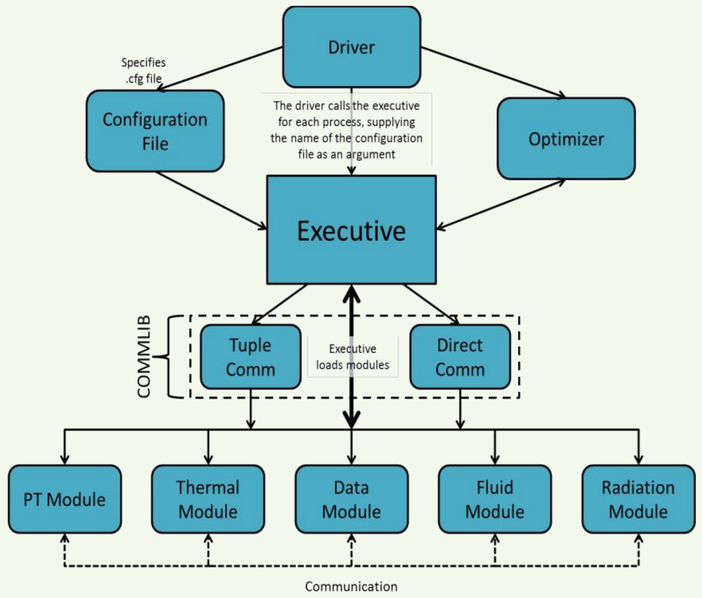
# Workflow and Direct Communication in the openDIEL

Nick Moran (UTK), Tanner Curren (Maryville College)
Mentor: Kwai Wong (UTK)

## About openDIEL

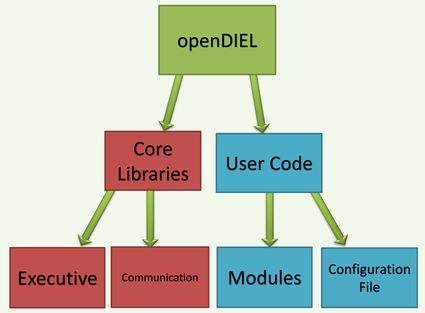
- The **openDIEL** (open Distributive Interoperable Executive Library) aims to facilitate communication between **user-created loosely coupled simulations**.
- **Loosely coupled simulations** are mostly serial programs that rely on data points from other simulations; these simulations get their input from and send output to other simulations.
- The openDIEL communicates through a tuple server-based method, and now is able to communicate large chunks of contiguous memory more efficiently through **direct communication**.
- The organization of these simulations is organized and facilitated by the **workflow** implementation of the openDIEL.

## openDIEL Organization




## User Interaction with openDIEL

- The user interacts with two major parts of the openDIEL: **modules** (parts of the loosely coupled simulation) and the **configuration file**.
- **Modules** are adaptions of the user's code. These are easily generated using **modMaker**.
- The **configuration file** allows the user to easily set up simulations.




Project Organization

```
shared_bc_sizes = []
tuple_space_size=0
modules=(
    {
    function="modep7"
    args=()
    libtype="static"
    library="libmodep7.a"
    splitdir="ep7-rv-workflow"
    size=5
    },
    {
    function="modreadvars"
    args=("in.rvi", "monthly")
    libtype="static"
    splitdir="ep7-rv-workflow"
    library="libmodreadvars.a"
    size=5
    },
  {
    function="RAnalysis"
    args=()
    libtype="static"
    library="libRAnalysis.a"
    splitdir="ep7-rv-r-workflow"
    size=1
  }
)

workflow:
```

Sample Configuration File

## Direct Communication

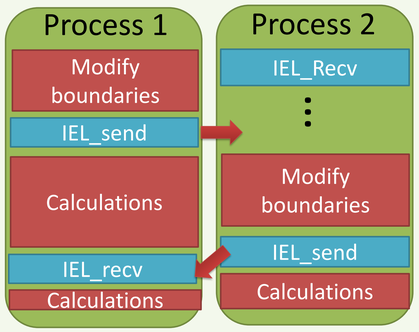
- Wrappers around MPI to communicate large chunks of data
- Functions **IEL_send** (nonblocking) and **IEL_recv** (blocking)
- Uses a set of **shared boundary conditions** in a conceptual grid, each process having access to a different section of this grid

Configuration file using direct communication:

```
shared_bc_sizes = [2, 2, 3]
tuple_space_size = 0;

modules = (
{
    function = "sampletest";
    args = ();
    libtype = "static";
    library = "libsampletest.a";
    size = 2;
    points = (
        (([0,2]),([0,2]),([0,3])),
        (([0,1]),([1,1]),([1,2]))
    );
}
)
```

Example of direct communication flow:




## Workflow

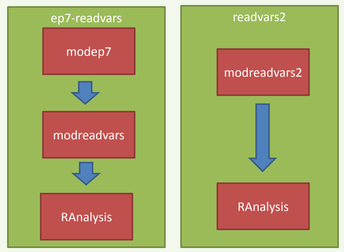
- Users may create **groups**
- Options available are **order** and **iterations**
  - **Order:** The modules in a group
  - **Iterations:** Number of times a groups executes

```
workflow:
{

  groups:
  {
    ep7-readvars:
    {
      order=("modep7", "modreadvars", "RAnalysis")
    }

    readvars2:
    {
      order=("modreadvars2", "RAnalysis")
    }
  }
}
```

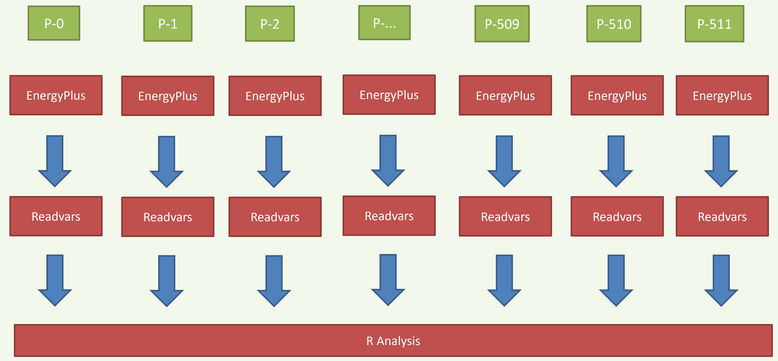
An example of a workflow configuration




How the workflow described above will execute

## Use Case: Energy Plus 7

- Run EnergyPlus in SPMD fashion
- **Workflow** schedules Readvars and R analysis code to follow each EnergyPlus run




## Acknowledgments

Mentorship
Kwai Wong

Special Thanks
NSF: Funding
JICS: Compute Resources, Workspace
ORNL: Hosting