Summary Report: IESP Workshop 2

Paris, France
June 28-29, 2009

Background

The initial focus of the International Exascale Software Project (IESP) is on staging a series of three international meetings, one each in the United States, Europe and Asia beginning in the spring 2009. Information about all these meetings can be found at the project website, www.exascale.org. This is a summary report of activities of the second IESP workshop, which was held in Paris France on June 28 and 29, 2009. Attendees included members from industry, academia, and government, with expertise in a range of critical areas. This was a follow up to the workshop that was held in Sante Fe, New Mexico, USA, on April 7 and 8, 2009.

The overarching goal of this series of meetings is to create a plan for a coordinated international effort to create a common software cyberinfrastructure capable of meeting the architectural challenges of current and future peta/exascale systems and the application challenges of the visionary research agendas they must serve. The objectives this plan should incorporate, and which therefore need be addressed at these meetings, include the following:

- *Provide a framework for organizing the software research community*: The IESP will articulate an organizational framework designed to enable the international software research community to work together to deliver more capable and productive HPC systems. The framework will include elements such as initial working groups, outlines of a system of governance, alternative models for shared software development with common code repositories, feasible schemes for selecting valuable software research and incentivizing its translation into usable, production-quality software for application developers, etc. This organization must also foster and help coordinate R&D efforts to address the emerging needs of users and application communities on new platforms, such as Jaguar and Blue Waters in the US, JUGENE in Germany, and the next generation RIKEN system in Japan.
- *Create a thorough assessment of needs, issues and strategies*: As part of its planning process, the IESP will assess the short-term, medium-term and long-term needs of applications for peta/exascale systems. Participation in the IESP from representative application communities and vendors will help ensure the adequacy of these assessments. The work of the organization that emerges from the IESP must be prepared to provide the key research-oriented funding agencies around the world with a series of well-crafted reports on the critical technical issues of peta/exascale software infrastructure and with alternative strategies, both technical and programmatic, for solving them.
- *Initiate coordinated software roadmap*: Working with the results of its application needs assessment; the IESP will initiate the development of a coordinated roadmap to guide open source HPC software development with

better coordination and fewer missing components. This roadmap will help to guide both cooperative development and joint research efforts.

- *Encourage and facilitate collaboration in education and training*: The magnitude of the changes in programming models and software infrastructure and tools brought about by the transition to peta/exascale architectures will produce tremendous challenges in the area of education and training. The IESP plan will therefore provide for cooperation in the production of education and training materials to be used in curricula, at workshops and on-line.
- *Engage and coordinate vendor community in crosscutting efforts*: To leverage resources and create a more capable software infrastructure for supporting exascale science, the IESP will engage and coordinate with vendors across all of its other objectives. Vendor participation in and contributions to all of these objectives — comprehensive application needs assessment, well-ordered but adaptive software roadmap, organized framework for cooperation, coordinated R&D programs for new exascale software technologies — will be encouraged and facilitated.

Attendance to these workshops was by invitation only. The Paris workshop had 65 people in attendance.

Place of the second workshop in the planning process

The main goal of the Paris meeting was to *initiate development* of a coordinated software roadmap and *get the participants fully engaged* with the process of developing a draft version. Since the roadmap is intended to be a living document, used to guide and coordinate the work of the international open source software community over time, it is intended to serve as the vehicle through which progress on the other objectives in the plan are expressed. For example, the most fundamental set of inputs into the roadmap are the assessments of the short, medium and long term needs, issues, and strategies emerging from an analysis of emerging peta/exascale architectures and the scientific research agendas we aspire to serve. Consequently the majority of the effort at the second workshop was concentrated on those analyses and assessments, which are described in more detail below. On the other hand, the other objectives of the plan can be thought of as a derived product of the roadmap, once it is created. These objectives include the following:

- Provide a framework for organizing the software research community
- Encourage and facilitate collaboration in education and training
- Engage and coordinate vendor community in crosscutting efforts

On this analysis, therefore, the development of the IESP roadmap is essentially the pivot point for the overall planning effort.

The ultimate goal of the entire community effort – *the development of a common, high quality computational environment for peta/exascale systems to serve the entire research community* – will be realized through the successful execution of the plan in all its elements. The discussion of strategies for catalyzing, coordinating, and sustaining the effort of the international open source software community to create that environment as quickly as possible began at this workshop, but could not be fully engaged. It seems clear that this part of the plan can be made concrete only after the roadmap has become more clear and complete in its direction and details. Therefore, the workshop concentrated on that goal.

Summary of structure and content of workshop activities

After plenary sessions to establish common context, the participants at the workshop divided into four breakout groups –software, applications, vendors (industrial collaborators), and funders. The division was intended to facilitate the analysis and assessment of the underlying needs, issues and strategies that must feed into the roadmap. The software group, which was the largest, found it natural to divide again in order to attend to intra-processor and inter-processor issues separately; the results of their discussions were then consolidated at the end of the workshop.

In a general sense, these four breakout groups identify the key structural components of the roadmap. Each of the breakouts groups prepared more detailed reports of the work, and these accounts are included below. But their separate results may be summarized here as follows:

- *Software subgroup* – In a very real sense, the software part of the IESP roadmap forms the center of the effort. The results of the work of the software subgroup during and following the second meeting reflect this fact. This includes not only a detailed inventory of the wide range of issues that were raised, discussed and assessed at the meeting, but more importantly it includes an analysis of the goals, assumptions, expected benefits and basic requirements for the roadmap itself. The former are in the notes and the summary from the breakout sessions, which were intellectually strenuous group efforts, condensed into two days, to survey all the critical issues in this area so far as they can be seen today. In each main category the participants described the current situation, the state of the art, the actions that we can see need to be taken to reach the next level, and the research paths that remain to be explored to make the necessary discoveries that we cannot yet see. The latter are included in the initial draft of the software roadmap, which focuses on goals, assumptions, priorities and requirements that the software roadmap should meet.
- *Application subgroup* – The application subgroup divided its roadmap problem into three parts: selecting representative application domains that tie research to platforms available over the next ~10 years; detailing the software challenges that are likely to be raised by effort to scale up to these problems; and establishing a list of experts from these domains who can provide further insight, analysis and validation of roadmap issues. Their report provides some details on all three items. The groups charge for the third workshop was to contact the experts listed and get their input on the computational and software infrastructure issues that the group identified. To gather input for planning a community mobilization strategy, the experts will also be asked to identify the impact of the last machine change they had to face on their applications (porting, optimization, re-writing, etc.)

and on their approach to doing simulation, and the expected impact of the next machine change (going from Tflops to Pflops, Pflop to Eflops).

