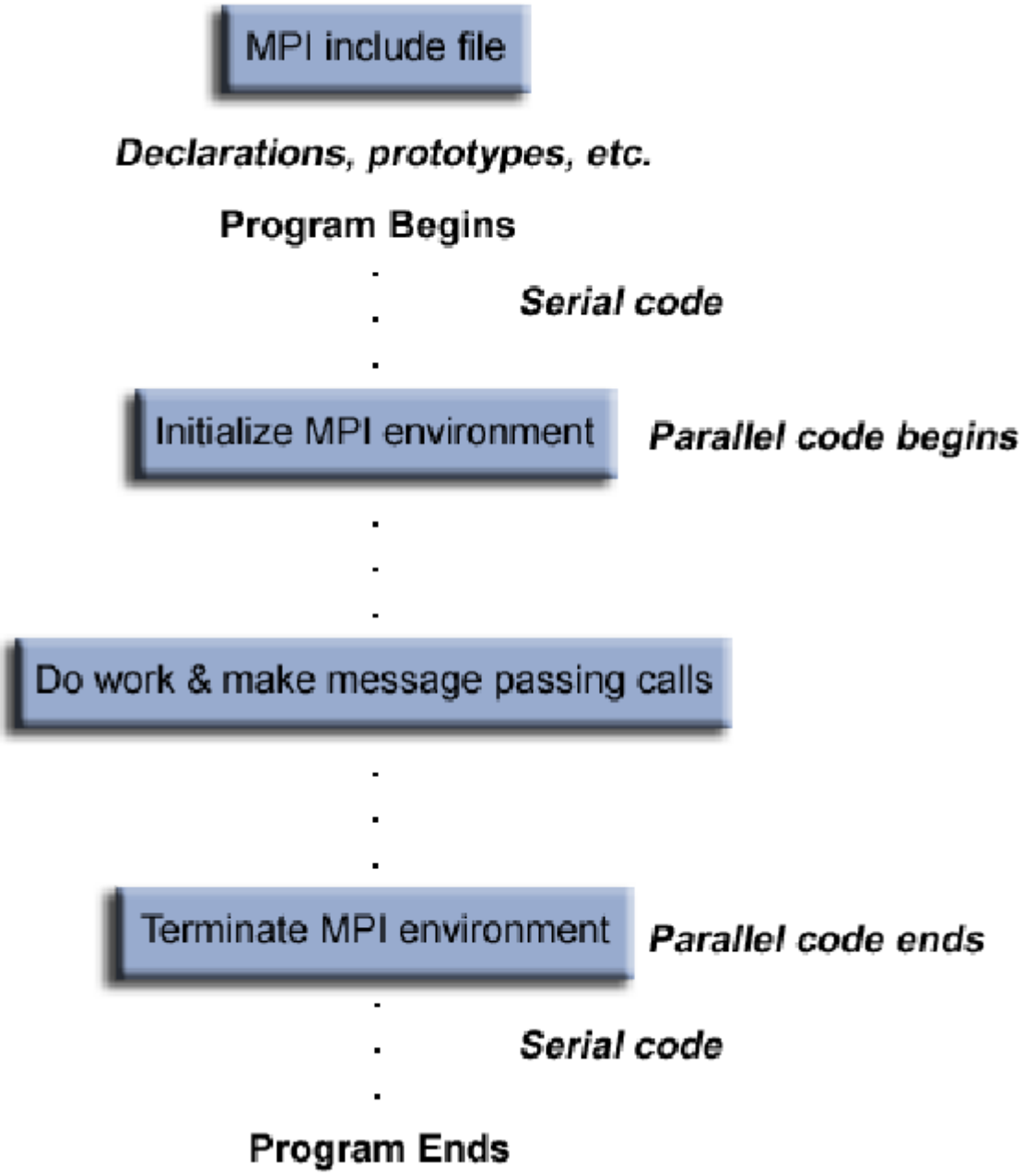
MPI program stucture

MPI uses **objects** called **communicators** and **groups** to define which **collection of processes** may communicate with each other. Most MPI routines require you to specify a communicator as an argument.



MPI program schema

It is very important to learn the MPI program structure before we proceed to

coding message passing programs. Following are the common functions used while developing an MPI code:

MPI Init:

```
MPI_Init (&argc,&argv)
MPI_INIT (ierr)
```

MPI intialisation.

MPI_Comm_size:

```
MPI_Comm_size (comm,&size)
MPI_COMM_SIZE (comm,size,ierr)
```

Determines the number of processes in the group associated with a communicator.

MPI_Comm_rank:

```
MPI_Comm_rank (comm,&rank)
MPI_COMM_RANK (comm,rank,ierr)
```

Determines the rank (task ID) of the calling process within the communicator. Value 0...p-1

MPI_Abort:*

```
MPI_Abort (comm,errorcode)
MPI_ABORT (comm,errorcode,ierr)
```

Terminates all MPI processes associated with the communicator.

MPI_Finalize:

```
MPI_Finalize ()
MPI_FINALIZE (ierr)
```

Terminates the MPI execution environment. This function should be the last MPI routine called in every MPI program - no other MPI routines may be called after it.

Running MPI jobs under OpenMPI and batch systems

Running MPI jobs under OpenMPI and batch systems

MPI jobs under OpenMPI require the use of the `mpirun` command. **OpenMPI** will automatically detect when it is running in a batch environment and assign processes to the assigned resources. The policy for placing processes can be modified using arguments to the `mpirun` command. See `man mpirun` for details.

```
#!/bin/bash
#PBS -l nodes=3:ppn=12
cd $PBS_O_WORKDIR
module load openmpi
mpirun ./a.out
```