CLUSTER HPC LOVELACE
Guía de utilización del entorno via *modules* v.1.0 – 03/03/2016

ICMAT INSTITUTO DE CIENCIAS MATEMÁTICAS

GUÍA DE GESTIÓN DEL ENTORNO DE USUARIO MEDIANTE *MODULES*

Índice:

La gestión del entorno de usuario

Debido a la gran cantidad de software y tecnologías instaladas en el CLUSTER, que intenta adaptarse a diversos usos, a menudo surgen diversos problemas con respecto a las variables de entorno que requiere cada combinación de producto/versión/librería.

Por ello, se encuentra instalado el software `environment-modules` como herramienta para facilitar la gestión.

Este software permite la definición del entorno de usuario (variables de entorno) de manera controlada mediante scripts, disminuyendo la complejidad de las tareas.

Utilización de los comandos para la gestión del entorno

El comando básico de gestión de entorno es `module`. Mediante parámetros, pueden realizarse las siguientes acciones:

Ver la lista de módulos disponibles

```
$ module avail

------------------------ /usr/share/Modules/modulefiles ------------------------
dot         module-git  module-info modules     null        use.own

---------------------------- /LUSTRE/apps/Modules/ -----------------------------
Anaconda                libs/nag                matlab/2016b
compilers/gcc/4.4.7     Mathematica             mpi/mpich-x86_64
compilers/gcc/6.3.0     matlab/2010b            mpi/openmpi
compilers/intel/15.0.3  matlab/2015a            openfoam-4.1
gnuplot                 matlab/2016a            openfoam-v1612+
```

Ver la lista de módulos disponibles con descripción

```
$ module whatis
dot                 : adds `.' to your PATH environment variable
module-git          : get this version of the module sources from SourceForge.n
et
module-info         : returns all various module-info values
modules             : loads the modules environment
null                : does absolutely nothing
use.own             : adds your own modulefiles directory to MODULEPATH
Anaconda            : loads the environment for Anaconda (Python)
```

```
Mathematica          : loads the environment for Mathematica
compilers/gcc/4.4.7  : sets the environment for using gcc-4.4.7 compilers (C, Fo
rtran)
compilers/gcc/6.3.0  : sets the environment for using gcc-6.3.0 compilers (C, Fo
rtran, Java, Objective-C)
compilers/intel/15.0.3: Sets the environment for using intel compiler (C, Fortra
n)
gnuplot              : loads the environment for gnuplot binary (gnuplot 5.0 pat
chlevel 3)
libs/nag             : loads the environment for NAG library
matlab/2010b         : loads the environment variables for MATLAB R2010b
matlab/2015a         : loads the environment variables for MATLAB R2015a
matlab/2016a         : loads the environment variables for MATLAB R2016a (runtim
e only)
matlab/2016b         : loads the environment variables for MATLAB R2016b (runtim
e only)
mpi/mpich-x86_64     : sets the environement for using MPICH MPI libraries
mpi/openmpi          : sets the environment for using openmpi MPI libraries
openfoam-4.1         : loads the environment for OpenFOAM 4.1
openfoam-v1612+      : loads the environment for OpenFOAM
```

Cargar un módulo

```
$ module load compilers/gcc/6.3.0
```

Mostrar la lista de módulos cargados

```
$ module list
Currently Loaded Modulefiles:
  1) /compilers/gcc/6.3.0
```

Ciertos módulos, habitualmente los que están en el mismo directorio, son incompatibles entre sí.
En este caso, al cargar un módulo incompatible con uno que tengamos cargado, se muestra un mensaje de error:

```
$ module load compilers/gcc/4.4.7
compilers/gcc/4.4.7(20):ERROR:150: Module 'compilers/gcc/4.4.7' conflicts with the
currently loaded module(s) 'compilers/gcc/6.3.0'
compilers/gcc/4.4.7(20):ERROR:102: Tcl command execution failed: conflict
compilers
```

En este caso es necesario descargar previamente el módulo incompatible.