- *Funding subgroup*: If international cooperation in creation of tomorrow software cyberinfrastructure is to succeed, the national funding agencies for HPC software must work together. The results of the funding breakout sessions, both in the slides and in post-meeting summary, make it clear that they are motivated to do so. These agencies recognize that the combined urgency and scope of the problem necessitate a coordinated, global response. But such a response will require in turn that an exascale software integration methodology and strategy be developed. The software roadmap will play a key role in that effort, since it will be essential for funding agencies to plan their programs and to define potential major initiatives. At a minimum, those requirements must be expressed in the roadmap by the classification of software topics into those that are mainly implementation issues, those that require a stopgap solution, and those in which a completely new approach is needed. Helping these agencies succeed for the community also means that the societal benefits of exascale, as well science and engineering case, need to be clarified and further developed.
- *Vender/Industrial collaborators* – The discussions of community collaboration with vendor or industrial partners focused on potential mechanisms and models that could facilitate such collaboration in a mutually productive and sustainable way. The analysis this group was able to provide is detailed, wide-ranging and penetrating. Leveraging lessons derived from experience with other large-scale science and engineering projects (e.g. big instruments), it draws out the practical implications of hardware and software on high-performance computing markets and reviews plausible models for industry/government/community collaboration.

In general, the second workshop provided the framework that the project needs to pursue in going forward. More specifically, the key components of roadmap have been identified and their content has begun to be filled in. An initial group of authors who will lead the effort to take the roadmap components forward has been identified, though it is clear many volunteers will be needed to help finish this work in or for the project to stay on schedule. In preparation for the meeting in Japan in October, review of the workshop results continues in order to ensure that cross cutting issues and possible gaps are being identified and covered.

Technical challenges and needs of academic and industrial software infrastructure research and development

Breakout Group 1

IESP, Saclay, June 28 & 29, 2009

Participants: Pete Beckman, ANL, US; Franck Cappello, INRIA, France; Barbara Chapman, U of Houston, US; Alok Choudhary, NWU, US; Jack Dongarra, U of Tennessee, US; Sudip Dosanjh, SNL, US; Luc Giraud, CERFACS, France; Bill Gropp, UIUC, US; Mike Herioux, Sandia, US; Yutaka Ishikawa, U of Tokyo, Japan; Jesus Labarta, BSC, Spain; Bob Lucas, ISI, US; Barney MacCabe, ORNL, US; Satoshi Matsuoka, TiTech, Japan; Bernd Mohr, Juelich, Germany; Hiroshi Nakashima, U of Kyoto, Japan; Mitsuhisa Sato, U of Tsukuba, Japan; John Shalf, LBNL, US; David Skinner, LBNL, US; Thomas Sterling, LSU, US; Anne Trefethen, Oxford, UK; Mateo Valero, BSC, Spain; Jeffery Vetter, ORNL, US; Vladimir Voevodin, Moscow State U, Russia; Kathy Yelick, LBNL, US;

Preliminaries

Prior to discussing research topics and issues, we tried to reach some consensus on architectural trends needed to frame the discussion. Among the important assumptions were the following: (i) We will have an exascale system in 2018-2020. (ii) Performance will come from $O(1\text{B})$ way parallelism regardless of architecture. (iii) Socket-wide cache coherence will not be feasible. (iv) The memory bandwidth "wall" will be solved. (v) Although SSD will improve I/O performance, fundamental I/O performance issues will still exist.

During the discussion of assumptions, there was a frequent concern raised of whether or not our assumptions were constraining in the potential exascale solutions. In particular, there was discussion about whether or not we need a re-evaluation of our model of computation.

After the initial discussion of assumptions, the software infrastructure group split into two subgroups, one focused on node-level issues, the other on cross-node issues. In most cases, these issues were discussed in the framework of what the current situation is, the state-of-the-art, the needed actions and a list of roadmap and research topics.

Intra-Node Discussions

The intra-node group identified the following issues as important research topics:
1-billion way concurrency and load balancing: Exascale systems will have $O(1\text{B})$ cores, or a similar measure of computational units. There will be hierarchy in both algorithms and platforms. Present use of concurrency assumes a relatively flat processor layout, static compile or submission time specification of partitioning and scheduling, based on preconceived knowledge of machine and problem. Effective exascale applications and programming models will need to expose fine grain, dynamic parallelism for several times the number computational units in order to achieve adequate parallelism and effective latency hiding and load balancing. We will need to express flow control and synchronization.
Locality and distributed data: Locality of reference and some kind of global view of data objects are already important for large-scale parallel applications, and will become even more important. Remote memory access will become even more expensive and complex cache hierarchies will be difficult to manage. Presently, programming models have a variety of ways of expressing scope and locality, but there is no standard, mature environment that is suitable for exascale. Programming models need to support expression of scope and locality at both algorithmic and application code levels, especially for global view programming. We need a telescoping approach to optimize data placement and motion, from automatic to selectively user controlled.
Sustainability and interoperability: Our community has a large software base that not only encodes published numerical algorithms for science and engineering problems, but also thousands of heuristics gleaned from the use of these codes in solving challenging problems. New programming models must, to the extent possible, protect the large investment in existing software, and provide incremental migration paths. In addition to compilers, a full tool-chain from debuggers to performance analyzers will be important. Our community must also be aware of industry efforts in programming models and leverage them where possible, influencing design where our experience can enhance the quality of the industry-standard solutions.
Operating systems: Current operating systems were designed for single processor or SMP nodes and do not handle heterogeneous hardware or unconventional memory structures well. Linux variants are most common and assume homogeneous shared memory system. Operating systems will need to work on heterogeneous hardware and with nonconventional memory structures. They will need to provide applications and runtime environments more control over scheduling and resources. In some instances we need to remove the OS from the critical path to resource access, granting protected "bare-metal" access. OS's will need global namespace management, power management, scalable mechanisms for fault isolation, protection and information propagation to applications and runtime environments.
Algorithms & Software Libraries: Algorithm and library development has largely been done to target networked collections of serial processors, although a legacy of SMP work is certainly available for reconsideration. Exascale systems will have

