Parallel Job Support in the Spanish NGI

Enol Fernández del Castillo

Instituto de Física de Cantabria (IFCA)
Spain

Introduction (I)

- Parallel applications are common in clusters and HPC systems
- Grid infrastructures are capable, but
  - HEP grid users not much interested in parallel jobs
  - Hard to deal with underlying heterogeneity
    - MPI implementations, batch systems, file systems,…
- Users must deal with the complexity by themselves

Introduction (II)

- Execution:
  - File distribution
  - Batch system interaction
  - MPI implementation details

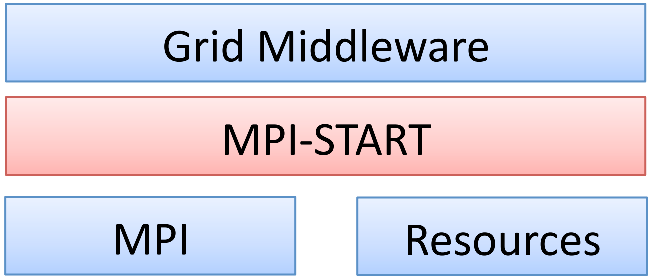
MPI-Start

- Submission:
  - Definition of job characteristics
  - Search and select adequate resources
  - Allocate (or coallocate) resources for the job

CrossBroker

MPI-Start goals

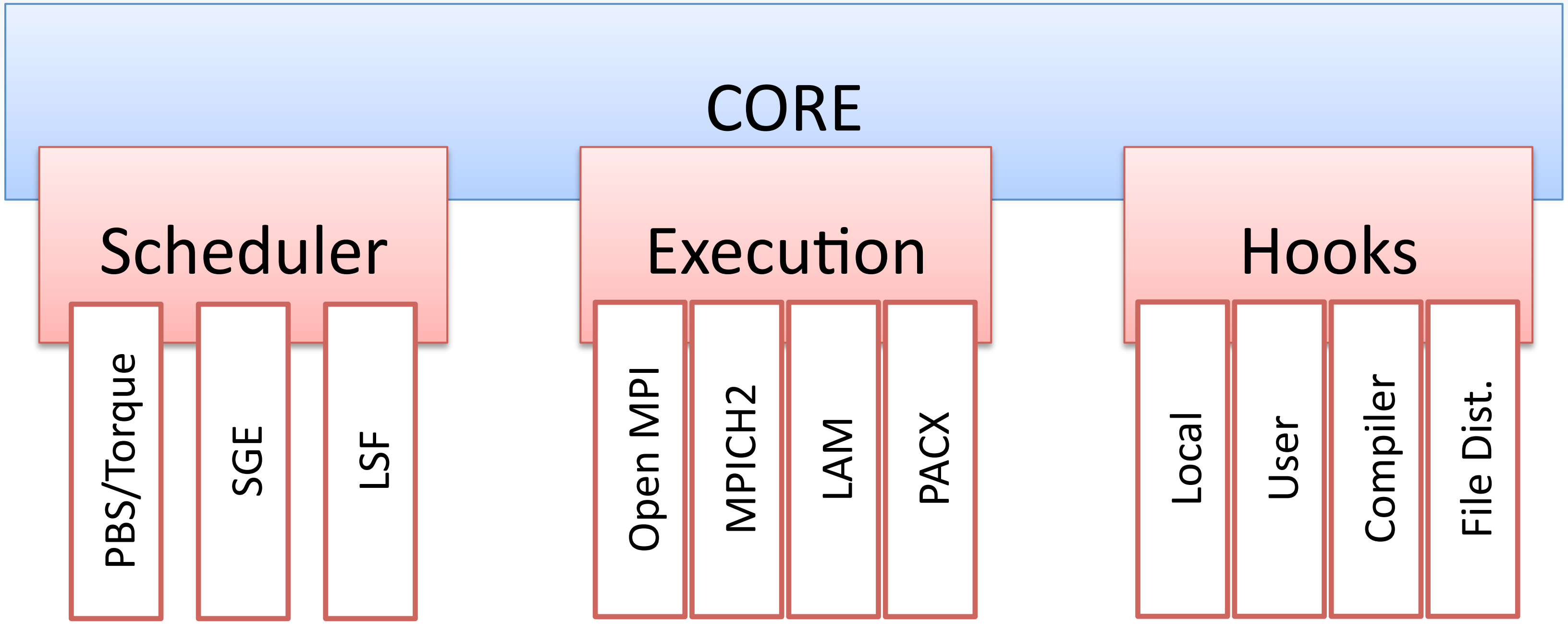
- Specify a unique interface to the upper layer to run a MPI job
- Allow the support of new MPI implementations without modifications in the Grid middleware
- Support of "simple" file distribution
- Provide some support for the user to help manage his data



