Hands-On Session
Resource Specification Language

Display the hostname of remote machine

RSL File: **hostname.rsl**

```
+
(
      &(resourceManagerContact="che01")
      (count=2)
      (executable=/bin/hostname)
)
```

Command to submit the job:

$globusrun –o –f hostname.rsl

Display the content of a directory in a remote machine

RSL File : ls.rsl

```
+
(
      &(resourceManagerContact="che01")
      (count=1)
      (executable=/bin/ls)
      (directory=/tmp)
)
```

Command to submit the job:

$globusrun –o –f ls.rsl

Submit a Job with arguments

RSL File : ls-l.rsl

```
+
(
      &(resourceManagerContact="che01")
      (count=1)
      (executable=/bin/ls)
```

```
      (arguments="-l")
      (directory=/tmp)
)
```

Command to submit the job:

$globusrun -o –f ls-l.rsl

Sequential Program: Matrix Multiplication

```
/* Matrix_Mul.c
**
** multiply two 3X3 matrices.
**
*/

#include <stdio.h>
void mult_matrices(int a[][3], int b[][3], int result[][3]);
void print_matrix(int a[][3]);

int main(void)
{
  int p[3][3] = {  {1, 3, -4}, {1, 1, -2}, {-1, -2, 5}  };
  int q[3][3] = {  {8, 3, 0}, {3, 10, 2}, {0, 2, 6}  };
  int r[3][3];

  mult_matrices(p, q, r);
  print_matrix(r);
}

void mult_matrices(int a[][3], int b[][3], int result[][3])
{
  int i, j, k;
  for(i=0; i<3; i++)
  {
      for(j=0; j<3; j++)
      {
        for(k=0; k<3; k++)
        {
            result[i][j] = a[i][k] + b[k][j];
```

```
        }
      }
  }
}

void print_matrix(int a[][3])
{
  int i, j;
  for (i=0; i<3; i++)
  {  for (j=0; j<3; j++)
      {
        printf("%d\t", a[i][j]);
      }
      printf("\n");
  }
}
```

Compilation on Che01:

$gcc -o Matrix_Mul Matrix_Mul.c

RSL File: Matrix_Mul.rsl

```
+
(
 &(resourceManagerContact="che01/jobmanager-fork ")
   (label="subjob 0")
  (environment=(LD_LIBRARY_PATH /usr/local/globus2.4.3 ) )
  (directory="/home/workflow/DAG-07/Application_Development")
  (executable="Matrix_Mul")
  (arguments="" )
  (stdout="/home/workflow/DAG-07/Application_Development/out")
  (stderr="/home/workflow/DAG-07/Application_Development/err")
 )
```

Command To Submit Job:

$globusrun –f Matrix_Mul.rsl

Parallel Program: MPI_Matrix_Mul.c

```
#include <stdio.h>
#include "mpi.h"
```

```
#define NRA 3
#define NCA 3
#define NCB 3
#define MASTER 0
#define FROM_MASTER 1
#define FROM_WORKER 2

MPI_Status status;
main(int argc, char **argv)
{
int numtasks,
   taskid,
   numworkers,
   source,
   dest,
   nbytes,
   mtype,
   intsize,
   dbsize,
   rows,
   averow, extra, offset,
   i, j, k,
   count;
 double a[NRA][NCA],
      b[NCA][NCB],
      c[NRA][NCB];

intsize = sizeof(int);
dbsize = sizeof(double);


MPI_Init(&argc, &argv);
MPI_Comm_rank(MPI_COMM_WORLD, &taskid);
MPI_Comm_size(MPI_COMM_WORLD, &numtasks);
numworkers = numtasks-1;

/**************************** master task
************************************/
if (taskid == MASTER) {
  printf("Number of worker tasks = %d\n",numworkers);
```

```
 for (i=0; i<NRA; i++)
   for (j=0; j<NCA; j++)
 a[i][j]= i+j;
 for (i=0; i<NCA; i++)
   for (j=0; j<NCB; j++)
     b[i][j]= i*j;

 /* send matrix data to the worker tasks */
 averow = NRA/numworkers;
 extra = NRA%numworkers;
 offset = 0;
 mtype = FROM_MASTER;
 for (dest=1; dest<=numworkers; dest++) {
   rows = (dest <= extra) ? averow+1 : averow;
   printf("   sending %d rows to task %d\n",rows,dest);
   MPI_Send(&offset, 1, MPI_INT, dest, mtype, MPI_COMM_WORLD);
   MPI_Send(&rows, 1, MPI_INT, dest, mtype, MPI_COMM_WORLD);
   count = rows*NCA;
   MPI_Send(&a[offset][0], count, MPI_DOUBLE, dest, mtype,
MPI_COMM_WORLD);
   count = NCA*NCB;
   MPI_Send(&b, count, MPI_DOUBLE, dest, mtype,
MPI_COMM_WORLD);

   offset = offset + rows;
   }

 /* wait for results from all worker tasks */
 mtype = FROM_WORKER;
 for (i=1; i<=numworkers; i++) {
   source = i;
   MPI_Recv(&offset, 1, MPI_INT, source, mtype, MPI_COMM_WORLD,
&status);
   MPI_Recv(&rows, 1, MPI_INT, source, mtype, MPI_COMM_WORLD,
&status);
   count = rows*NCB;
   MPI_Recv(&c[offset][0], count, MPI_DOUBLE, source, mtype,
MPI_COMM_WORLD,
            &status);
```

```
    }

  /* print results */
  printf("Here is the result matrix\n");
  for (i=0; i<NRA; i++) {
printf("\n");
    for (j=0; j<NCB; j++)
      printf("%6.2f   ", c[i][j]);
    }
  printf ("\n");

  }  /* end of master section */



/***************************** worker task
************************************/
if (taskid > MASTER) {
  mtype = FROM_MASTER;
  source = MASTER;
  printf ("Master =%d, mtype=%d\n", source, mtype);
  MPI_Recv(&offset, 1, MPI_INT, source, mtype, MPI_COMM_WORLD,
&status);
  printf ("offset =%d\n", offset);
  MPI_Recv(&rows, 1, MPI_INT, source, mtype, MPI_COMM_WORLD,
&status);
  printf ("row =%d\n", rows);
  count = rows*NCA;
  MPI_Recv(&a, count, MPI_DOUBLE, source, mtype,
MPI_COMM_WORLD, &status);
  printf ("a[0][0] =%e\n", a[0][0]);
  count = NCA*NCB;
  MPI_Recv(&b, count, MPI_DOUBLE, source, mtype,
MPI_COMM_WORLD, &status);
  printf ("b=\n");
  for (k=0; k<NCB; k++)
    for (i=0; i<rows; i++) {
      c[i][k] = 0.0;
      for (j=0; j<NCA; j++)
        c[i][k] = c[i][k] + a[i][j] * b[j][k];
```

```
    }

  mtype = FROM_WORKER;
  printf ("after computer\n");
  MPI_Send(&offset, 1, MPI_INT, MASTER, mtype,
MPI_COMM_WORLD);
  MPI_Send(&rows, 1, MPI_INT, MASTER, mtype,
MPI_COMM_WORLD);
  MPI_Send(&c, rows*NCB, MPI_DOUBLE, MASTER, mtype,
MPI_COMM_WORLD);
  printf ("after send\n");

  }  /* end of worker */
  MPI_Finalize();
} /* of main */
```

Compilation for Homogenous job on che01:

$ /usr/local/mpich-1.2.6/bin/mpicc –o MPI_Matrix_Mul MPI_Matrix_Mul.c

RSL File:(MPI_Matrix_Mul.rsl)

```
 +
(
 &(resourceManagerContact="che01/jobmanager-fork")
   (label="subjob 0")
   (jobtype = "mpi")
  (count = 3)
  (environment=(LD_LIBRARY_PATH $(GLOBUS_LOCATION)/lib)
(GLOBUS_DUROC_SUBJOB_INDEX 0) )
  (directory="/home/workflow/DAG-07/Application_Development")
  (executable="/home/workflow/DAG-07/Application_Development/MPI_Matrix_Mul")
  (arguments="" )
 (stdout="/home/workflow/DAG-07/Application_Development/Mpi_out”)
 (stderr="/home/workflow/DAG-07/Application_Development/Mpi_err”)
 )
```

Job Submission :

$globusrun –f MPI_Matrix_Mul.rsl

Compilation for Heterogeneous job on Che01:

$/usr/local/mpichG2/bin/mpicc –o Mpi_Hetro MPI_Matrix_Mul.c

RSL File: MPI_Hetro_Matrix.rsl

```
+
(
 &(resourceManagerContact="che01/jobmanager-fork")
   (label="subjob 0")
  (count = 3)
  (environment=(LD_LIBRARY_PATH $(GLOBUS_LOCATION)/lib
)(GLOBUS_DUROC_SUBJOB_INDEX 0) )
  (directory="/home/workflow/DAG-07/Application_Development")
  (executable="/home/workflow/DAG-07/Application_Development/MPI_Hetro")
  (stdout="/home/workflow/DAG-07/Application_Development/MPI_out")
  (stderr="/home/workflow/DAG-07/Application_Development/MPI_err")
  (arguments="" )
 )
```

Job Submission :

$globusrun –f MPI_Hetro_Matrix.rsl