



# Collaborative Climate Community Data and Processing Grid (C3-Grid)

## Environment-Modules im C3-Grid

---

Arbeitspaket:	AP7
Autor:	Volker Winkelmann
Version:	Juni 2007
Veröffentlichungsdatum:	Juli 2008
AP-Koordination:	
Partner:	
Ansprechpartner:	Volker Winkelmann
Email:	Winkelmann@Uni-Koeln.DE

---

GEFÖRDERT VOM



Bundesministerium  
für Bildung  
und Forschung

*Projekt im Rahmen des D-Grid  
Gefördert vom BMBF*



# Environment-Modules im C3-Grid

Anbieter von C3-Grid-Workflow-Modulen stehen vor dem Problem, ihre Module portabel und unabhängig von Site-spezifischen Gegebenheiten des Computerressourcenproviders gestalten zu müssen, obwohl Site-spezifische Tools (wie CDOs, NetCDF-Libraries etc.) eingesetzt werden oder andere lokale Informationen benötigt werden. Dortmund und Köln schlagen daher vor, dass Computerressourcenprovider im C3-Grid das Paket "Environment Modules" für die Workflow-Module-Anbieter als unabhängige Schnittstelle installieren, über die das benötigte Environment initialisiert werden kann. Dieses Dokument gibt dazu die notwendigen Hinweise für die Computerressourcenprovider und die Workflow-Module-Provider.

## Inhalt

1.	Admin-Guide (Hinweise für Ressourcenprovider).....	3
1.1.	Das Paket "Environment Modules" .....	3
1.1.1.	Installation von "Environment Modules" .....	3
1.1.2.	Konfiguration von "Environment Modules" .....	3
1.1.3.	Erstellung von Modulefiles zur Initialisierung des Environments .....	4
1.2.	Versionierung von Tools.....	7
1.3.	Standardisierung von Toolnamen und Shell-Variablen.....	8
1.4.	Veröffentlichung im Informationsdienst.....	8
2.	User's Guide (Hinweise für WF-Module-Provider) .....	10
2.1.	Das Paket "Environment Modules" .....	10
2.1.1.	Bereitstellung von Tools innerhalb von Standard-Shells .....	10
2.1.2.	Tools entfernen .....	10
2.1.3.	Übersicht über bereitgestellte Tools .....	10
2.1.4.	Übersicht über alle zur Verfügung stehenden Tools.....	10
2.1.5.	Dokumentation .....	11
2.2.	Initialisierung der C3-Grid-Umgebung .....	11
2.2.1.	Beispiel für den Header in einem Workflow-Module unter Verwendung der Bourne-Shell.....	11
2.2.2.	Beispiel für den Header in einem Workflow-Module unter Verwendung der C-Shell	12
2.2.3.	Beispiel für den Header in einem Workflow-Module unter Verwendung von Perl	12
2.3.	Standardisierung von Toolnamen und Shell-Variablen.....	13
A.	Anhang: Toolnamen und Shell-Variablen .....	14
A.1.	Vorinstallierte Pakete ohne Modulefiles.....	14
A.2.	Tools mit Modulefiles.....	14
A.3.	Shell-Variablen .....	15

# 1. Admin-Guide (Hinweise für Ressourcenprovider)

## 1.1. Das Paket "Environment Modules"

Das Paket "Environment Modules" ist ein Werkzeug zur einfachen Verwaltung einer Software-Umgebung (PATH, MANPATH, etc.). Durch Aufruf des "Kommandos" (Bourne-Shell-Function oder C-Shell-Alias)

```
module load <paketname>
```

sollen für den Anwender automatisch die notwendigen Variablen wie etwa PATH, MANPATH, LD\_LIBRARY\_PATH gesetzt oder zusätzliche Initialisierungen durchgeführt werden, die das gewünschte Paket benötigt. Durch die Anlage sogenannter Modulefiles durch den Systemadministrator wird das explizite Ausführen Shell-abhängiger Resourcefiles durch den Enduser überflüssig, sämtliche Initialisierungsinformation steckt im Modulefile eines Tools und wird beim obigen Aufruf umgesetzt.