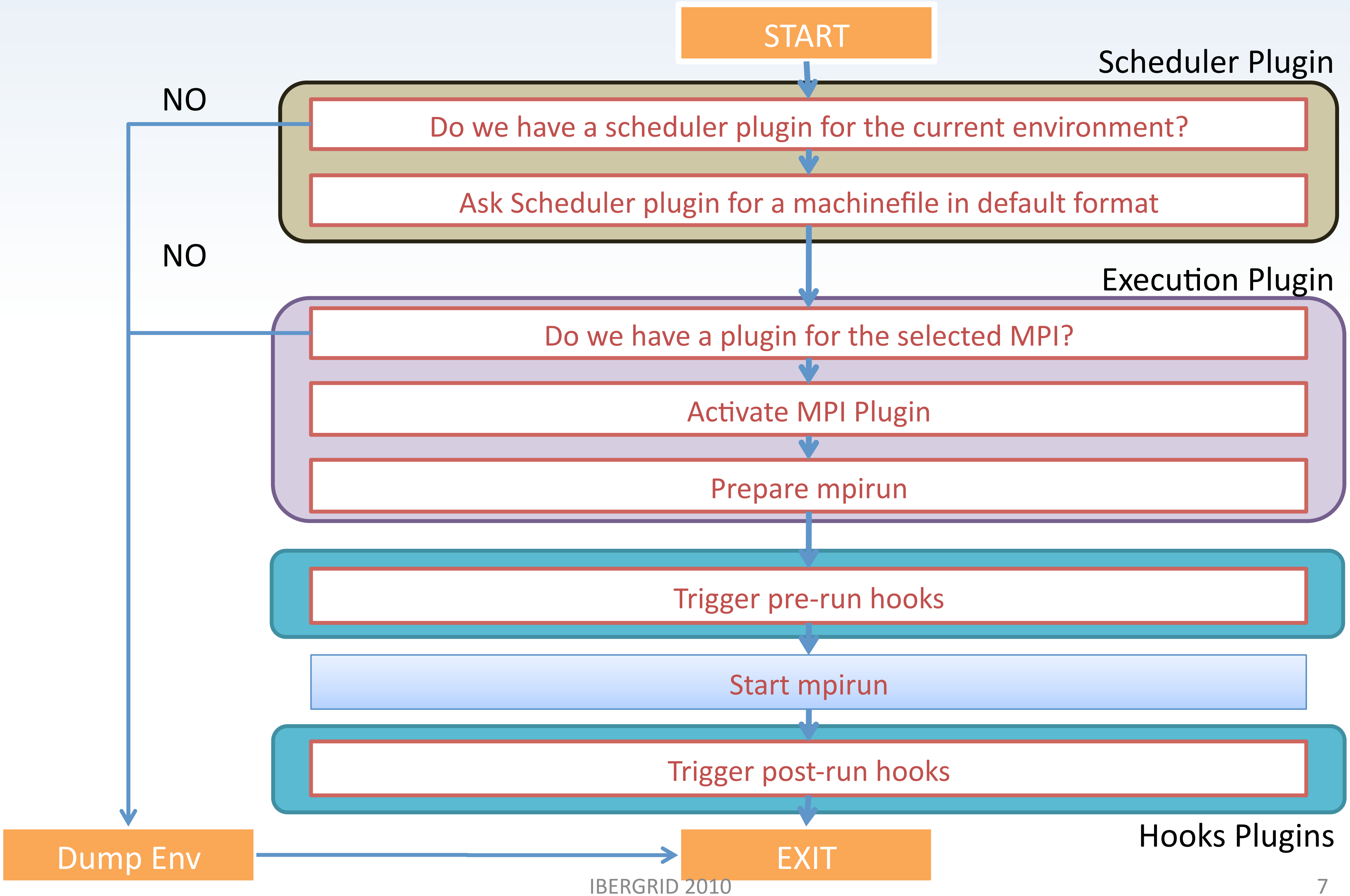
MPI-Start design goals

- Portable
  - The program must be able to run under any supported operating system
- Modular and extensible architecture
  - Plugin/Component architecture
- Relocatable
  - Must be independent of absolute path, to adapt to different site configurations
  - Remote "injection" of mpi-start along with the job
- "Remote" debugging features

MPI-Start architecture



MPI-Start flow


START
Scheduler Plugin
NO
Do we have a scheduler plugin for the current environment?
Ask Scheduler plugin for a machinefile in default format
NO
Execution Plugin
Do we have a plugin for the selected MPI?
Activate MPI Plugin
Prepare mpirun
Trigger pre-run hooks
Start mpirun
Trigger post-run hooks
Dump Env
EXIT
Hooks Plugins

MPI-Start: batch systems

Batch system support

- PBS/Torque (integration with OSC mpiexec if found), LSF and SGE

```
mpi-start [DEBUG  ]: enable debugging
mpi-start [INFO   ]: search for scheduler
mpi-start [DEBUG  ]: source /opt/i2g/bin/../etc/mpi-start/lsf.scheduler
mpi-start [DEBUG  ]: checking for scheduler support : lsf
