Mitglied der Helmholtz-Gemeinschaft

System Software Stacks Survey

Bernd Mohr

Jülich Supercomputing Centre

Potential System Architecture Targets

System attributes	2010	"2015"		"2018"		Difference 2010 & 2018
System peak	2 Pflop/s	200 Pflop/s		1 Eflop/sec		O(1000)
Power	6 MW	15 MW		~20 MW		
System memory	0.3 PB	5 PB		32-64 PB		O(100)
Node performance	125 GF	0.5 TF	7 TF	1 TF	10 TF	O(10) – O(100)
Node memory BW	25 GB/s	0.1 TB/sec	1 TB/sec	0.4 TB/sec	4 TB/sec	O(100)
Node concurrency	12	O(100)	O(1,000)	O(1,000)	O(10,000)	O(100) – O(1000)
Total Concurrency	225,000	$O(10^8)$		$O(10^9)$		O(10,000)
Total Node Interconnect BW	1.5 GB/s	20 GB/sec		200 GB/sec		O(100)
MTTI	days	O(1day)		O(1 day)		- O(10)

Key Issues with Existing Stack

- Scalability
- Fault-tolerant / fault-aware components
- Power-saving / Power-saving-aware components
- Heterogeneity of HW and SW components
- I/O and memory
- Strong resistance in user community to revolutionary approaches

Roadmap Components

4.1 Systems Software

- 4.1.1 Operating systems
- 4.1.2 Runtime Systems
- 4.1.3 I/O systems
- 4.1.4 Systems Management
- 4.1.5 External Environments

4.2 Development Environments

- 4.2.1 Programming Models
- 4.2.2 Frameworks
- 4.2.3 Compilers
- 4.2.4 Numerical Libraries
- 4.2.5 Debugging Tools

4.3 Applications

- 4.3.1 Application Element: Algorithms
- 4.3.2 Application Support: Data Analysis and Visualization
- 4.3.3 Application Support: Scientific Data Management

4.4 Crosscutting Dimensions

- 4.4.1 Resilience
- 4.4.2 Power Management
- 4.4.3 Performance Optimization
- 4.4.4 Programmability

THE INTERNATIONAL JOURNAL OF
HIGH PERFORMANCE COMPUTING APPLICATIONS
Special Issue on Tools in the ACTS Collection
VOLUME 21 NUMBER 1 SPRING 2007
SAGE Publications
ISSN 1094-3420
http://hpc.sagepub.com

see IJHPCA, Feb 2011, http://hpc.sagepub.com/content/25/1/3

The Survey

- System Software Stack Survey sent out to 28 HPC centers (78% response rate):

USA	Asia/Pacific	EUROPE	
ALCF	ITC Tokyo	BSC	CINECA
LLNL	NSCC-TJ	CSCS	EPCC
NERSC	RIKEN	GENCI/IDRIS	GENCI/CINES
OLCF	SCCAS	GENCI/CCRT+TGCC	HLRS
SNL		JSC	MSU
	CSIRO	NCF	RZG MPI
UIUC/NCSA	Tsukuba		
LANL	TiTech	CSC	LRZ

Disclaimer

- Sites selection by Jack and me
- Wide range of responses
  - Listed only most critical / most used items
  - Regarded example provided as multiple-choice list
  - Listed every single piece of software installed on site
- Survey only covers **evolutionary** part of IESP roadmap!
- Comments/evaluations/analysis is my personal opinion
  - not of Jülich, EESI, IESP, … ;-)

Roadmap Components

Covered in Survey

4.1 Systems Software

4.1.1 Operating systems
4.1.2 Runtime Systems
4.1.3 I/O systems
4.1.4 Systems Management
4.1.5 External Environments

4.2 Development Environments

4.2.1 Programming Models
4.2.2 Frameworks
4.2.3 Compilers
4.2.4 Numerical Libraries
4.2.5 ~~Debugging~~ Tools

4.3 Applications

4.3.1 Application Element: Algorithms
4.3.2 Application Support: Data Analysis and Visualization
4.3.3 Application Support: Scientific Data Management

4.4 Crosscutting Dimensions

4.4.1 Resilience
4.4.2 Power Management
4.4.3 Performance Optimization
4.4.4 Programmability

Survey: Operating Systems

	USA	Asia/Pacific	Europe	TOTAL
AIX			4	4
Linux	10	6	16	32
- SLES	5	2	7	14
- RHEL	4	3	4	11
- other	1	1	5	7
LightWeightKernel	4	1	3	8
Other		1	1	2

> 90%

> 66%

> 50%

- Other: Solaris, NEC UX
- Trend to Linux and/or LightWeightKernel
  - SLES before RHEL?
- Do we need HPC/Exascale-Linux?
  - How open is Linux community to changes needed for HPC?