### 1.1.1. Installation von "Environment Modules"

In den einschlägigen Archiven gibt es bereits vorbereitete Binär-Pakete:

Paket-Name	Betriebssystem
environment-modules	Redhat, Fedora
Modules	Suse

Die Original-Quellen befinden sich bei SourceForge, Source-RPMs können an verschiedensten Stellen, wie etwa RPMfind.net unter den Bezeichnungen *environment-modules* oder *Modules* gefunden werden.

Für die Installation der Environment-Modules sind u.a. eine installierte Tcl sowie TclX erforderlich.

**Die Environment Modules müssen auf jedem Cluster-Knoten installiert werden!**

### 1.1.2. Konfiguration von "Environment Modules"

Nach erfolgreicher Installation befinden sich der Software- und Konfigurationsbaum typischerweise im Verzeichnis `/usr/share/Modules`, sowie die Initialisierungsscripte (ev. als symbolische Links) unter `/etc/profile.d` :

```
modules.csh  
modules.sh
```

für Environment-Modules.

Die Man-Pages erhält man mit den Aufrufen

```
man module
man modulefile
```

bzw. bei Sourceforge HTML-Versionen von [module](#) und [modulefile](#).

Die o.g. Initialisierungsscripte, die zumindest beim Login automatisch ausgeführt werden, stellen zum einen die Funktion bzw. den Alias `module` bereit, zum anderen werden die globalen Variablen `MODULESHOME` und `MODULEPATH` gesetzt.

`MODULEPATH` bestimmt den Suchpfad, unter dem die Initialisierungen (sogenannte Modulefiles) der einzelnen Software-Pakete gesucht werden sollen. `MODULEPATH` muss nicht direkt in den o.g. beiden Initialisierungsscripten verändert werden, es existiert eine Datei `$MODULESHOME/init/.modulepath` mit Pfadangaben für die Modulefiles, die mit einem ASCII-Editor ergänzt/verändert werden kann.

Für C3-Grid empfiehlt es sich, ein eigenes Verzeichnis zu verwenden, dass auf allen Cluster-Knoten zur Verfügung steht, um flexibel beim Angebot von speziellen Softwareversionen zu bleiben, die sich nicht unbedingt mit den "offiziellen" Softwareversionen der Computeressourceprovidern decken müssen.

Die Datei `$MODULESHOME/init/.modulepath` könnte dann wie folgt aussehen:

```
/usr/share/Modules/modulefiles # general module files
/opt/MySite/etc/modulefiles    # std. module files of my site
/opt/c3grid/etc/modulefiles    # special module files for C3
```

**Jede Änderung der Konfiguration muss auf jeden Cluster-Knoten übertragen werden.**

Wenn aus betrieblichen Gründen das Modulefiles-Verzeichnis nicht geshared ist, muss selbst für den Abgleich des Verzeichnisses auf allen Cluster-Knoten gesorgt werden.

### 1.1.3. Erstellung von Modulefiles zur Initialisierung des Environments

Die Art und Weise, wie Software-Pakete per *Environment-Modules* (Aufruf `module`) initialisiert werden, wird in sogenannten *Modulefiles* festgelegt, die in einem der speziellen Verzeichnisse liegen, und die in der Shell-Variablen `MODULEPATH` spezifiziert sind. Jedes Software-Paket erhält einen eigenen Modulefile, der Filename ist gleichzeitig der Aufrufparameter bei der Option `load`.

**Alle Modulefiles müssen auf allen Cluster-Knoten installiert werden.**

#### Beispielaufruf zur Initialisierung der Softwaresammlung CDOs

Der Modulefile zu den CDOs liege etwa unter `/opt/c3grid/modulefiles/cdo`:

```
module load cdo
```

Damit `module` das Environment für die CDOs setzen kann, muss im einfachsten Fall im o.g. File die Initialisierung der Variablen `PATH` und `LD_LIBRARY_PATH` oder `LIBPATH` in einer TCL-artigen Sprache beschrieben werden.

