OpenDIEL: A Parallel Open Source Workflow Engine
Students: Tristin Baker (Maryville College), Jordan Scott (Morehouse College),
Zac Trzil (UTK), Mentor: Dr. Kwai Wong (UTK)

What is OpenDIEL?
OpenDIEL stands for open Distributive Interoperable Executable Library. It facilitates communication between loosely coupled modules leveraging the MPI (Message Parsing Interface) system. With a user-defined configuration file and driver, openDIEL can output a single executable managing the communication between multiple modules, allowing for efficient data sharing between unique, data-intensive modules.

Distributed Tuple Space
Overview: Modules may use a distributed array of tuple servers to store data in system memory that other modules may access. The sender places the data using IEL_dist_put() and a user-defined data tag as an argument of the function. The receiver, using the same tag and the IEL_dist_get() function will be able to retrieve the data from the distributed array.

Sending data: A client can send data to the distributed array of tuple servers by calling IEL_dist_put():
- Distributes data even among available tuple servers
- Sets up two arrays of meta data:
  - The server rank in the order used
  - The size of the data sent to that tuple server
- Stores the meta-data on the first tuple server

Better Direct Communication Example
Previously, the examples showing off the direct communication aspect of the OpenDIEL did not adequately show the OpenDIEL’s capabilities as well as it could have. Therefore, an example using Laplace Transformation Matrices was created.

User Interface Improvement
The User Interface serves to replace how OpenDIEL currently operates. It does this by allowing users to enter information directly into a single Interface as opposed to editing multiple files, and lines of code.

The Interface will allow for users to enter modules, groups and sets. This information will be used to make the “workflow.cfg” file necessary for running OpenDIEL.

Future work for the User Interface are
1. Fully implement functionality listed above.
2. Allow users to convert their C or Fortran code into an OpenDIEL module.
3. To implement an open database of Modules.
4. Launch OpenDIEL executables on HPCs through the GUI via SSH.
5. Create runscripts by analyzing user code and determining how many processes the OpenDIEL executable needs.

Distributed Tuple Space
Receiving data: A client can receive data stored on the distributed array of tuple servers by calling IEL_dist_get():
- Queries the meta data server for the information corresponding to the tag the function was called with
- Uses the meta data to pull the data from the servers in the order in which is was stored
- Reconstructs the data into an array that the client passed to the function

Results: Distributing data across multiple tuple servers shows almost no increase in program running time while file I/O grows at an exponential rate.

The above graph shows the running time of two modules sharing data through openDIEL, one sending and one receiving data.

Future Work
Taking advantage of the distributed tuple servers, data can be duplicated across the servers to provide RAID-like failure protection. A single or subset of tuple servers can be designated to backup to disk without interfering with the primary processes and their data.

Direct communication should take place in local a shared_bc rather than a global. In its current implementation, the shared_bc wastes memory and is not scalable.

Acknowledgements
This project is made possible by funding provided by the NSF, facilities provided by the University of Tennessee, and computational resources provided by XSEDE.