Roadmap: Operating Systems

critical

- Category I: Uniquely Exascale
  - Define the base OS (Standard API)
  - APIs for resilience (access to RAS, etc)
  - System wide power management i.e., power aware job scheduling
  - Collective OS operations

- Category II: Exascale plus trickle down
  - Scalable system simulation environment
  - Improved APIs for scalable performance monitoring and debugging
  - New APIs for energy management

- Category III: Primarily Sub-exascale
  - Improved APIs for explicit memory management
  - Improved APIs for threading (⇨ many-core)

Survey: I/O: File systems

	USA	Asia/Pacific	Europe	TOTAL
GPFS	3		6	9
Lustre	4	3	6	13
NFS	3	4	10	17
PANFS	1		2	3
Other	1	4	3	8

> 90%

> 66%

> 50%

- Other: HDFS, CXFS, ZFS, ADIC SNFS, SRFS, SAM-QFS, PVFS1
- Future of Lustre development?
  - multiple Lustre "support groups" exist in US and EU
- GPFS for non-IBM systems?

Survey: I/O: Libraries

	USA	Asia/Pacific	Europe	TOTAL
netCDF	5	2	11	18
parallel netCDF	4	1	9	14
HDF5	5	2	11	18
Parallel HDF5	3	0	1	4
MPI-IO	5	3	10	18

- Other: SILO
- Clear outcome: netCDF, HDF5, MPI-IO (85%)

> 90%

> 66%

> 50%

Roadmap: I/O

critical

- Category I: Uniquely Exascale
  - Customization within I/O, purpose-driven I/O
  - New I/O models, software, runtime systems and libraries
  - Intelligent/proactive caching mechanisms for I/O
  - Fault-tolerance mechanisms
- Category II: Exascale plus trickle down
  - Balanced architectures with newer devices
  - File Systems or alternative mechanisms
  - Active Storage
  - Wide-Area I/O and integration of external Storage Systems
  - Special purpose network protocols for parallelism
  - I/O into Programming Models and Languages
- Category III: Primarily Sub-exascale
  - Balanced architectures with newer devices embedded within nodes

Survey: Batch Systems

	USA	Asia/Pacific	Europe	TOTAL
LoadLeveler	0	0	6	6
PBS pro	0	1	3	4
Torque	3	1	5	9
MOAB	3	1	5	9
ALPS	3	1	2	6
SLURM	2	2	3	7
LSF	0	1	1	2
NQS	0	1	1	2

> 90%

> 66%

> 50%

- Other: MAUI, CLEO, Condor, Grid Engine, OAR, Cobalt
- Most mentioned overall: Torque/MAUI (42%)
- LoadLeveler strong in Europe (50%)

Survey: Programming Models

	USA	Asia/Pacific	Europe	TOTAL
MPI	14	7	33	54
- MPICH	5	2	6	13
- MVAPICH	3	0	3	6
- Open MPI	4	2	8	14
- Other / NA	2	3	16	21
OpenMP	4	4	9	17
Pthreads	4	3	9	16
SHMEM	2	2	5	9
GlobalArrays	3	1	5	9
ARMCI	3	1	4	8
StarSs (SMPSs)			4	4

- Other MPI: POE, Intel, HP, BlueGene, Parastation, Cray, Bull, SGI, Fujutsu, PMPISX
- Other: PVM, LAPI, BSP, MC#, DAPL, TBB, DMAPP

> 90%

> 66%

> 50%

Survey: Programming Languages

	USA	Asia/Pacific	Europe	TOTAL
C/C++	5	4	12	21
F77	5	4	10	19
F90/95	5	4	12	21
Python	5	4	8	17
CAF	2	2	5	9
UPC	4	2	5	11
Java	4	4	6	14

- Other: F2008, X10, Ox, Charm++, Chapel
- Obviously, different interpretations
  - installed / heavily used / wish list
  - used at site / used for HPC

> 90%

> 66%

> 50%

Roadmap: Programming Models

critical

- Category I: Uniquely Exascale
  - Exascale programming model
  - Scalable, fault-tolerant MPI
  - Application development tools

- Category II: Exascale plus trickle down
  - Heterogeneous node programming model
  - Domain-specific programming models
  - Language features for massively parallel I/O
  - Language support for adaptive computation