## Beispiel für einen Modulefile /opt/c3grid/modulefiles/cdo

```
##Module1.0#####  
## modulefile for CDOs - very simple version  
##  
proc ModulesHelp { } {  
    global version  
    puts stderr "\tThis module sets pathes for standards C3-Grid apps"  
}  
  
module-whatis    "initialize C3-Grid apps"  
  
prepend-path PATH                /opt/c3grid/software/cdo/bin  
prepend-path LD_LIBRARY_PATH    /opt/c3grid/software/cdo/lib  
prepend-path LIBPATH            /opt/c3grid/software/cdo/lib
```

Modulefiles können auf anderen Modulfiles aufbauen und diese direkt laden. Man kann so Abhängigkeiten von Standard-Tools, Environment-Variablen oder Pfaden umgehen, indem in einem Modulefile die erforderlichen Modulefiles explizit geladen werden.

## Beispiel für den Modulefile grads von GrADS - Implizites Laden anderer Tools

Die Software GrADS sei in einem allgemeinen C3-Grid-Software-Baum untergebracht, die Variable GADDIR soll noch gesetzt werden:

```
##Module1.0#####  
## modulefile for GrADS - loads environment "c3grid"  
##  
proc ModulesHelp { } {  
    global version  
    puts stderr "\tThis module sets pathes for GrADS in C3-Grid"  
}  
  
module-whatis    "initialize GrADS for C3-Grid"  
  
# initialize C3-Grid software environment, especially GrADS  
module load c3grid  
  
# GrADS version number to find correct addons  
set gradsversion "1.9b4"  
set gradslibdir grads-$gradsversion  
  
# root for addons  
set c3home $env(C3_WN_TOOLS)  
set c3lib  $c3home/lib  
  
# global variable for addons  
setenv GADDIR $c3lib/grads-$gradsversion
```

Es wird auf das Environment *c3grid* verwiesen. Der zugehörige Modulefile *c3grid* könnte wie folgt aussehen:

```

##Module1.0#####
## modulefile for C3-Grid apps
#
proc ModulesHelp { } {
    global version
    puts stderr "\tThis module sets pathes for standard C3-Grid apps"
}

module-whatis "initialize C3-Grid apps"

# Initialize C3-Grid environment variables, especially C3_WN_TOOLS
module load c3env

# get root of installed software from environment variable C3_WN_TOOLS
set c3home $env(C3_WN_TOOLS)

prepend-path PATH $c3home/bin
prepend-path LD_LIBRARY_PATH $c3home/lib
prepend-path LIBPATH $c3home/lib

```

Es wird auf das Environment `c3env` verwiesen, in dem C3-Grid-Umgebungsvariablen festgelegt werden, die für (in Workflow-Modulen) benutzte Tools von Bedeutung sein könnten, wie etwa der Pfad zum Scratch-Bereich. Der zugehörige Modulefile `c3env` könnte wie folgt aussehen:

```

##Module1.0#####
## modulefile for C3-Grid environment variables
##
proc ModulesHelp { } {
    global version
    puts stderr "\tThis module initializes the C3-Grid environment"
}

module-whatis "initialize C3-Grid envoronment vars"

# ##### global vars #####
# scratch directory for large temporary files on worker node
setenv C3_WN_SCRATCH /scratch

# root of C3-Storage-Filesystem
setenv C3_WN_ROOT /storage/c3grid

# root of workspace for I/O on worker node
setenv C3_WN_WORKSPACE /storage/c3grid/dms_workspace

# root directory of C3-Tools like GraDS, CDOs etc. and profiles
setenv C3_WN_TOOLS /projects/c3tools

# root directory of C3-Workflows
setenv C3_WN_WORKFLOWS /projects/c3tools/workflows

```

## Der Aufruf

```
module load grads
```

initialisiert die Variablen C3\_WN\_SCRATCH, C3\_WN\_ROOT, C3\_WN\_WORKSPACE, C3\_WN\_TOOLS, C3\_WN\_WORKFLOWS, erweitert die Variablen PATH, LD\_LIBRARY\_PATH und LIBPATH um die benötigten Einträge, und setzt schließlich die Variable GADDIR.

## 1.2. Versionierung von Tools

Für einzelne Workflow-Module können spezielle Versionen von Tools erforderlich sein, die ihrerseits wieder spezielle Versionen anderer Tools benötigen, so dass im C3-Grid eine Versionierung vorgesehen werden sollte. Die prinzipielle Vorgehensweise ist in einem [Aufsatz](#) unter

[http://modules.sourceforge.net/docs/MC2\\_whitney\\_paper.pdf](http://modules.sourceforge.net/docs/MC2_whitney_paper.pdf)

Ch. 3.3, beschrieben.

