Software Tools

ACF Spring HPC Training Workshop
Match 15-16, 2016

IPM – Integrated Performance Monitoring

- portable
- low-overhead
- selectable details at runtime
- various text and web reports.

Link your code to ipm library

C/C++

$IPM_LIB -lipm

FORTRAN

$IPM_LIB -lipmf -lipm

On ACF:

```
module load ipm
```

(papi, ploticus)

```
mpiicc -o pi_ipm.x pical.c $IPM_LIB -lipm
```

Output

```
##IPMv2.0.6####################################################
#
# command   : ./pi_ipm.x
# start     : Wed Mar 14 23:02:47 2018   host      : acf-login3
# stop      : Wed Mar 14 23:02:47 2018   wallclock : 0.10
# mpi_tasks : 6 on 1 nodes               %comm     : 6.35
# mem [GB]  : 0.10                       gflop/sec : 0.00
#
#           :       [total]        <avg>         min         max
# wallclock :          0.61         0.10        0.10        0.10
# MPI       :          0.04         0.01        0.00        0.01
# %wall     :
#   MPI     :                       6.34        0.05        9.50
# #calls    :
#   MPI     :            42            7           7           7
# mem [GB]  :          0.10         0.02        0.01        0.02
#
###############################################################
```

IPM_REPORT controls the verbose level

export IPM_REPORT=terse(default), full, none

terse

Aggregate wallclock time, memory usage and flops are reported along with the percentage of wallclock time spent in MPI calls

full

Each HPM counter is reported as are all of wallclock, user, system, and MPI time. The contribution of each MPI call to the communication time is given.