complicated node architectures that will require new algorithms. We need to develop algorithms that are fault oblivious and error tolerant, that work on hybrid and hierarchical system, use mixed precision where possible, are energy efficient and minimize communication, especially at synchronization points. We need continued work in optimal complexity algorithms. Software libraries will be the early adopters of new programming models, will support autotuning, be architecture-aware, support simultaneous execution on heterogeneous nodes and support error propagation and sensitivity results.

Performance: Presently, performance is not a high priority compared to functionality and correctness, and is too manual and labor intensive, lacking performance models. We need performance-aware design, development and deployment, integrated with compilers and runtime systems. We need support for performance observability in hardware and in the runtime system. We need support for heterogeneous hardware and mixed programming models.

Fault tolerance: System faults will increase, probably dramatically, as we approach exascale, to the point where checkpoint restart will not be feasible. Errors will occur in hardware, software and memory. Soft errors will increase. Presently we assume that our environment will be fault-free and if not we simply restart from the latest checkpoint. Going forward, we need to distinguish between isolated failures and full system interrupts, a major challenge. We also need programming models to support fault detection including soft errors, and recovery including features such as transaction support, retry. We also need to explore features that selectively increase reliability through software means.

Inter-node Discussion

The inter-node discussion group first identified a number of research topics, and attempted to categorize their importance with respect to timeline leading up to exascale in 2018-2020 timeframe. These were: (1) those that are problematic already at petascale, and require immediate high priority R&D actions with new ideas, (2) those that are currently manageable but will quickly become problematic at the expected scale in a few years due to research gaps, and require new solutions, and (3) those that are rather oblivious to growth in scale, and can be handled with additional implementation efforts to the current software artifacts, or operational adaptations.

I/O for Exascale (1): Scalable I/O is identified as already critical for petascale exascale, with two major issues. One is programming and abstraction, i.e., how I/O would be viewed from 100K+ processes---is the current file I/O abstraction appropriate, or other high-level data persistence models including databases more appropriate. Another is S/W Performance and optimizations, in terms of achievable BW and latency, including new software caching techniques as well as possibly exploiting new storage technologies such as SSDs. Diversifications on the requirements on I/O for different purposes (ephemeral bulk I/O requirements such as checkpointing versus results of data analytics requiring long-term provenance) would result in applications of different abstractions, and more workload analysis is necessary.

Resiliency (RAS,HA, Fault prevention, detection, recovery, management) (1): With system growth faults become everyday already at petascale, due to SW/HW faults, some of them silent. Moreover, recovery costs would also increase due to increase in system state overwhelming I/O for checkpointing. Combined, this will result in substantially shorter MTTI, while the current state of the art in both SW/HW does not deal with faults in a systematic manner throughout the system stack. This would require R&D at all fronts, from characterization of faults at various levels, new HW/SW system design and interfaces that manifests faults as first-class entities, as well as new algorithms for fault tolerance and recovery at various levels, both system as well as applications, possibly exploiting new devices such as SSDs.

Eliminate bottlenecks to strong scaling (hidden latencies in SW) (1): There are impeding problems that are already critical at today's petascale and will be most problematic for exascale. Many algorithms, either at application level or systems level, embody some O(N) (or beyond) portions, severe bottleneck when N~= 1B. Scaling achieved in today's systems is weak scaling; but problem size enlargement, coupled with higher-order algorithmic requirements, will force strong scaling and a decrease in memory per core. This will be problematic, as most systems provide communication with high latency, resulting in systems that will not be able to exploit the 1B-way parallelism. In the light of this analysis, we need to revolutionize software and hardware on all fronts, from architectures to programming models to algorithms, in order to expose and manage (fine-grained) concurrency at all levels, and also allow it to be hidden with local asynchrony, removing global synchronization requirements when physical (speed-of-light) limitations exist.

Parallel programming models (distributed memory and PGAS) (1): Programming models are extremely important in handling 1B way parallelism. In this area, most of the issues have strong connections with intra-node parallelism. We concluded that this issue should be synthesized with intra-node issues.

Parallel Debugging (2): Since past research and the resulting debuggers of today allow for debugging up to ~1,000 MPI tasks, the current practice is to extrapolate the scalability to ~10,000s, which may or may not work. This approach will likely not work for 100,000~1B concurrency, especially with the increased asynchrony needed for scaling. This will require fundamentally new debugging techniques and tools, including applications of simulations and formal methods.

Performance monitoring, feedback, parallel autotuning (2): With increased scale and system complexity, monitoring of various performance metrics –not only that of CPUs but also other metrics such as power, cooling, faults, interconnect, and file system – would become important for characterizing and automatically optimizing the system, the applications, and their interactions. Such profiling needs to be low overhead, scalable, as well as composable with standard interfaces and formats so as to intelligently manage the system in an effective fashion, without excessive overheads..