Dabei werden für Tools nicht einfach Modulefiles mit dem Toolnamen angelegt, sondern stattdessen ein **Verzeichnis** mit dem Toolnamen. In diesem Verzeichnis werden dann mehrere Modulefiles für verschiedene Versionen des Tools angelegt; diese Modulefiles erhalten als Dateinamen die nackten Versionsnummern.

### Beispiel zu den CDOs

Im Modulefile-Verzeichnis `/opt/c3grid/etc/modulefiles` wird das Verzeichnis `cdo` für die CDOs angelegt, und darunter für jede installierte Version, etwa 1.0.2, 1.0.6, 1.0.7, die zugehörigen Modulefiles:

```
/opt/c3grid/etc/modulefiles/  drwxr-xr-x  c3grid  c3grid
|
+--cdo/                      drwxr-xr-x  c3grid  c3grid
|  |
|  +--1.0.2                  -rw-r--r--  c3grid  c3grid
|  +--1.0.6                  -rw-r--r--  c3grid  c3grid
|  +--1.0.7                  -rw-r--r--  c3grid  c3grid
|
+--grads/                    drwxr-xr-x  c3grid  c3grid
|  |
|  +--1.9b4                  -rw-r--r--  c3grid  c3grid
|  .
|  .
|  .
|  .
```

Das Laden der CDOs geschieht wie bisher mit dem Aufruf

```
module load cdo
```

wobei standardmäßig die in der alphabetischen Reihenfolge letzte Version, hier also Version 1.0.7, zur Initialisierung verwendet wird. Die Überprüfung mit

```
module list
```

zeigt, dass Version 1.0.7 verwendet wurde/wird:

```
Currently Loaded Modulefiles:
```

```
1) cdo/1.0.7
```

Um stattdessen ältere/andere Versionen zu initialisieren, kann die gewünschte Versionsnummer an den Toolnamen mit einen "/" angehängt werden. Beispiel für CDO Version 1.0.6:

```
module load cdo/1.0.6
```

Die Überprüfung mit

```
module list
```

zeigt, dass Version 1.0.6 verwendet wurde/wird:

```
Currently Loaded Modulefiles:
```

```
1) cdo/1.0.6
```

Weitere Hinweise, etwa zur gezielten Auswahl einer Version als Default, kann man dem o.g. [Aufsatz](#) entnehmen.

### 1.3. Standardisierung von Toolnamen und Shell-Variablen

Um für Entwickler von Workflow-Modulen die unabhängige Bereitstellung von Tools zu gewährleisten, müssen die Bezeichnungen für die eingesetzten Tools C3-Grid-übergreifend vereinbart werden, ebenfalls die Art und Weise der Versionsnummern, sowie die zur Verfügung stehenden Environment-Variablen.

Nun scheint es übertrieben zu sein, für Produkte wie Perl oder TCL, die standardmäßig installiert sind und über den Standard-PATH gefunden werden, eigene Modulefiles anzulegen. (Auf der anderen Seite ist es durchaus denkbar, dass wegen namensgleichen Einträgen in Bibliotheken Einfluß auf die Reihenfolge beim Linken von Bibliotheken genommen werden soll.)

Es ist daher notwendig, Community-übergreifend eine Konvention von vorinstallierten Paketen ohne Modulefiles, Tools mit Modulefiles und vorbesetzten Umgebungsvariablen zutreffen.

### 1.4. Veröffentlichung im Informationsdienst

Andere Grid-Dienste, wie Scheduler, Datenmanagement oder Portal, müssen über die eingerichtete Ausführungsumgebung jeder Site informiert werden. Für diesen Zweck pflegt jeder Anbieter ein lokales XML File, das die angebotenen Ressourcen beschreibt. Darin werden die installierten Modulnamen und -versionsnummern als Liste aufgenommen.