mpi-start [DEBUG  ]:  checking for $LSB_HOSTS
mpi-start [DEBUG  ]: source /opt/i2g/bin/../etc/mpi-start/pbs.scheduler
mpi-start [DEBUG  ]: checking for scheduler support : pbs
mpi-start [DEBUG  ]:  checking for $PBS_NODEFILE
mpi-start [DEBUG  ]: source /opt/i2g/bin/../etc/mpi-start/sge.scheduler
mpi-start [DEBUG  ]: checking for scheduler support : sge
mpi-start [DEBUG  ]:  checking for $PE_HOSTFILE
mpi-start [INFO   ]: activate support for sge
mpi-start [DEBUG  ]:  convert PE_HOSTFILE into standard format
mpi-start [DEBUG  ]: dump machinefile:
mpi-start [DEBUG  ]: => gcsic015wn.ifca.es
mpi-start [DEBUG  ]: => gcsic015wn.ifca.es
mpi-start [DEBUG  ]: => cms15wn.ifca.es
mpi-start [DEBUG  ]: => cms15wn.ifca.es
mpi-start [DEBUG  ]: starting with 4 processes.
```

MPI-Start: hooks (I)

Compiler detection:

```
mpi-start [DEBUG  ]: mpi_start_check_compiler_flags
mpi-start [DEBUG  ]: detected 32 bit compiler flags in 64 bit system, will try to fix them
mpi-start [DEBUG  ]: Updating MPI_MPICC_OPTS variable.
mpi-start [DEBUG  ]: Updating MPI_MPICXX_OPTS variable.
mpi-start [DEBUG  ]: Updating MPI_MPIF90_OPTS variable.
```