none

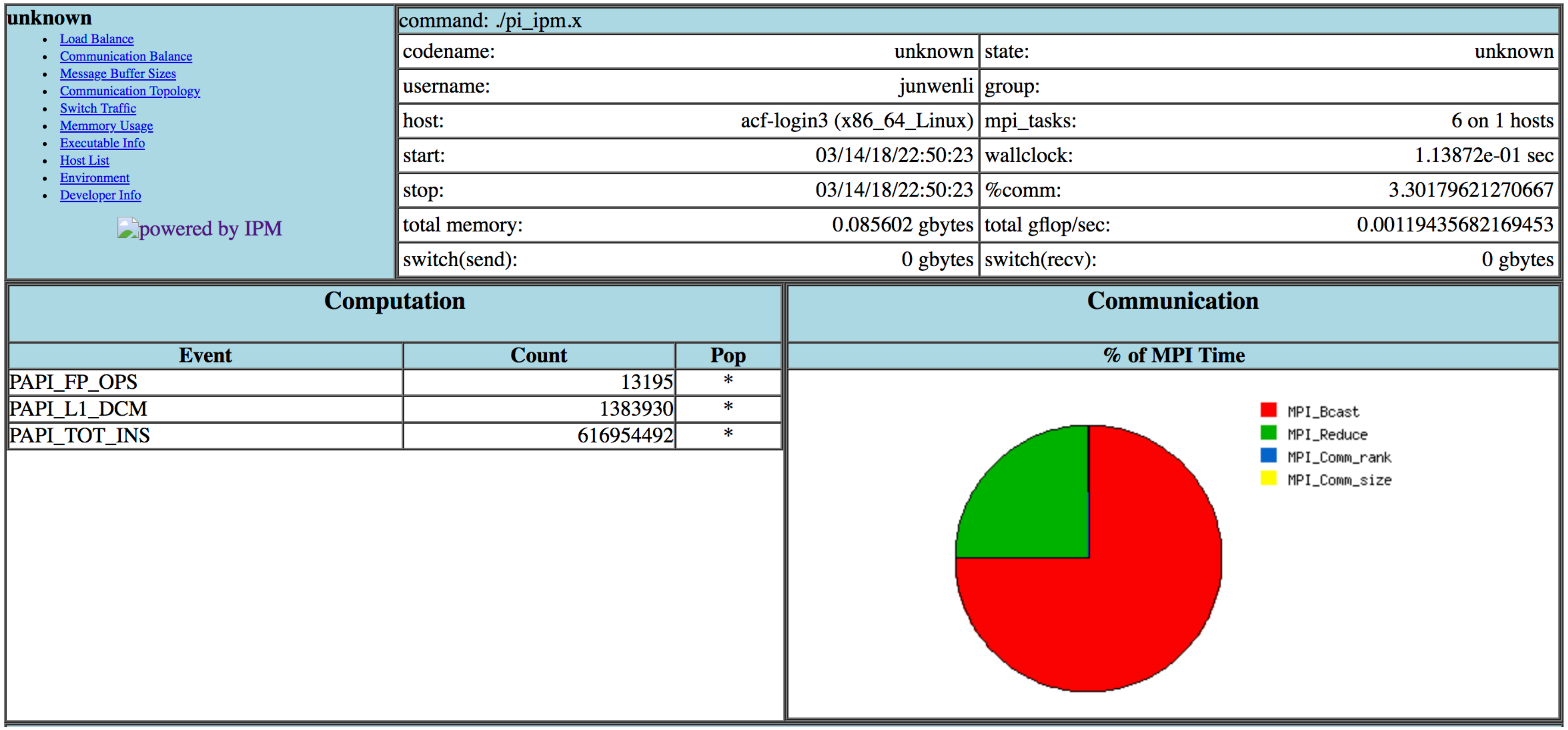
No report

IPM_REPORT=full

```
##IPMv2.0.6########################################################
#
# command   : ./pi_ipm.x
# start     : Wed Mar 14 23:01:32 2018   host      : acf-login3
# stop      : Wed Mar 14 23:01:32 2018   wallclock : 0.10
# mpi_tasks : 6 on 1 nodes               %comm     : 4.19
# mem [GB]  : 0.09                       gflop/sec : 0.00
#
#           :       [total]        <avg>          min          max
# wallclock :          0.59         0.10         0.10         0.10
# MPI       :          0.02         0.00         0.00         0.01
# %wall     :
#   MPI     :                       4.18         0.05         6.29
# #calls    :
#   MPI     :            42            7            7            7
# mem [GB]  :          0.09         0.02         0.01         0.02
#
#                             [time]        [count]        <%wall>
# MPI_Bcast                     0.02             12           3.11
# MPI_Reduce                    0.01              6           1.08
# MPI_Comm_size                 0.00              6           0.00
# MPI_Init                      0.00              6           0.00
# MPI_Finalize                  0.00              6           0.00
# MPI_Comm_rank                 0.00              6           0.00
#
###################################################################
```

ipm_parse to get graphic report

ipm_parse -html junwenli.1521082223.ipm.xml



unknown

- Load Balance
- Communication Balance
- Message Buffer Sizes
- Communication Topology
- Switch Traffic
- Memmory Usage
- Executable Info
- Host List
- Environment
- Developer Info

powered by IPM

command: ./pi_ipm.x

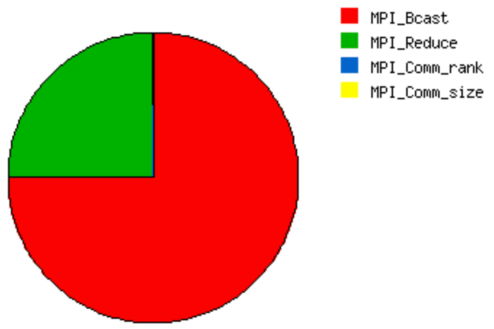
codename:	unknown	state:	unknown
username:	junwenli	group:	
host:	acf-login3 (x86_64_Linux)	mpi_tasks:	6 on 1 hosts
start:	03/14/18/22:50:23	wallclock:	1.13872e-01 sec
stop:	03/14/18/22:50:23	%comm:	3.30179621270667
total memory:	0.085602 gbytes	total gflop/sec:	0.00119435682169453
switch(send):	0 gbytes	switch(recv):	0 gbytes