Ein Beispiel:

```
<C3GridResource xmlns="..." xmlns:xsi="...">
  <site id="WDCfoobar">
    <workspace>
      <baseUrl>
        gsiftp://c3host.dummy.host.de/Pfad/zum/Workspace/Basisverzeichnis
      </baseUrl>
    <execution>
      <wsGramUrl>
        https://gridftp.dummy.host.de:8443/Pfad/zum/WS-GRAM/Dienst
```

```
</wsGramUrl>
<modules>
  <module><name>gcc</name><version>4.2</version></module>
  <module><name>cdo</name><version>1.0.7</version></module>
</modules>
</execution>
</workspace>
</site>
</C3GridResource>
```

Die Information des Files muss im zentralen MDS-Dienst verfügbar gemacht werden. Dazu ist ein entsprechender Patch in Globus Toolkit einzuspielen. Das [Format der Ressourcenbeschreibung](#) ist als XML-Schema spezifiziert.

## 2. User's Guide (Hinweise für WF-Module-Provider)

### 2.1. Das Paket "Environment Modules"

Das Paket "Environment Modules" ist ein Werkzeug zur einfachen Verwaltung einer Software-Umgebung (`PATH`, `MANPATH`, etc.). Im C3-Grid soll "Environment Modules" den Entwicklern von Workflow-Modulen dienen, ihre Module unabhängig von lokalen Eigenheiten benötigter Tools zu gestalten. Der Entwickler muss nicht mehr wissen, wo ein benötigtes Tool installiert ist, welche zusätzlichen Shell-Variablen notwendig sind, oder welche Libraries benötigt werden. Nur noch der standardisierte Toolname ist erforderlich.

#### 2.1.1. Bereitstellung von Tools innerhalb von Standard-Shells

Durch Aufruf des "Kommandos" (Bourne-Shell-Function oder C-Shell-Alias)

```
module load <toolname1> <toolname2> ... <toolnameN>
```

werden für den Anwender dann automatisch die notwendigen (toolabhängigen) Environment-Variablen wie etwa `PATH`, `MANPATH`, `LD_LIBRARY_PATH` ergänzt und/oder zusätzliche Initialisierungen durchgeführt, die die gewünschten Tools benötigen.

#### 2.1.2. Tools entfernen

Die Änderungen des Environments, die durch den Aufruf von `module` bewirkt werden, können wieder rückgängig gemacht werden:

```
module unload <toolname1> <toolname2> ... <toolnameM>
```

#### 2.1.3. Übersicht über bereitgestellte Tools

Eine Übersicht über die mit `module` bereitgestellten Tool-Environments erhält man über

```
module list
```

#### 2.1.4. Übersicht über alle zur Verfügung stehenden Tools

Eine Übersicht über die Tools, die mit `module` bereitgestellt werden können, erhält man über

```
module avail
```

## 2.1.5. Dokumentation

Eine Übersicht über die sonstigen Möglichkeiten von `module` erhält man über

```
man module
```

bzw. über eine HTML-Seite bei [SourceForge](http://modules.sourceforge.net/man/module1.html) :

<http://modules.sourceforge.net/man/module1.html>.

## 2.2. Initialisierung der C3-Grid-Umgebung

Um die "Environment Modules" zur Initialisierung der C3-Grid-Umgebung verwenden zu können, sind einige Regeln zu beachten.

- Bereitstellung des `module`-Aufrufs - Initialisierung des Paketes "Environment Modules"

Normalerweise wird "Environment Modules" bereits beim Login initialisiert, der `module`-Aufruf funktioniert, Tools lassen sich initialisieren und starten. (Das liegt daran, dass im Verzeichnis `/etc/profile.d/` Initialisierungsscripte für "Environment Modules" liegen, die beim Login eingelesen werden.)

Normale Shellscripte hingegen werden nicht automatisch als Login-Shell gestartet; es kann daher auch nicht automatisch davon ausgegangen werden, dass etwa der `module`-Aufruf in einem Shellscript sofort zur Verfügung steht. Man könnte etwa durch Optionen beim Aufruf des Shellscriptes dafür sorgen, dass das betreffende Script in einer Login-Shell abläuft. Das ist allerdings möglicherweise nicht gewünscht, weil alle anderen Profiles aus `/etc/profile.d/` ebenfalls geladen werden, und/oder Seiteneffekte durch die Verwendung einer Login-Shell auftreten könnten.

Besser scheint die Methode, den notwendigen Profile zur Initialisierung der "Environment Modules" explizit einzulesen (s.Beispiel unten).