Power and Facilities (SW only, thermal process migration, energy scheduling /charging) (2): The 100TF-1PF systems of today consume 500KW-6MW; even with HW improvements, power requirements of 100MW or more are envisioned for Exascale. This is primarily due to “beyond Moore’s law scaling”, a factor of x1000 instead of x100 over 10 years, making power/energy possibly the fundamental limiting factor with respect to operational costs at $10s millions/year. We need a drastic overhaul of the power/energy co-design in SW/HW beyond what is being attempted now with DVFS (Dynamic Voltage & Frequency Scaling) and minor improvements in cooling efficiency; rather, alternative architectures and devices with fundamentally order(s)-of-magnitude better power-performance characteristics and their exploitation in system and application SW, e.g., optimized usage of GPUs, phase change memory, SSDs, as well as machine- and facility-level optimizations in novel cooling methods coupled with intelligent power-aware scheduling. The results of these should be measured extensively with underlying monitoring network, and matched to the power-performance models subject to aggressive auto-tuning to conserve power while not sacrificing performance levels.

Resource provisioning (including heterogeneous nodes, workflow scheduling) (2): Although migration is used in enterprise computing for consolidation, in HPC virtualization can be beneficial in providing a level of indirection to support properties that are not associated with the physical system (e.g., dynamic node allocation in workflows, migration after a fault, etc), as well as stronger isolation between applications is critical when multiple applications share a single system. Still, virtualization is rarely supported in today’s supercomputer due to reservations about performance and concerns over scalability of control of a massive number of VMs in a system. Lighter-weight, high-performing VMs, with support for heterogeneous architectures, and scaling to 100,000 to millions of nodes would be desirable.

Interaction with external resources: Clouds, archiving, real-time data streams (2): This topic was not covered due to lack of time and the number of people assigned to other tasks (not necessarily the lack of expertise), and should be covered in future versions of this document.

Systems integration and management (SW bringup, factory testing, transition to production SW state) (3): System SW (including middleware and libraries) as well as their configuration settings, are brittle and require extensive efforts to stabilize, making upgrades difficult since downtime is extremely expensive. This is primarily due to the fact that the impact of multiple changes is unverifiable, as well as the fact that various types of deployment/configuration/change management SW available in enterprise space have not been typically adopted at HPC centers. A more systematic approach to system SW testing and verification in experimental systems, as well as effective use of management SW, would be required.

Topics for future discussion (both intra and inter-node)
There were a number of topics that came up in discussion that were recorded for future discussion.
Characterization of a node: The definition of a node is fluid and ill defined. Although there is no single type of node, it will be useful for discussion to come up with a common taxonomy and agreement on node types. Nodes can be characterized as multi vs. many core, UMA vs. NUMA, OS single system image. These characterizations are important because, although it is not certain that the inter-node programming model will change, the intra-node model must change.
Intra-node and inter-node interplay: Our discussion was intentionally split into two subgroups, but there is significant interaction between the two, which was not discussed at length.
Evolution-Revolution: There was quite a bit of discussion about the implicit (and explicit) constraints that evolutionary thinking can impose on the solution set we will eventually consider. In particular, there was a strong argument that we need to thoroughly explore new models of computation that might be revolutionary but more appropriate to the exascale environment.
Connection to application drivers: Although there is ample evidence of the need for the kind of research our subgroup discussed, and ample analysis of application needs, e.g., Paul Messina's presentation at this meeting, our subgroup did not draw explicit connections to application drivers.

Computational Challenges and Needs for Academic and Industrial Applications Communities

Breakout Group 2

IESP, Saclay, June 28 & 29, 2009

Participants: Giovanni Aloisio, University of Salento, Lecce, Italy; Jean-Claude Andre, CERFACS, France; Mutsumi Aoyagi, UKyushu, Japan; Mike Ashworth, Daresbury, UK; Jean-Yves Berthou, EDF, France; Guillaume ColindeVerdière, CEA, France; Stefan Heinzel, MaxPlanckDEISA, Germany; Laxmilkant Kale, UIUC, US; David Keyes, Columbia U., US; Thomas Lippert, Juelich, Germany; Peter Michielse, NWO, NL; Jean-Michel Muller, CNRS, France; Wolfgang Nagel, Dresden, Germany; Claude Puech, INRIA, France;

The IESP **Application subgroup** was given two main objectives: establish a roadmap to Exascale for scientific domains and document software issues (type of issues, time frame).

Establishing a roadmap to Exascale for scientific domains

The subgroup decided to first identify the representative application domains to be considered, taking into account the HPC EUR and Extreme Computing DoE efforts. The subgroup then listed the scientific and technical questions raised by Exascale simulation and, finally, established a list of experts in US, Japan and Europe that could provide inputs for the "Application roadmap".

The application domains selected are:
- Weather, Climate, Earth Sciences
- Astrophysics, HEP and Plasma Physics
- Materials Science, Chemistry and Nanoscience
- Life Sciences
- Engineering and Finance & Optimization

The table of experts for these domains is given below. A contact person has been identified for each expert among the member of the Application subgroup. The contact persons are in charge of interviewing the experts by mid-September, addressing the following issues :
- Scientific and computational challenges: brief overview of the underlying scientific and computational challenges and potential impact
- Software issues – 2009: brief overview of identified software issues for addressing state of the art machines
- Software issues – 2012, 2015, 2020: expected scientific and technical hurdles

- Expert feedback: identification of the impact of the last machine change the expert has had to face on their applications (porting, optimization, re-writing, …) and on their approach to doing simulation, and the expected impact of the next machine change (going from Tflops to Pflops, Pflop to Eflops).