Computation

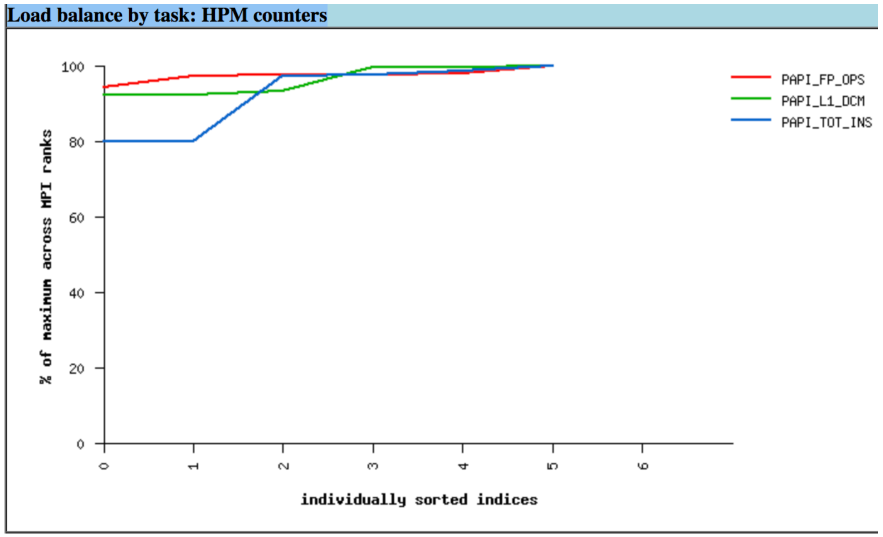
Event	Count	Pop
PAPI_FP_OPS	13195	*
PAPI_L1_DCM	1383930	*
PAPI_TOT_INS	616954492	*

