Software stack development/support/int. collab. Break out session

Franck Cappello, Pete Beckman

IEPS Cologne 2011

Software stack development/support/int. collab.

- Look at the Exascale software stack from a per country perspective
- Gather the software components by country
  - In 3 categories
  - Type of development and support
- Gather the intentions of international collaboration
  - for 3 classes of international relations
- Gap Analysis

Stack components: Definitions

3 Categories:

- Cat 1 (a group commits to provide in production quality)
- Cat 2 (experimental):
- Cat 3 (no major funded project at this time: rely on others- vendors, other countries):

- Color code **for category 1**: developer provided support, collaboration with vendor for vendor support, release as opensouce.
- Underscored: Non open-source

International relations: Definitions

- Classes of international relations:
  - C1: Cooperation (loose sharing without common outcomes: discussions, workshops)
  - C2: Collaboration (produces common outcomes and are funded for that: roadmap: IESP, EESI ; standard, APIs, joint research software, working groups, G8,)
  - C3: Co-development (concrete deliverable: MUST, Score-P, Lustre)
  - C?: don't know yet

Stack components: Japan

- Cat 1 (commit to provide in production quality): programming language: C2: XMP, C2-C3: OS kernel (for many core architecture MIC, node level scheduling, including PGAS style), file system (C?:Luster, C2: GFARM), C2: communication libraries (below MPI)
- Cat 2 (experimental): C2: Fault tolerant libraries (FTI, fault detectors), C2: Domain specific languages, C2: numerical libraries having autotuning functions, C?: MPI implementations (flow control), C1: Power management and scheduling framework (scheduling strategies limiting peak power), C2: I/O middleware
- Cat 3 (no major funded project at this time: rely on others- vendors, other countries): Debugger, perf tools, perf tools, node level scheduling, Batch scheduler, scalable RAS system, deployment systems, Node level compilers

Color code for category 1: developer provided support, collaboration with vendor for vendor support, release as opensouce.

Stack components: Europe

- Cat 1 (commit to provide in production quality): C2:Programming models/ languages: MPI/OMPSs (C3:long run may move to OpenMP 4.0), C1:compiler for heterogeneous systems (HMPP); C?node level Runtime, C3:Tools (performance (Scalasca, Score-P, Paraver) including I/O, C1:performance (Vampir, ThreadSpotter) including I/O, C1:debuggers (DDT), C3:correctness (MUST), C1:correctness), C3:commit to contribute to MPI implementations (Open MPI), C3:file systems (Lustre), C3:numerical libraries, C?:numerical libraries (NAG), *OS kernel*
- Cat 2 (experimental): C1:communication libraries (below MPI), C2:Fault tolerant libraries (FTI, MPI), C?:Domain specific languages (for autotuning), C?:Batch scheduler, C?:deployment systems, commit to contribute to MPI implementations, C?:Power measurement management, C?:I/O middleware
- Cat 3 (no major funded project at this time: rely on others- vendors, other countries, have to buy for a specific Vendor): Node level compilers (vendor compilers), PGAS languages, scalable RAS system,

Color code for category 1: developer provided support, collaboration with vendor for vendor support, release as opensouce.

Stack components: USA

(We are still re-planning, so answers are mostly examples from ESC plan)

- Cat 1: programming models (C2:CAF, C?:UPC, C2:OpenMP,), C3:OS Kernel, C2: communication library (C3:MPICH, C3:OpenMPI), runtime (C2:Unistack/ Plum), performance tool (C3:PAPI, C3: TAU), C1: I/O delegation, I/O middleware (C1:IOFSL), Visualization (C3: VTK, C3:Visit), Numerical libraries (C3:MAGMA/PLASMA, C3: PETSc, C1:Trilinos, C?:SuperLU, C?:Hypre), C2/ C3: I/O libraries (MPI-IO, HDF5, pNetCDF), resource manager (C?:SLURM)

- Cat 2: performance tool (C3OpenSpeedShop, C3mpiP, C?HPCToolkit), fault tolerance backplane (C2:CIFTS), tool infrastructure (C?:MRNet), compiler framework (C?: Rose), debugging and validation (C?:Memcheck,C?: STAT, C3:MUST), C3:PowerMgmt Layer, low level threading lib. (C1?: Qthreads), Task scheduler (ExM), File system (PVFS, PLFS), fault tolerance environment (C3:SCR), data analytics libraries (C1: Xanalytics), Data model storage library (C2: Damsel), Composition frameworks (C2:COMPOSE-HPC), New programming models and approaches (C3?:Sketch, Domain specific languages?), benchmarks and mini-apps (C3:SHOC)

- Cat 3 Debuggers, commercial compilers, etc.

Color code for category 1: developer provided support,collaboration with vendor for vendor support, release as opensouce.

Stack components: China

- Cat 1 (commit to provide in production quality): C1: kylin os (derived from freeBSD, compatible with Linux:same exec format), deployment tool (os+lustre-custom), NR-MPI, HPC virtual environment
- Cat 2 (experimental): C3:resilience computing framework, C3:hybrid tiered file system, C2:heterogeneous programming runtime system (load balance &pipeline), C2:application programming framework, C3: autonomic resource management and scheduler
- Cat 3 (no major funded project at this time: rely on others- vendors, other countries, have to buy for a specific Vendor): compilers for commercial processors

Color code for category 1: developer provided support, collaboration with vendor for vendor support, release as is (without support).

Stack components: Russia

- Cat 1 (commit to provide in production quality):
- Cat 2 (experimental):
- Cat 3 (no major funded project at this time: rely on others- vendors, other countries, have to buy for a specific Vendor):

Color code for category 1: developer provided support, collaboration with vendor for vendor support, release as opensouce.

Early Results of the GAP analysis

- GAP analysis:
  - Do we cover all the software stack in CAT 1?
    - It is true for all hardware (cannot tell now)
- What seems missing in CAT1:
  - but usually provided by vendors:
    - Low level compilers
    - RAS system, system management,
    - Batch scheduler?
  - Limited power management / fault tolerance (MPI)