In order to help the experts answering these issues, a list of potential scientific and technical hurdles have been identified*:*

- *Load imbalance*
- *Communication scheme: synchronization, point-to-point and collective communications*
- *Scaling: weak, strong (time per step)*
- *I/O issues*
- *Parallel file systems*
- *Multilevel of parallelism: core and heterogeneity at core- and chip-level*
- *Memory issues: size per node, bandwidth, latency*
- *Measure of efficiency: per core, per node, global*
- *Solvers*
- *Pre- & post-processing: data management visualization, …*
- *Debugging, performance analysis*

The result of each interview will be summarized in a slide according to a pre-defined format (slide 5 of Breakout#2 PresentationV2.ppt). These slides will be aggregated in a working document by October 2009, third IESP workshop (in best effort mode).

Software issues

The Application subgroup identified the following Computer Science and Applied Mathematic issues. These lists will have to be compared and combined during the Japanese IESP workshop with the technical issues identified by the subgroup 1.

Computer Science issues

Parallel code optimizations
Parallel programming abstractions and support
Adaptive optimizations (load balancing, etc.)
Post-processing of large data sets, pre-processing
Coupling multiple modules: computational schemes, control and data aspects, parallel composition of modules
Data distribution, replication, integration, integrity, security
Metadata, ontologies management
Storage management and parallel I/O
Integrated framework: workflow, common data models, components interoperability

Applied Mathematic issues

Partial differential equations
Solution adaptive methods
Molecular dynamics modeling and algorithmics

Particle methods (distinct from classical MD in terms of inhomogeneous distributions, e.g., cosmology)
DFT/electronic structure
(hybrid) Monte Carlo
Stochastic optimization
Scalable implicit solvers (linear and nonlinear)
Eigenanalysis techniques
FFTs and other fast transforms
Smoothed particle hydrodynamics
Agent-based methods
Coupling multiple models (stability aspects)
Data assimilation/fusion of observations and models
Data mining, clustering, classification, statistical analysis and representation
Uncertainty quantification

Annex: list of identified experts

	Experts to be contacted	Expert Organisation	Email expert	IESP contact	Region	Comment
Weather, Climate, Earth Sciences	ECMWF	ECMWF		JC André	EU	Climate,
	NCAR	NCAR		Giovanni	US	Weather,
	A. Sumi			Mutsumi AOYAGI	Japan	Weather, Climate
	H. Calandra	TOTAL	Henri.CALANDRA@total.com	JP Nominé	EU	Earth Science expert
	??			Mutsumi AOYAGI	Japan	Earth Science expert
	??			Sanje Kale	US	Earth Science expert
Astrophysics, HEP and Plasma Physics						
	W. Hillebrandt			Stefan Heinzel	EU	Astro
	??			Stefan Heinzel	EU	Astro
	Quinn			Sanje Kale	US	Astro
	E. Audit	CEA	edouard.audit@cea.fr	JC Nomine	EU	Astro
	Ricker			Sanje Kale	EU	Astro
	M. Normann			Sanje Kale	US	Cosmology
	S. White			Stefan Heinzel	EU	Cosmology
	J. Makino			Mutsumi AOYAGI	Japan	Cosmology
	T. Lippert			T. Lippert	EU	LQCD
	R. Kenway			T. Lippert	EU	LQCD
	A. Ukawa			Mutsumi AOYAGI	Japan	LQCD
	S. Gottlieb			T. Lippert	US	LQCD
	S. Gunter			Stefan Heinzel	EU	Plasma physics

	E. Sonnendruecker			C. Puech	EU	Plasma physics
	Sato			Mutsumi AOYAGI	Japan	Plasma physics
	R. Parinello			Nagel	EU	Materiels science
	G. Martyna			Sanje Kale	US	Materiels science
	Binder			Nagel	EU	Materiels science
	G. Zerah			Colin de la Verdière	EU	Materiels science
	K. Terakura			Mutsumi AOYAGI	Japan	Materiels science
	Scheffler			Stefan Heinzel	EU	Materiels science
	T. Dunning			Sanje Kale	US	Chemistry
	J. Pople			Mutsumi AOYAGI	Japan	Chemistry
	K. Hirao			Mutsumi AOYAGI	Japan	Chemistry
	D. Marx			Nagel	EU	Chemistry
	R. Harisson			Sanje Kale	US	Chemistry
	T. Schulthess	CSCS	thomas.schulthess@cscs.ch	JC Nominé	EU	Nanomateriels
	T. Deutsch			Colin de la Verdière		Nanomateriels
	S. Bluegel			Nagel	EU	Nanomateriels
	Cuniberti			Nagel	EU	Nanomateriels
Life Sciences						
	K. Schulten			Sanje Kale	US	Molecular biology
	A. Grubmueller			Stefan Heinzel	EU	Molecular biology
	T. Simonson			A. Lichnewsky	EU	Molecular biology
	R. Lavery			A. Lichnewsky	EU	Molecular biology
	H. Nakamura			Mutsumi AOYAGI	Japan	Molecular biology
	Masella			A. Lichnewsky	EU	Molecular biology
	K. Schulten			Sanje Kale	US	System biology
	Lengauer			Stefan Heinzel	EU	Bio informatics
	D. Bader			Sanje Kale	US	Bio informatics
	Aluru			Sanje Kale	US	Bio informatics
	Masella			A. Lichnewsky	EU	Molecular biology
Engineering						
	C. Rossow			JC André	US	Aeronautics
	E. Chaput	AIRBUS		JC André	EU	Aeronautics
	D. Emerson			Ashworth	EU	Aeronautics
	Schroeder			Nagel	EU	Turbo machinery
	P. Moin			JC André	US	Turbo machinery
	Y. Fournier	EDF	yva.fournier@edf.fr	JY Berthou	EU	Nuclear energy (CFD)
	T. Courau	EDF	tanguy.courau@edf.fr	JY Berthou	EU	Nuclear energy (Neutronics)
	C. Chauliac	CEA		JY Berthou	EU	Nuclear energy (Neutronics)
	??				US	