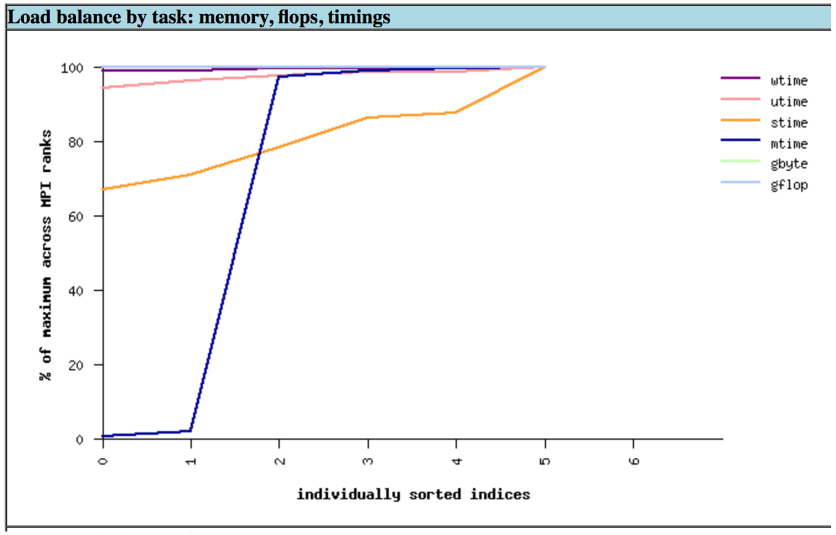
Communication

% of MPI Time



Statistics by MPI rank





papi_avail to check the supported features

export IPM_HPM=PAPI_FP_OPS,PAPI_TOT_INS,PAPI_L1_DCM,PAPI_L1_DCA

papi_avail

```
[junwenli@acf-login3 ipm]$ papi_avail
Available PAPI preset and user defined events plus hardware information.
--------------------------------------------------------------------------------
PAPI Version             : 5.5.1.0
Vendor string and code   : GenuineIntel (1)
Model string and code    : Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz (45)
CPU Revision             : 6.000000
CPUID Info               : Family: 6  Model: 45  Stepping: 6
CPU Max Megahertz        : 3300
CPU Min Megahertz        : 1200
Hdw Threads per core     : 1
Cores per Socket         : 8
Sockets                  : 2
NUMA Nodes               : 2
CPUs per Node            : 8
Total CPUs               : 16
Running in a VM          : no
Number Hardware Counters : 11
Max Multiplex Counters   : 384
--------------------------------------------------------------------------------


================================================================================
  PAPI Preset Events
================================================================================
    Name        Code    Avail Deriv Description (Note)
PAPI_L1_DCM  0x80000000  Yes   No   Level 1 data cache misses
PAPI_L1_ICM  0x80000001  Yes   No   Level 1 instruction cache misses
PAPI_L2_DCM  0x80000002  Yes   Yes  Level 2 data cache misses
PAPI_L2_ICM  0x80000003  Yes   No   Level 2 instruction cache misses
PAPI_L3_DCM  0x80000004  No    No   Level 3 data cache misses
PAPI_L3_ICM  0x80000005  No    No   Level 3 instruction cache misses
PAPI_L1_TCM  0x80000006  Yes   Yes  Level 1 cache misses
PAPI_L2_TCM  0x80000007  Yes   No   Level 2 cache misses
PAPI_L3_TCM  0x80000008  Yes   No   Level 3 cache misses
......
```

Online resources

IPM

https://github.com/nerscadmin/IPM

PAPI

http://icl.cs.utk.edu/papi/

Ploticus

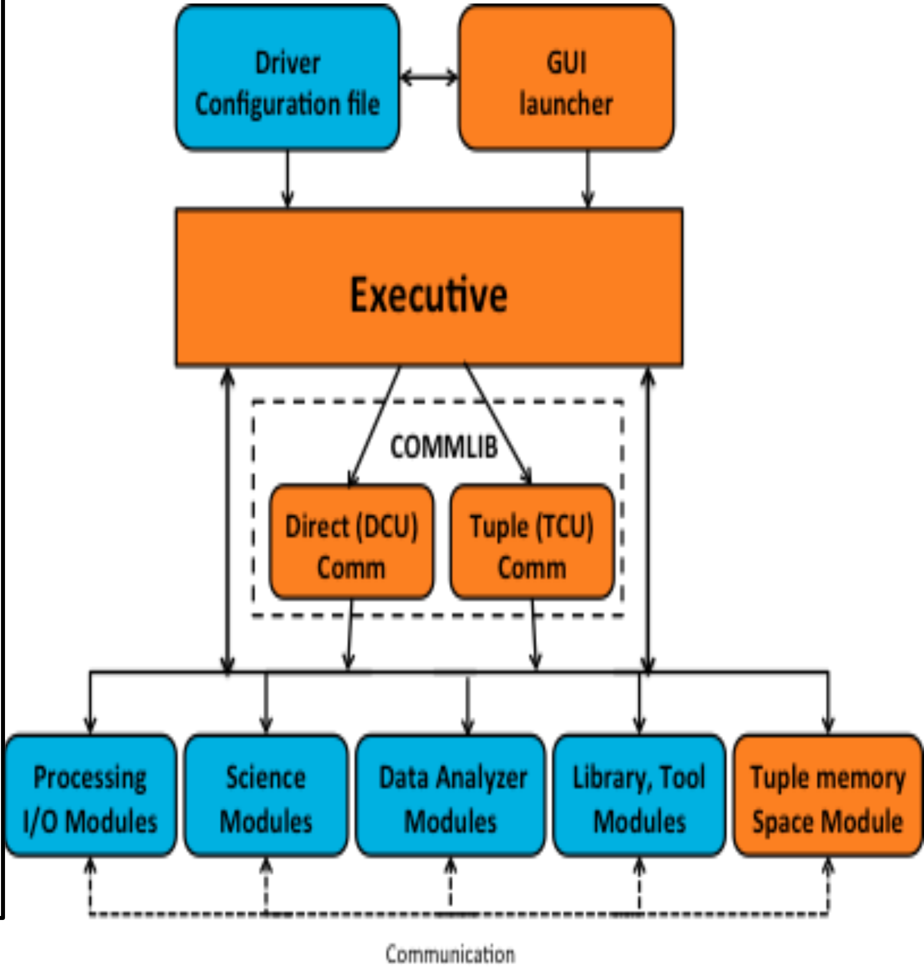
http://ploticus.sourceforge.net/doc/welcome.html