- Immer zuerst "Tool" `c3grid` initialisieren

Die Ressourcen-Provider stellen für allgemeine Zwecke, wie etwa die Bereitstellung von Shell-Variablen oder Tools, für die keine expliziten `module`-Aufrufe vorgesehen sind, das "Modul" `c3grid` zu Verfügung. `c3grid` sollte immer vor allen anderen Tools initialisiert werden, da sonst nicht für die ordnungsgemäße Bereitstellung der C3-Grid-Umgebung garantiert werden kann.

### 2.2.1. Beispiel für den Header in einem Workflow-Module unter Verwendung der Bourne-Shell

Verwendet man als Workflow-Module eine der Bourne-Shell-Arten (`/bin/sh`, `/bin/bash`, `/bin/ksh`), könnte der Header eines Workflow-Modules folgendermaßen aussehen:

```

#!/bin/sh
#
# just to be sure
export PATH
if [ -r "/etc/profile.d/modules.sh" ]; then
    . /etc/profile.d/modules.sh
else
    echo $0: /etc/profile.d/modules.sh not found
    exit 1
fi
#
module load c3grid
module load netcdf cdo/1.0.7 grads
#
.
.
.

```

### 2.2.2. Beispiel für den Header in einem Workflow-Module unter Verwendung der C-Shell

Verwendet man als Workflow-Module eine der C-Shell-Arten (`/bin/csh`, `/bin/tcsh`), könnte der Header eines Workflow-Modules folgendermaßen aussehen:

```

#!/bin/csh
#
if ( -r "/etc/profile.d/modules.csh" ) then
    source /etc/profile.d/modules.csh
else
    echo $0: /etc/profile.d/modules.sh not found
    exit 1
endif
#
module load c3grid
module load netcdf cdo/1.0.7 grads
#
.
.
.

```

### 2.2.3. Beispiel für den Header in einem Workflow-Module unter Verwendung von Perl

Verwendet man für ein Workflow-Module statt einer der Standard-Shells ein Perl-Script, so ist es nicht notwendig, das Perl-Script in einem Shell-Script zu verpacken. Man muss nur Perl-seitig zunächst die Initialisierung des Environment-Modules-Paketes durchführen, anschließend können mit Perl-Aufrufen sinngemäß die gewünschten Tools bereitgestellt, entfernt bzw. verwaltet werden. Für Perl könnte der Header eines Workflow-Modules folgendermaßen aussehen:

```

#!/usr/bin/perl
#
$initenvmodule="$ENV{'MODULESHOME'}/init/perl";
#
if (-r "$initenvmodule") {
    require "$initenvmodule";
}
else {
    die "could not init module environment with $initenvmodule";
}
#
module("load c3grid");
module("load netcdf cdo/1.0.7 grads");
#
.
.
.

```

Die zu Beginn des Dokumentes für die Standard-Shells beschriebene Aufrufe von `module` werden innerhalb eines Perl-Programms als Perl-Funktionsaufrufe abgesetzt, wobei die Argumente von `module` als 1 Zeichenkette übergeben werden.

### 2.3. Standardisierung von Toolnamen und Shell-Variablen

Um für Entwickler von Workflow-Modulen die unabhängige Bereitstellung von Tools zu gewährleisten, müssen die Bezeichnungen für die eingesetzten Tools C3-Grid-übergreifend vereinbart werden, ebenfalls die Art und Weise der Versionsnummern, sowie die zur Verfügung stehenden Environment-Variablen.

Andererseits gibt es Produkte wie Perl oder TCL, die standardmäßig installiert sind und über den Standard-PATH gefunden werden. Derartige Produkte werden in der Regel **nicht** wie oben beschrieben über den Aufruf von `module` initialisiert sondern sind **sofort** nutzbar.

Im Anhang soll daher eine Übersicht von vorinstallierten Paketen **ohne** `module`-Aufruf, Tools, von solchen **mit** `module`-Aufruf sowie eine Aufstellung der von Ressourcen-Providern vorbesetzten Umgebungsvariablen gegeben werden.

## A. Anhang: Toolnamen und Shell-Variablen

Um für Entwickler von Workflow-Modulen die unabhängige Bereitstellung von Tools zu gewährleisten, müssen die Bezeichnungen für die eingesetzten Tools C3-Grid-übergreifend vereinbart werden, ebenfalls die Art und Weise der Versionsnummern, sowie die zur Verfügung stehenden Environment-Variablen. Es soll dabei berücksichtigt werden, dass viele der benötigten Tools bereits von den Betriebssystem-Herstellern als installierbares Paket mitgeliefert werden und direkt nutzbar sind.