User hooks (compilation/data fetching):

```
pre_run_hook () {
   echo "pre run hook called ”
   # - compile program
   mpicc $MPI_CC_OPTS –o $I2G_MPI_APPLICATION $I2G_MPI_APPLICATION.c

   # - fetch input data
   /opt/lcg/bin/lcg-cp -v --vo ${MY_VO} lfn:${LFC_DIR}/${DATA} file://`pwd`/${DATA}
   return 0
}

pre_run_hook () {
   echo "post run hook called "
   # - upload results
   /opt/lcg/bin/lcg-lr -v --vo ${MY_VO} lfn:${LFC_DIR}/${DATA}
   return 0
}
```

MPI-Start: hooks (II)

Detection of shared filesystems:

```
mpi-start [DEBUG ]: mpi_start_pre_run_hook
mpi-start [DEBUG ]: mpi_start_pre_run_hook_generic
mpi-start [DEBUG ]: detect shared filesystem
mpi-start [DEBUG ]: dump mount point information:
mpi-start [DEBUG ]: => / = ext3
mpi-start [DEBUG ]: => /proc = proc
mpi-start [DEBUG ]: => /dev/pts = devpts
mpi-start [DEBUG ]: => /proc/bus/usb = usbdevfs
mpi-start [DEBUG ]: => /dev/shm = tmpfs
mpi-start [DEBUG ]: => /opt/exp_soft = nfs
mpi-start [DEBUG ]: => /root/conf_ieg = nfs
mpi-start [DEBUG ]: current working directory : /home/ngiops/globus- tmp.wn12-ieg.650.0/
https_3a_2f_2fi2g-rb01.lip.pt_3a9000_2f72Bm6Y433WNkuIA9eTTH8A_0
mpi-start [DEBUG ]: found local fs : ext3
```

File distribution:

```
mpi-start [DEBUG ]: mpi_start_post_run_hook_copy_ssh
mpi-start [DEBUG ]: fs not shared -> distribute binary
mpi-start [DEBUG ]: distribute "/bin/hostname" to remote node : cms15.ifca.es
mpi-start [DEBUG ]: skip local machine
```

MPI-Start: hooks (III)

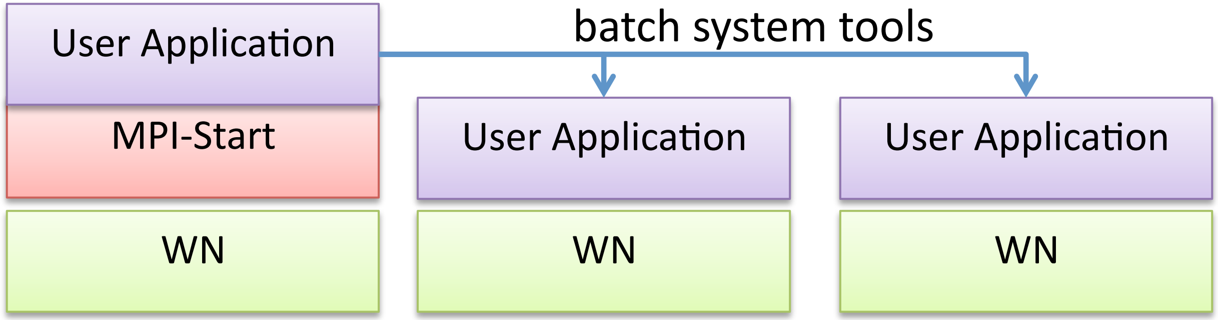
- File Distribution Methods
  - Copy to shared filesystem
    - The admin can define a path on a shared filesystem (different from $HOME) where the job can be run
  - SSH
    - Uses ssh/scp to copy the files to the remote machines. It needs password-less ssh
  - Mpiexec
    - Copies the files using mpiexec
  - Mpi_mt
    - Runs mpi_mt binary that copies the files to the nodes using mpi

MPI-Start: beyond MPI (I)