IPM usage

https://software.ecmwf.int/wiki/display/UDOC/How+to+use+IPM+to+profile+MPI
http://www.nersc.gov/users/software/performance-and-debugging-tools/ipm/
https://www.lrz.de/services/software/parallel/ipm/

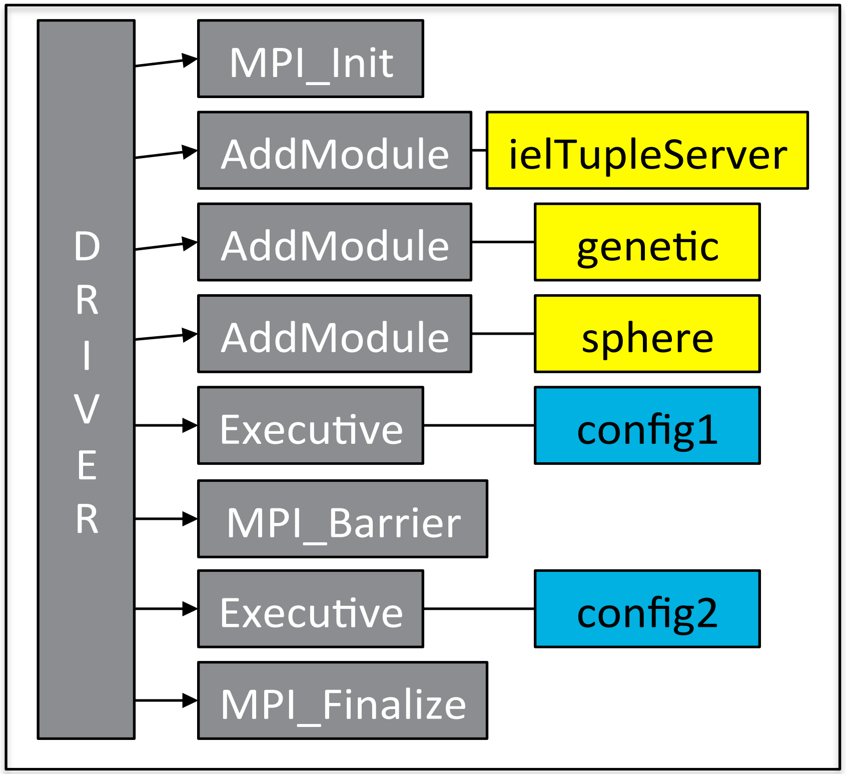
open Distributive Interoperable Executive Library (openDIEL) for Code Based Systems Simulations

- A workflow framework to facilitate system-wide inter-disciplinary simulations, specifically designed to run on large-scale HPC platforms.
- Encapsulate and run a collection of user's codes (serial or parallel) with a single MPI executable.
- Auto driver, opneDIEL-AM to run a collection of serialcodes seamlessly.
- A user admits a set of computer codes to openDIEL as function modules, define the workflow, and compute.
- Give elapse time of codes
- A Driver starts MPI_Init in MPI_COMM_WORLD communicator, adds function modules, launches IEL_Executive.
- IEL_Executive reads workflow configuration file, loads needed modules, splits MPI_COMM_WORLD, and manages execution components.