Descargar un módulo

```
$ module unload compilers/gcc/6.3.0
```

```
[acaso@master ~]$ module list
No Modulefiles Currently Loaded.
```

Comprobar el entorno

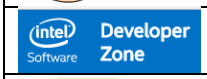
El uso de módulos no es incompatible con el uso de variables de entorno, ni modifica el comportamiento del sistema en manera alguna. Es transparente para las aplicaciones y procesos. Simplemente establece y modifica las variables de entorno dependiendo de las necesidades de cada aplicación.

```
$ module load compilers/gcc/6.3.0
$ which gcc
/LUSTRE/apps/gcc/6.3.0/bin/gcc
$ gcc --version
gcc (GCC) 6.3.0
Copyright (C) 2016 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
$ module unload compilers/gcc/6.3.0
$ module load compilers/gcc/4.4.7
$ which gcc
/usr/bin/gcc
$ gcc --version
gcc (GCC) 4.4.7 20120313 (Red Hat 4.4.7-17)
Copyright (C) 2010 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

Entornos específicos para las aplicaciones y programas instalados

Esta lista es necesariamente incompleta puesto que la lista de software se modifica con el tiempo según las necesidades, pero algunos de los productos comunes instalados son:

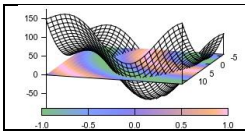
Compiladores y lenguajes de programación:

	GNU/GCC	C, C++, OBJECTIVE-C, FORTRAN y JAVA
	INTEL ICC	C, C++, FORTRAN
	nVIDIA	CUDA, lenguaje especializado en ejecutarse de forma paralela en las tarjetas gráficas nVidia.
	Python	Python, lenguaje muliplataforma, multiparadigma (imperativa, orientado a objetos y funcional), de tipado dinámico y orientado a producir código conciso y legible.
	R	Lenguaje y entorno orientado a la computación estadística y la visualización de datos.

Infraestructura de paralelización:

	OpenMPI	Librería Opensource para la comunicación y sincronización de procesos que se ejecutan en paralelo, a distintos niveles (core, cpu, slot, placa y cluster)
MPICH	MPICH	Otra librería alternativa con el mismo propósito que OpenMPI.

Representación gráfica:

	GNUPLOT	Utilidad de línea de comando para la representación gráfica

Librerías:

boost C++ LIBRARIES	BOOST	La conocida librería de funcionalidades comunes para C++.
	ANACONDA	Distribución de librerías para Python. Incluye Numpy y Scipy
	CAPD	Librería en C++ orientada a la prueba asistida por computador para grupos dinámicos.
	GMP	Librería para operaciones aritméticas de precisión arbitraria
	ISL	Librería para la manipulación de conjuntos y relaciones de puntos enteros limitados por restricciones lineales.
	MPC	Librería para el cálculo aritmético de números complejos con precisión arbitraria.
	MPFR	Librería en C para el cálculo aritmético de números en coma flotante con precisión arbitraria.
	NAG	Librería en fortran de algoritmos numéricos.
	QD	Una librería para C/C++ y FORTRAN90 para el uso de números de doble y cuádruple precisión (32 y 64 dígitos decimales aproximadamente).
	GSL	Librería GNU para cálculo científico

Aplicaciones y entornos de programación:

	Mathematica	Sistema de computación técnica
MathWorks®	Matlab	Incorpora los RUNTIME para distintas versiones, como R2010b, R2015a, R2016a y R2016b.

	OPENFOAM	Software para la computación en dinámica de fluídos (CFD)

Generación de módulos específicos de usuario

Es posible la generación de módulos específicos de usuario. Para ello, se incluye un módulo que genera un directorio conteniendo un fichero de ejemplo.

La sintaxis del fichero y el modo de configuración puede consultarse en la página de referencia mediante `$ man modulefile.`

Generar la estructura para disponer de ficheros de módulo propios:

```
$ module load use.own
```

Esto creará un directorio en $HOME llamado `privatemodules`, y un fichero `null` que es una plantilla para la creación de nuestro propio fichero de módulos:

```
$ ls privatemodules/
Null
$ cat privatemodules/null
#%Module#####################################################################
##
## null modulefile
##
proc ModulesHelp { } {
        global version

        puts stderr "   This module does absolutely nothing."
        puts stderr "   It's meant simply as a place holder in your"
        puts stderr "   dot file initialization."
        puts stderr "
        Version $version
"
}

module-whatis   "does absolutely nothing"

# for Tcl script use only
set     version      3.2.10
```