- Category III: Primarily Sub-exascale
  - Interoperability between models

Runtime Systems

critical

- Category I: Uniquely Exascale
  - Load balance (including tolerance to noise and temporary shortage of resources (i.e. as a result of faults))
  - Hierarchical execution models and scheduling
  - Scale/optimize Communications: MPI, routing, comm. schedule, ...
- Category II: Exascale plus trickle down
  - Asynchrony, overlap
  - Memory management & Locality scheduling
  - Heterogeneity: scheduling
- Category III: Primarily Sub-exascale
  - Fine grain mechanisms @ node level (for thread management & synchronisation support)

Survey: Compiler

	USA	Asia/Pacific	Europe	TOTAL
GNU	5	4	12	21
Intel	4	4	10	18
IBM	1	1	8	10
PGI	4	3	7	14
Pathscale	3	1	4	8
Cray	3	1	3	7

- Other: Oracle, LLVM, Fujitsu, CGG?

> 90%

> 66%

> 50%

- Need to take other compilers than GNU into account:
  - At least Intel, PGI, IBM
  - Issue for GNU/Linux build tools + basic software
    - especially C++ libraries (e.g. Qt)

Surrvey: Accelerator Support

	USA	Asia/Pacific	Europe	TOTAL
CUDA	5	4	9	18
OpenCL	4	3	7	14
HMPP		1	3	4
PGI		1	2	3

> 90%

> 66%

> 50%

- Other: ClearSpeed SDK, Cn, CXSL, PyCUDA
- Higher-level approaches (e.g. HMPP or PGI) are urgently needed?
  - OpenMP 4?

Roadmap: Compilers

critical

- Category I: Uniquely Exascale
  - Implement exascale language(s)
  - Support for resilience
- Category II: Exascale plus trickle down
  - Implement heterogeneous programming model
  - Support for massive I/O
  - New optimization frameworks (Locality, parallel program analyses, architecture-aware optimizations, Power)
  - Interactions between compilers and tools, runtime
- Category III: Primarily Sub-exascale
  - Implement enhancements to existing languages / APIs
    - MPI awareness in compilers, Interoperability
  - Automatic parallelization
  - Dynamic (re)compilation, feedback optimizations, autotuning
  - Refactoring tools

Survey: Numerical Libraries

	USA	Asia/Pacific	Europe	TOTAL
BLAS	9	7	24	40
ACML	3	3	5	11
FFTPACK	3	2	3	8
FFTW	5	3	10	18
hypre	3	1	4	8
LAPACK	5	4	11	20
libSci	3	2	4	9
ParMETIS	3	1	7	11
PETSc	5	3	11	19
ScaLAPACK	5	4	10	19
SPRNG	3	1	6	10
SuperLU	5	3	6	14
Trilinos	3	1	4	8

- BLAS := ATLAS + EESL + MKL + GOTO
- Many many others!

> 90%   > 66%   > 50%

Roadmap: Numerical Libraries

critical

- Category I: Uniquely Exascale
  - Fault oblivious, Error tolerant software
  - Smart (AI based) algorithms
- Category II: Exascale plus trickle down
  - Async methods
    - Overlap data and computation
  - Algorithms that minimize communications
  - Self-adapting
- Category III: Primarily Sub-exascale
  - Autotuning based software
  - Standardization activities
  - Architectural aware algorithms/libraries
  - Energy efficient algorithms
  - Mixed arithmetic
  - Hybrid and hierarchical based algorithms (e.g. linear algebra split across multi-core and GPU)

Survey: Debugger

	USA	Asia/Pacific	Europe	TOTAL
Totalview	5	3	10	18
DDT	4	1	6	11
Marmot	1		3	4
Intel Threadchecker	1		2	3
STAT	2	2	0	4

> 90%

> 66%

> 50%

- Other: Umpire
- Currently dominated by commercial offerings?
  - Interactions with/interfaces for open-source components
    - e.g. validation or performance tools

Roadmap: Debugger

critical

- Category I: Uniquely Exascale
  - Scalability of debugger methodologies (data volumes and APIs)
  - Debugging under frequent failover
  - Focus on multi-level debugging, communicating details of faults between software layers
  - Synthesis of fault information and understanding in the context of application and architecture
- Category II: Exascale plus trickle down
  - Specialized lightweight OS's
  - Automatic triggers, compile time bridge to debugger removing need to rerun
  - Scalable clustering of application process states and contexts
    - Filter/search within debugger
  - Vertical integration of debug and performance information across software layers
- Category III: Primarily Sub-exascale
  - Excision of buggy code snippets to run at lower concurrencies
  - Heterogeneity