User Input Workflow - Configuration File

openDIEL-AM – Auto Executable



Input – workflow file

```
shared_bc_sizes = [];    tuple_space_size = 0;
modules=(
      { function="MODULE-0";       #← PREPARE DATA STEP (SHELL SCRIPT)
        args=("./datacopy.sh"); libtype="static";             #← SCRIPT NAME
        size=1; points=( (() );    );    #← USE PROCESS 0 TO DISTRIBUTE DATA
      },
      { function="MODULE-1";            #← PRIMARY SIMULATION (C CODE)
        args=("./peakdetector","sample.mzML");            # ← MAVEN CODE
        libtype="static";  splitdir="SAMPLE";   #← PARALLEL COMPUTATION
        size=64;  points=(  (());  ); ;           #← USE 64 PARALLEL PROCESSING
      },
      {  function="MODULE-2";            #← ANALYZE DATA STEP ( R SCRIPT)
         args=("./Rscript", "data-analysis.r"); libtype="static";  # ← R NAME
         size=1; points=( (() );    );      #← USE PROCESS 0 TO ANALYZE DATA
       },
      );
workflow: ;  #←WORKFLOW SEQUENCE OF A SINGLE GROUP COMPUATION
{  groups:  {  g1: {  order=("MODULE-0","MODULE-1","MODULE-2"  } } }
```

- ✓ An auto-driver runs sets of serial codes, users simply input workflow files
- ✓ Allow multiple copies in different I/O directories
- ✓ Group contains multiple modules with linear dependency
- ✓ Set contains multiple groups in DAG dependency
- ✓ Multiple sets run concurrently
- ✓ mpirun –np xx./openDIEL-AM workflow.cfg