- Hybrid OpenMP/MPI applications:
  - force MPI-Start to initiate 1 process per allocated node
  - the env. variable I2G_MPI_NODE_SLOTS will be set with the number of slots in each node
    - applications should no use more than those slots
    - same value exported as OMP_NUM_THREADS
  - enable the behavior with I2G_MPI_SINGLE_PROCESS variable

MPI-Start: beyond MPI (II)

- Workflows in a set of nodes:
  - MPI-Start prepares the environment (detects allocated machines, does file transfer)
  - Lets the application start "subjobs" at the nodes
    - using native tools of batch system (e.g. qrsh in SGE)
    - ssh as a fall-back method
  - Kepler actor ready for using MPI-Start this way in Euforia



MPI-Start: more features

- Remote injection
  - Mpi-start can be sent along with the job
    - Just unpack, set environment and go!
- Interactivity
  - A pre-command can be used to “control” the mpirun call
  - $I2G_MPI_PRECOMMAND mpirun ….
  - This command can:
    - Redirect I/O
    - Redirect network traffic
    - Perform accounting
- Debugging
  - 3 different debugging levels:
    - VERBOSE: basic information
    - DEBUG: internal flow information
    - TRACE: set –x at the beginning. Full trace of the execution

CrossBroker

- CrossBroker is a grid metascheduler with automatic support for parallel and interactive jobs
  - interoperable with the gLite middleware
  - Open MPI, PACX-MPI, MPICH, MPICH2 and MPICH-G2 support with some JDL changes
  - Integration with MPI-Start

```
Type          = "Job";
JobType       = "Parallel";
CPUNumber    = 23;
SubJobType    = "openmpi";
Executable    = "my_app";
Arguments     = "-n 356 -p 4";
StdOutput     = "std.out";
StdError      = "std.err";
InputSandBox  = {"my_app"};
OutputSandBox = {"std.out", "std.err"};
```

CrossBroker: new job types

- Collections:
  - set of related jobs submitted with a single JDL
- Parametric jobs:
  - set of jobs that explore a parameter space
  - possibility of defining more than one parameter

```
JobType = "Parametric";
Executable = "myexec";
StdInput = "input-_PARAM_A_-_PARAM_B_.txt";
StdOutput = "output-_PARAM_A_-_PARAM_B_.txt";
StdError = "error-_PARAM_A_-_PARAM_B_.txt";
Parameters_A = {alpha, beta};
Parameters_B = 2;
ParameterStart_B = 0;
ParameterStep_B = 1;
InputSandbox = {"input-_PARAM_A_-_PARAM_B_.txt”}
```

CrossBroker: new job description (I)

- New JDL description for parallel jobs:
  - WholeNodes (True/False):
    - whether or not full nodes should be reserved
  - NodeNumber (default = 1):
    - number of nodes requested
  - SMPGranularity (default = 1):
    - minimum number of cores per node
  - CPUNumber (default = 1):
    - number of job slots (processes/cores) to use
- Not supported (yet!) by the CEs
  - already proposed by the EGEE MPI TF

CrossBroker: new job description (II)

- Multithread application with 4 threads in a single node:

```
…
SMPGranularity = 4;
WholeNodes = True;
…
```

- MPI job with 1 process per node:

```
…
NodeNumber = 16;
CPUNumber = 16;
…
```

- Hybrid MPI/OpenMP:

```
…
NodeNumber = 4;
WholeNodes True
SMPGranularity = 4;
…
```

Conclusions & Future work

- MPI-Start and CrossBroker provide a complete framework for parallel job execution
  - Hides underlying complexity with a uniform interface, and at the same time provides advanced features (hooks, job definition)
  - both projects maintained in the Spanish NGI
- MPI-Start official in EGEE, effort continued in EMI
- Future work:
  - better integration of non MPI jobs
  - Make CEs able to allocate jobs as defined