Survey: Performance Tools

	USA	Asia/Pacific	Europe	TOTAL
CrayPat/App2	3	1	3	7
gprof	3	4	11	18
mpiP	3	1	6	10
OSS	4	1	2	7
Scalasca	1	1	11	13
TAU	5	2	5	12
Valgrind	5	3	8	16
Vampirtrace	3	2	5	10

> 90%

> 66%

> 50%

- Other (>1): ThreadSpotter, FPMPI2, HPCToolkit, IBM IHPCT, IPM, ITAC, jumpshot, memP, Paraver, STAT
- Potential confusion: Intel or TUD Vampirtrace?!

Performance Tools Base Components

	USA	Asia/Pacific	Europe	TOTAL
dyninst	3	1	3	7
OTF	3	1	4	8
PAPI	5	3	9	17

> 90%

> 66%

> 50%

- Other (>1):PDToolkit
- Future candidate: Score-P
  - European cross-tool instrumentation and measurement infrastructure

Roadmap: Performance

critical

- Category I: Uniquely Exascale
  - Extremely-scalable performance methods and tools (online reduction and filtering, clustering), analysis (clustering, data mining), and visualization (hierarchical) ⇨ Handle billions of components
  - Performance measurement and modeling in presence of noise / faults / power adaption related changes

- Category II: Exascale plus trickle down
  - Automated / automatic diagnosis / autotuning
  - Vertical integration across SW layers (app, middleware, runtime, OS)
  - Performance-aware design and implementation
  - Performance optimization for other metrics than time (e.g. power)

- Category III: Primarily Sub-exascale
  - Support for heterogeneous hardware and hybrid programming models including analysis and modeling of asynchronous tasks

Survey: Scripting and Building

	USA	Asia/Pacific	Europe	TOTAL
sh/bash	5	4	12	21
Perl	5	4	12	21
Python	5	4	12	21
Tcl/TK	5	4	9	18
make	5	4	12	21
cmake	3	1	6	10
configure / autoconf	5	4	8	17

- Other: ruby, ant, mercurial
- sh, Perl, Python, make only items besides GNU compiler with 100% result
- cmake/autotools: "weak" support for HPC issues (Fortran, cross-compiling, …)

> 90%

> 66%

> 50%

Survey: Data Analysis and Visualization

	USA	Asia/Pacific	Europe	TOTAL
Ensight	3	1	4	8
gnuplot	3		2	5
IDL	3	1	5	9
Matlab	3	1	5	9
NCAR	3	1	5	9
OpenGL	4	2	7	13
ParaView	4	2	7	13
VisIt	4	3	6	13
VTK	4	2	8	14

- Other (> 1): AVS, COVISE, Ferret, GDL, gimp, mathematica, ncview, parallel R, R, VMD

> 90%

> 66%

> 50%

Survey: Not enough / No responses

- Runtime Systems
  - ZeptOS (3)
- I/O: archiving
  - HPSS (4)
- System Management
- External Environments
  - DEISA (3), PowerMan (2), FreeIPMI (2), Conman (2)
- Workflow Tools
  - UNICORE (3), Globus (3), bbcp (4), Kepler (2)
- Scientific Data Management
  - iRODS (3), HOPPER (2)

Roadmap: Frameworks

critical

- Category I: Uniquely Exascale
  - Resilience API and Utilities

- Category II: Exascale plus trickle down
  - Multi-institutional/multi-project collaboration plan
  - Tool chain development/selection
  - Programming model evaluation/adoption
  - Data placement
  - Multi-component simulation utilities
  - Software libraries access

Roadmap: Scientific Data Management

critical

- Category I: Uniquely Exascale
  - Scalable Data Analysis and Mining Software and Tools
  - Scalable Data Format and High-level Libraries

- Category II: Exascale plus trickle down
  - Scientific WorkFlow Tools
  - Search and Query Tools
  - Wide-Area data access, movement and query tools
  - Scientific Databases

Other issues

- What system is needed for successful development for Exascale? (for which tasks?)
  - Smaller dedicated system in 2015?
  - Production 100PF system in 2015?
  - Dedicated 100PF system in 2015?
- Interactions between open-source ⬄ commercial/vendor components
  - especially if NDAs are required?
- Maintenance, support, documentation, training for open-source components
- Reliance on open-source base software (Linux kernel, compiler, build tools, base libraries) ⬄ HPC specific support (non-GNU compiler, Fortran, cross-compilation, microkernels, …)