	??				Japan	
	M. Heath			Sanje Kale	US	Rockets
	K. Fuji			Mutsumi AOYAGI	Japan	Rockets
	JD Mattei	EDF	jean-daniel.mattei@edf.fr	JY Berthou	EU	Hydraulics
	Kaneda			Mutsumi AOYAGI	Japan	Hydraulics
Finance & Optimisation						
	X. Warin	EDF	xavier.warin@edf.fr	JY Berthou	EU	Production optimization
	G. Zerah			Colin de la Verdière	EU	Finance

Role And Participation Of National And International Funding Agencies

Breakout Group 3
IESP, Saclay, June 28 & 29, 2009

Participants: Franck Barbier, ANR, France; Bertrand Braunschweig, ANR, France; Fabrizio Gagliardi, Microsoft, Italy; Kostas Glinos, EU, EU; Fred Johnson, DOE, US; Alain Lichnewsky, Genci, France; Paul Messina, ANL, US; Hervé Mouren, Teratec, France; Abani Patra, NSF, US; Catherine Rivière, GENCI, France; Ed Seidel, NSF, US;

This breakout group considered existing funding programmes and cooperation mechanisms that could be used in support of research in exascale computing, the motivations and conditions for cooperation at funding authority level, as well as actions that could be undertaken in the near future.

It should be noted that the group could not be considered representative of funding authorities. Whereas the US was relatively well represented, from Europe there were only representatives of France and the EC and there was no-one from Asia. It is presumed that the next meeting will try to attract more participants from these agencies.

HPC research funding in the US is of the order of $200 -$300 M/year (without DARPA programmes, and without development and procurement) dispersed among multiple agencies and often embedded in science funding. In Europe it is probably in the order of €80-100 M/year, including several national and European programmes. Cross-Atlantic R&D partnerships are feasible and not uncommon; they are however challenging due to problems in synchronisation of funding. Typically each country supports its own researchers.

Global cooperation in exascale research seems to be motivated by three considerations: first, exascale computers will be necessary to address overarching societal challenges such as climate and energy, which are by definition global; second, extreme computing will be driven in the future by all science fields rather than by a few classified needs and can therefore be more open rather then the "privilege" of a few; and third, and most important, the research challenges and associated costs and risks to develop exascale systems would be so high that no country or even continent would be able to go it alone. It was stressed that exascale represents a paradigm shift in computing and therefore requires *ab initio* efforts, whereas current petascale efforts should be continued in parallel. Education and training of a new generation of "global" researchers for whom concurrency comes natural would be key in order to achieve the critical mass of human resources to face the exascale challenge.

The justification of exascale computing on the basis of social and scientific needs is strong and clear although needing quantification. On the other hand, however, industry may not perceive a significant and short term ROI, especially since this is uncharted and high-risk territory. Better envisioning of potential applications and involving application software vendors would be helpful.

Participants agreed that international collaboration in exascale should involve joint road mapping and periodic workshops, coordination of investments to avoid excessive duplication, standardisation, as well as joint projects in specific cases. A "clearing house" of all funded efforts could be useful. Common policies to promote effective reuse, such as open source policies, are certainly necessary; as most software is expected to be produced in open source, funding authorities should address the problems of long-term support and maintenance.

The participants identified the need for a structured and stable framework for international collaboration, which should be open to all those who can contribute. Current collaborations are bilateral and mostly driven by individual initiative; multilateral agreements at institution, funding agency or political level could help in achieving a more stable framework.

Future meeting opportunities are during the EGEE 2009 Conference in Barcelona on 21-22 September and then in Tsukuba on 18-20 October. It was thought that first draft plans for an international effort could be presented by April 2010.

Economic and Management Challenges of Computational Resource Providers and Industry Partners

Breakout Group 4

IESP, Saclay, June 28 & 29, 2009

Participants: Patrick Aerts, NWO; David Barkai, Intel; Taisuke Boku, U of Tsukuba; Iris Christadler, LRZ; Hugo Falter, ParTec; Alan Gara, IBM; Jean Gonnord, CEA; Andrew Jones, NAG; Kimmo Koski, CSC; Bill Kramer, NCSA; Peter Michielse, NWO; Jean-Francois Lavignon, Bull; Dan Reed, Microsoft; Christian Saguez, Teratec; Makoto Taiji, Riken; Peg Williams, Cray;

Introduction

The working group was composed of representatives from government and academic resource providers and major high-performance computing hardware and software vendors. The primary focus of the extended discussion was on collaboration mechanisms and processes between resource providers and industry partners (vendors). The group identified five major relationship models among vendors, national laboratories, universities and provider consortia, noting that multiple models could be used in any given program.

Within this framework, the participants discussed the relevance of different approaches, as driven by the potential market size for exascale systems and the probability of broader market penetration via "trickle down." The group's operating assumption was that any exascale program had to have broader market penetration at below exascale levels for the technology that is developed and used at exascale. Otherwise, the entire non-recurring engineering (NRE) must be borne by the limited number of exascale system purchasers. In the latter case, an exascale initiative would be more similar to limited experimental facilities such as the Large Hadron Collider (LHC) or the International Thermonuclear Experimental Reactor (ITER).

With this backdrop, the group discussed the lessons that could be drawn from other large scale projects, the practical implications of hardware and software on high-performance computing markets, and mechanisms for possible collaboration.

Lessons from Large-Scale Projects

One can draw insights from the history of other large scale, national and international projects, ranging from computing at terascale and petascale to large science and technology projects such as the LHC, ITER, telescope consortia, satellite instrumentation and similar ventures. The culture and competitive market (whether commercial or research) strongly shapes outcomes.

In most of these cases, there is a single, coherent science problem being addressed, which is deemed by a (typically large) portion of the related science community, as being worth the cost of the instrument or project. Often, as with the LHC of the Joint Dark Energy projects, there is a single primary question andfor the instrument and then possible several secondary explorations within the same general area of science. Few of these projects have explicit broader marketability and in general these are view as "one of a kind" and, in some ways, "disposable" equipment – with the knowledge of the result as the lasting benefit.