Die folgenden Tabellen geben Auskunft über die im C3-Grid zur Verfügung stehenden Bezeichner für Shell-Variablen und Tools im Environment.

### A.1. Vorinstallierte Pakete ohne Modulefiles

In der folgenden Tabelle werden Standard-Pakete aufgeführt, die von Workflow-Module-Entwicklern explizit benötigt werden, und die typischerweise von allen Betriebssystem-Herstellern mitgeliefert werden oder zumindest vom Systemadministrator leicht nachinstalliert werden können. Bei den genannten Paketen muss normalerweise das Environment nicht gesondert angepasst/verändert werden, die Pakete sind sofort nutzbar, so dass diese Pakete keiner expliziten Bereitstellung per `module`-Aufruf bedürfen.

Tool/Paket	Module-Name	Versioning	Quellen/weitere Infos
GnuPlot	-	-	gnuplot - an interactive plotting program
GhostScript	-	-	PostScript and PDF language interpreter
KSH	-	-	Korn Shell
Perl	-	-	perl - Practical Extraction and Report Language
Python	-	-	Python interpreter
TCL	-	-	TCL
...	-	-	

### A.2. Tools mit Modulefiles

In der folgenden Tabelle werden Tools aufgeführt, die von Workflow-Module-Entwicklern benötigt werden, die aber typischerweise nicht von allen Betriebssystem-Herstellern mitgeliefert werden, und/oder die in eigenen Dateibäumen verwaltet werden. Für diese Tools ist daher die Verwaltung über das Paket "Environment Modules" notwendig. In der folgenden Tabelle sind die Tools und die Namenskonvention bei Benutzung des `module`-Aufrufs aufgeführt.

Tool/Paket	Module-Name	Versionierung	Quellen/weitere Infos
C3-Grid	c3grid	-	allg. Initialisierung der C3-Grid-Umgebung ( <b>required</b> )
CDO	cdo	x.y.z	<a href="http://www.mpimet.mpg.de/~cdo/">http://www.mpimet.mpg.de/~cdo/</a>
GMT	gmt	x.y.z	<a href="http://gmt.soest.hawaii.edu/">http://gmt.soest.hawaii.edu/</a>
GrADS	grads	x.y.z	<a href="http://www.iges.org/grads/grads.html">http://www.iges.org/grads/grads.html</a>
Java RE	jre	x.y.z	Java Runtime Environments JRE mit allen erforderlichen Variablen
NetCDF	netcdf	x.y.z	<a href="http://www.unidata.ucar.edu/software/netcdf/">http://www.unidata.ucar.edu/software/netcdf/</a>
Pingo	pingo	x.y	<a href="http://www.mad.zmaw.de/Pingo/pingohome.html">http://www.mad.zmaw.de/Pingo/pingohome.html</a>
...			

### Beispiel: Initialisierung von CDO Version 1.0.7, NetCDF und GrADS

```
module load c3grid cdo/1.0.7 netcdf grads
```

*Anmerkung: c3grid sollte immer initialisiert werden, da es den Ressourcenprovidern die Möglichkeit bietet, spezielle Umgebungen für C3-Grid-Nutzer bereitzustellen, wie etwa Tools, für die normalerweise kein module-Aufruf vorgesehen ist. Ein Ressourcen-Provider kann das aber auch implizit in den "echten" Modulefiles verstecken.*

### A.3. Shell-Variablen

Die folgenden Shell-Variablen beinhalten Pfad-Informationen, die vom jeweiligen Ressourcenprovider bereitgestellt werden, um die Anfertigung von C3-Grid-Workflow-Modulen unabhängig vom Ressourcenprovider gestalten zu können.

Shell-Variable	Bedeutung	weitere Infos
C3GRID_SCRATCH	Scratch-Bereich auf Workernode	Plattenbereich, in dem ein Workflow-Modul temporäre Dateien anlegen kann
C3GRID_ROOT	Root des Workspace-Bereiches auf Workernode	vgl. <a href="#">Richtlinien für die Konfiguration des Workspaces</a>
...		