async)]
hidden [yes|no (no)]
host_filter (all)
level [0|1|2|3|4|5|6|7|8 (1)]
[ OPH_TERM ] >> oph_list
measure_filter (all)
ncores (1)
ntransform (all)
objkey_filter [all|none|list (all)]
path (-)
recursive [yes|no (no)]
sessionid (null)
src_filter (all)
```

Oph_Term Features : autocompletion

6) [OPH_TERM] >>oph_<operator> arg=[a-zA-Z0-9_]<TAB>

In this case perform autocompletion over the default value or all possible values of an operator argument.
In case of 1 possible match the line is automatically updated with the symbol ";".
In case of more than 1 possible matches they are all printed with default values between parenthesis.
The same kind of autocompletion occurs even if there is the following pattern:
[OPH_TERM] >>oph_<operator> arg1=val1;arg2=[a-zA-Z0-9_]<TAB>

7) ...<Shift><TAB>

Instead of the "classic" autocompletion, perform a "menu" autocompletion, cycling over possible matches directly inline. With this form of autocompletion additional characters are never appended.

```
[ OPH_TERM ] >> oph_list exec_mode=sync
```


```
[ OPH_TERM ] >> oph_list exec_mode=async
```

Oph_Term Features : automatic updates

If OPH_SERVER_HOST and OPH_SERVER_PORT are correctly set at Oph_Term startup, a request is forwarded to the Oph_Server to resume the last session the user was connected to and get the URL of the Ophidia operators XML repository. Oph_Term will automatically update local XML definitions to the latest version in order to provide enhanced autocompletion over Ophidia operators.

```
[…][~] -> oph_term
Resuming last session... Done.
Current session is now "http://localhost/sessions/151166199835442427221411578964324484/experiment".
Last working directory was "/".
Last produced datacube was "http://localhost/4/21".

Getting list of Ophidia operators XML files from "http://localhost/operators_xml/"... Done.
Downloading necessary files... Done.
Remote XML files: 54 - Downloaded XML files: 0 - Removed XML files: 0

Oph_Term - the Ophidia shell, version 1.5.1-20
Copyright (C) 2013-2014 CMCC - www.cmcc.it

Welcome to Oph_Term !
Use the power of the Ophidia framework right from your terminal.
If you are going to use Oph_Term for the first time and need something
to get you started, just try entering "help"

[15..4484] >>
```

A user can eventually check for updates during an interactive Oph_Term session through the command update. It is also possible to force Oph_Term to delete all local XML definitions for a particular server and to download all remote corruption-free versions with update -f. To switch to another session use the command resume. To view the output of a job without switching session use the command view.

Oph_Term : other features

- history navigation with ↑↓
- history recursive search with *<Ctrl>-<r>*
- *<Ctrl>-<a>*, *<Ctrl>-<e>* …
- history expansion with *<!>*
- startup automatic loading of environment variables
- ...

```
# .bashrc

export OPH_USER="oph_user"
[...]
```

```
[…][~] -> oph_term
Welcome to Oph_Term !

Use the power of the Ophidia framework right from your terminal.
If you are going to use Oph_Term for the first time and need something
to get you started, just try entering "help"

[ OPH_TERM ] >> env
OPH_TERM_PS1=
OPH_USER=oph_user
[...]
```

Oph_Term : config & install

- Set custom preferences (i.e. login parameters) in your ~/.bashrc
- *. ~/.bashrc* or *source ~/.bashrc*

```
# ~/.bashrc

export OPH_USER="oph_user"
export OPH_PASSWD="oph_passwd"
export OPH_SERVER_HOST="http://server.host.com"
export OPH_SERVER_PORT="12345"
export OPH_TERM_PS1="red"
export OPH_EXEC_MODE="sync"
[...]
```