An alternative approach is the creation of a new family of aircraft or a new line of automobiles. In these, cases, the investment is similar and involved large teams of engineers to produce what on the surface is a new machine. Under the surface there is probably a large amount of new software to make the machine work, as well. These projects are driven my market analysis and market forces in that the systems created have to serve a range of requirements, be flexible to appeal to a broad market and be efficient in ownership.

The exascale computing project is a mixture of these types of projects, and needs to adapt effective approaches from both camps. What do the ranges of ultra-scale projects tell us needs be in the exascale project?

One overarching conclusion from the working group discussions was the importance and the danger of metrics. Metrics define collaboration and system outcomes and must be chosen wisely. An excessive focus on peak performance, for example, can lessen the probability of producing reliable, usable systems that are applicable to a broader market. Similarly, how funds are distributed across hardware, software and applications, and across procurement versus research and development, strongly determines the types of systems produced.

Another conclusion is an integrated, coupled approach to hardware and software design, development and testing has to be part of the project. This is coupled with effective project management in order to make good choices, but also to remain flexible so that new information and improve the result.

Markets and Implications

Vendors typically make investments that can be amortized across a broad base of customers. If we examine the history of the Top500 list, we see that while there are few systems to reach a key milestone (GFLOP, TFLOP) initially, roughly 10 years post the initial system, the top 25 is dominated by systems at the milestone performance level. This would seem to imply that investments to reach a certain performance level could be easily justified. However, the system scaling characteristics are very different between the initial system and the "quantity" systems. Consider three dimensions of scaling with regard to software:

- **Number of processors/cores managed by a single OS image (Scale-Up):** The Operating System, compiler, math library, and OpenMP software scaling requirements are directly tied to this. Historically, this has been the number of processors in an SMP or the number of cores in a multi-core socket. This line is blurring as some vendors have been building a multi-socket nodes.

•**Number of Operating System Images in the system (Scale-out):** System management software scaling requirements are directly tied to this. Historically, there has been a one-to-one correlation between the number of hardware nodes and the number of OS images though the definition of a node varies.

•**Number of MPI tasks in an application that runs on the full-system (Scale-Across):** MPI, parallel math library, global debugger, and performance analysis software scaling requirements are directly tied to this. Historically, this has been the Number of OS images * Number of processors/cores managed by a single OS image. However, with accelerators such as cell processors, GPUs, this line is again blurred. Additionally, the flat programming model of one MPI task per core/processor is becoming very difficult for application developers and system software providers as well.

The table below summarizes these scaling dimensions for 1 GF and 1 TF systems when they first appear and then five years later. By ten years later, systems at those performance levels no longer appear on the Top500.

	1GF		**1TF**	
	Initial	Initial + 5 years	Initial	Initial + 5 years
Scale-up	8	4	O(1)	O(1-10)
Scale-out	1	1	O(1000)	O (10-100)
Scale-across	8	4	O(1000)	O(100)

In the GF era (1988- 2003), these systems were predominantly shared memory SMP machines. As processor frequency was increasing, the Scale-up dimension was decreasing. In the TF era (1998-2003), the 1 TF systems five years later were roughly an order of magnitude smaller in the Scale-out and Scale-Across dimensions and up to an order of magnitude larger in the Scale-up dimension.

This would suggest that investments required to meet the needs of the scaling needs of initial systems are not easily justified by the vendors – at least in the Scale-out and Scale-across dimensions. This is largely driven by economics; the $/MF five years after the initial systems appear are much less than and are more affordable for more customers. Investments in Scale-up are more easily justified assuming that the vendors are using the same building blocks in smaller systems as they are in the larger scale-out systems.

It is also important to note that between the GF era and the TF era, a major architectural shift occurred from large SMPs to MPPs or clusters of SMPs. Large software investments from both government agencies and vendors were needed to facilitate this shift.

As we look ahead to exascale, there appears to be another architectural shift occurring as well as increases in all scaling dimensions. If we assume the first exascale systems will appear in ~2018, vendors are unlikely to any financial benefits to investing in software needed to facilitate the architectural shift until 2023 (five years after the first system) and even then the scaling needed would likely be less. This suggests that the vendors must be subsidized in order to make the necessary investments now.

Additionally, we must work from the assumption that there will be a small number of HPC exascale systems in the period of interest (circa 2017-2020). It also behooves us to assume that no single vendor or a single solution will capture all of these early deployments. Therefore, the system developers will have to be cautious not to invest too much in capabilities or features that benefit only exascale systems. Even if we expect that, in time, novel features will waterfall to mainstream computing, this does not mean it is economically wise to invest in such features earlier. For example, the underlying technologies may still be too immature or in experimental phase. The implications of the above are twofold: (1) we expect the exascale solutions to build on mainstream components wherever possible. (2) improvements and innovations needed for the early exascale deployments will have to either be deployable on smaller scale systems (for example, a fast interconnect chip; a modest size network switch, etc.), or will require Government and public sector support – financially or in terms of resources (for example, power and cooling aspects for a very large tightly coupled system).

Interaction Models Throughout the Software Lifecycle

MPI, LAPACK, and Linux are good examples of development models that have been successful previously. In each of these cases, the specifications are clear, source code for implementations is available, and interfaces are well-documented. Vendors have the opportunity to optimize for their platforms while maintaining the interfaces provided. As we proceed with software plans for Exascale, learning from and building upon these successes is critical.

For any new technologies developed, the vendors should be active participants in the specifications and implementation stages, not just in optimization, test, and support. Likewise, the community must be active participants in the test and support stages, not just in specifications and implementation. In the Linux world, the Linux distributors play a key role in transitioning between community-developed code and hardware vendors. Vendors can rely on the distributors to work with them on critical fixes, general software maintenance, version control, etc. Since the software infrastructure for Exascale will likely be much more complex than we have today, we should consider creating a "software clearing house" to perform a role similar to the role the Linux distributors play for Linux. Additionally, the initial Exascale systems will likely be a complex integration of new hardware components, a new software infrastructure, and new software components. The procurement model in place today at most customer sites shifts most of the financial risk to the vendor. Alternate shared-risk models seem more appropriate for this type of task.