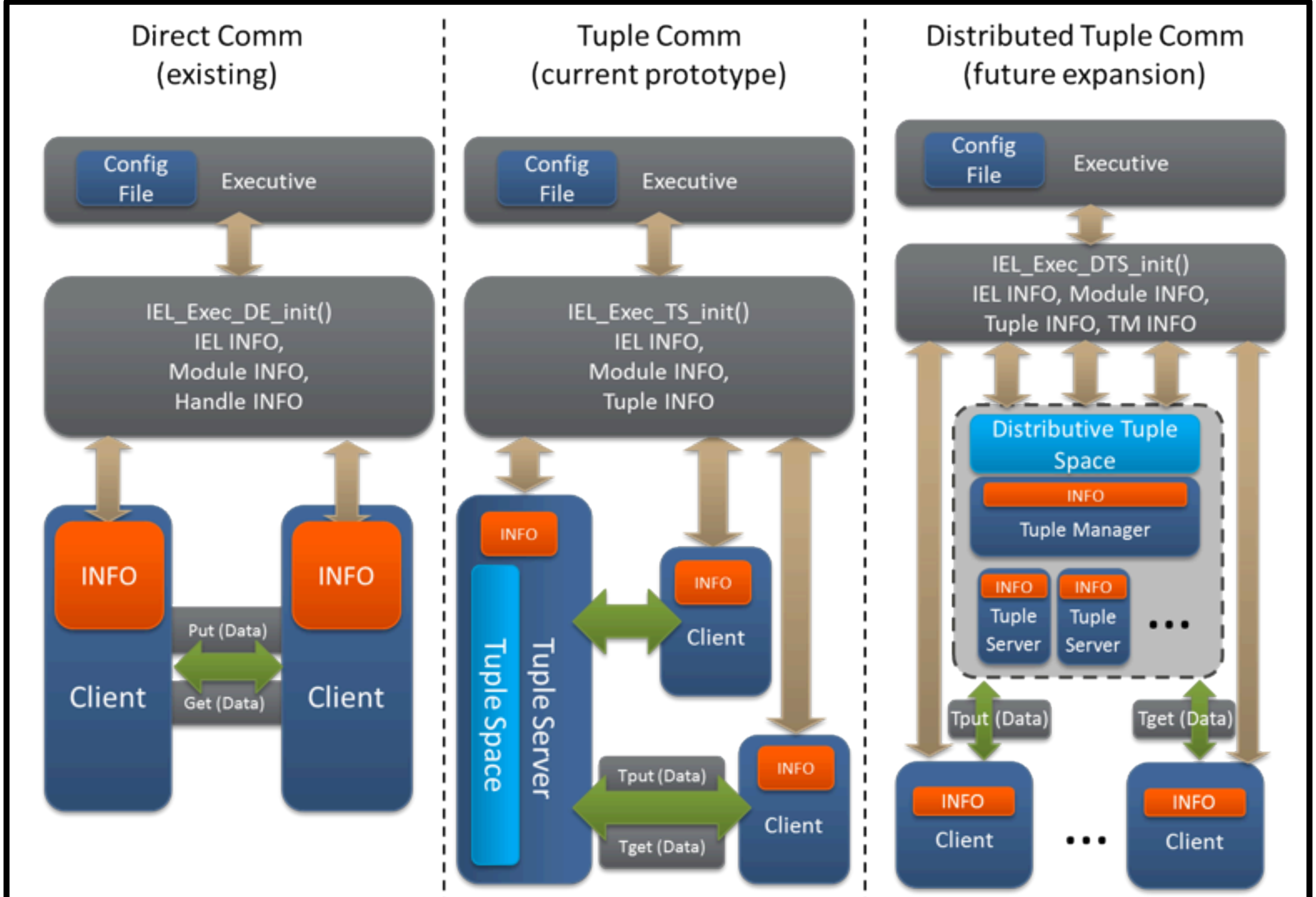
```
shared_bc_sizes = []
tuple_space_size=1
modules=(
  {
    function="ielTupleServer"
    args=()
    libtype="static"
    library="libIELexec.a"
    size=1
  },
  {
    function="MODULE-1"
    args=("../serialexecutable1")
    splitdir="ENERGY_ANALYSIS"
    copies=3
  },
  {
    function="MODULE-2"
    args=("../serialexecutable2")
    libtype="static"
    copies=1
  },
  {
    function="energy_analysis"
    args=()
    exec_mode="parallel"
    libtype="static"
    processes_per_copy=4
    copies=3
    size=12
    splitdir="ENERGY_OUTPUT"
  },
  {
    function="fluid_flow"
    args=()
    libtype="static"
    copies=3
    threads_per_process=4
    cores=12
    splitdir="ENERGY_OUTPUT"
  },
  {
    function="fluid_flow"
    args=()
    libtype="static"
    copies=3
    threads_per_process=4
    cores=12
    splitdir="FLUID_OUTPUT"
  },
```

```
  {
    function="output_analysis"
    args=()
    libtype="static"
    processes_per_copy=4
    copies=3
    size=12
    threads_per_process=3
    cores=36
    splitdir="ANALYSIS"
  }
)

workflow:
{
  tupleset:
  {
    tuplegroup:
    {
      order=("ielTupleServer")
      iterations=1
    }
  },
  set1:
  {
    group1:
    {
      order=("MODULE-1", "MODULE-2"
      iterations=2
    }
  },
  set2:
  {
    num_set_runs = 2

    group2:
    {
      order=("energy_analysis")
      iterations=1
    },
    group3:
    {
      order=("fluid_flow")
      iterations=1
    },
    group4:
    {
      order=("output_analysis")
      depends=("group2", "group3")
      iterations=1
    }
  }
}
```

Workflow Engine - openDIEL

- Open Distributive Interoperable Executive Library
- Combine a set of computer codes and run as a single executable
- Work for parallel codes
- module load openDIEL/V3-AM
- /sw/cs400_centos7.3_acfsoftware/openDIEL/V3-AM/centos7.3_intel17.2.174
- openDIEL-V3-AM/EXAMPLES: DEPEND-BASIC, HELLOWORLD
- Readme file, workflow.cfg, pbssub-acf, OUT

Communication Interfaces of openDIEL



- Synchronous
- MPI send/receive wrapper

- Asynchronous exchanges
- One-way communication

- Scalable asynchronous
- Many-to-many exchanges