The demands of exascale require a complete rethinking of the software and hardware development process. For the past 10-15 years, horizontal layers of software and hardware design and development have been the de facto standard of creating HPC software. This situation is due in part to the influences of funding methods, research incentives, software methods, the open source movement and commercial outsourcing and specialization. This horizontal design approach leads to the development of discrete components in the software stack and independent hardware components – all developed with different methods, requirements and quality. Past generations of system software (from the earliest operating systems through to the original community source movement with Unix) and hardware, showed some degree of top-to-bottom vertical design. The last 10-15 years, however, have been dominated by plug-and-play componentization that is focused on horizontal functionality and portability.

The horizontal design approach has notable successes like the Linux kernel, MPI and a variety of job schedulers. It also has many challenges that are inhibiting progress and making even petascale systems challenging to fully exploit. As many who field terascale clusters know, every cluster is now unique with different horizontal components (often in name, at least in version). Currently, at the tera- and petascale, there is only one company that produces a system software stack that is vertically designed from top to bottom and two other companies that is providing a scaled, vertically tested stack that has specifically designed components added to the horizontal components.

To reach petascale, the HPC community has mitigated many of the issues in the horizontal design method such as:

- relying on vendors to do vertical testing and integration,
- standing up extra test bed resources for integration testing and error correction,
- taking excessive time from the few large-scale production systems to do integration, testing, diagnosis and correction, or
- living with inefficient and error-prone systems.

The current horizontal design method presents a number of insurmountable challenges to reach exascale. Yet economics and cost effectiveness will not let us return to the days of completely proprietary vertical methods. Nor can one organization alone, be it government or private, afford to pioneer exascale and make it a success.

Exascale requires abandonment of the horizontal approach of developing essentially isolated software components that have a narrow function and role in the system. Instead, the community must organize hardware and software development activities with component cross cutting principles. These principles should define the requirements, function, interfaces, integrations and performance needs for each horizontal component. Instead of thinking of integration as the final step in defining and developing an exascale system, it will be the first step at exascale.

The cross cutting requirements for the vertical design approach were identified to first order at the first IESP workshop. They include: resilience (reliability and fault tolerance); performance; programmability; computational models; I/O; consistency and verification; resource management; and power management/total cost of ownership.

There are limited proofs of existence that hint that the vertical design approach yields an effective and long-lived, yet flexible, solution to this conundrum. Some examples include:

- The DOE SciDAC program. SciDAC introduced the concept of software application development teams and software infrastructure teams that are linked in an iterative approach to developing applications that rely on increasingly more effective software infrastructure.
- Scientific “framework” development for large-scale experiments and long-lasting infrastructure. High energy physics regularly uses a formal process of vertical architecture definition, software development and testing often incorporating thousands of funded and unfunded contributors. The processes here are notable for progressive demonstrations of integrated milestones (e.g. Data Challenges) and timely delivery of equipment that is being co-developed.
- The methods used to produce community-based software such as the High Performance Storage System, which follows formal methods and shares development and support across multiple organizations via formal agreements.
- Commercial operating system development methods such as those at IBM, Sun and Cray.
- Formal testing methods that are used in the verification and validation of network protocol change proposals.

One factor motivating a renewed emphasis on vertical integration is the dominance of software failures as the causative factors in large-scale system lack of availability. Failure at scale of system software such as file systems, batch schedulers and even authentication mechanisms such as LDAP is a major problem for HPC resource managers. In many cases, vendors leverage software that works well at smaller scales, but they place too much reliance on the ability of the software to integrate seamlessly at all levels. Some studies indicate that on large systems, across vendors and architectures, software accounts for the majority of systemwide failures on HPC systems.

User experience can also suffer when developers pay insufficient attention to end-to-end software functionality. The usability of tools such as debuggers and performance profilers can diminish significantly when they are used beyond the scale that software vendors are capable of testing. Usability and thus the value of the HPC resource to science can be improved by a more goal-oriented vertical approach, one that builds in expectations of usability at scale.

The vertical approach is not at odds with previous approaches to HPC software development. A tightly coupled vertical design can still produce software that is of lateral use, but attention to vertical integration diminishes risks of software failure when relying upon "off the shelf" generic software. Vertical integration does not replace these software components but improves them for HPC.

Exascale Software Base: Top500 Extrapolations

It might be argued that the collective effort and cost incurred by the proposed international Exascale software project is only justifiable if the developed software ecosystem is useful to multiple (even many) systems, rather than focused on a very small number of "hero" systems – i.e. only the first 1-5 exascale supercomputers. This concern is especially true in relation to the desired strong participation by the industry partners and vendors. The proposed software ecosystem roadmap must therefore be of value to other systems: either "near-exascale" systems (i.e. many petaflops) installed at the same time with comparable hardware characteristics to the initial exascale supercomputers; or exascale systems deployed several years after the "hero" installations, when as expected, exascale systems become numerous (and the heroes are heading to many-exaflops).

Thus, in return for the collectively invested effort, the IESP collaborators expect that (software) products will evolve from the research deployments of the hero exascale supercomputers. This enables the industry partners and vendors to anticipate an additional source of return on investment from their efforts, and enables the academic collaborators justify the funding agency investments by impacting the broadest set of potential scientific users.

Thus, the question is whether the characteristics of supercomputers across the top ~50 are sufficiently similar to the first exascale systems for them to derive substantial benefit from the exascale software achievements. Likewise, will the characteristics of the few hundred exascale supercomputers expected to be deployed ~5 years after the initial hero systems, be sufficiently similar to the heroes for the software breakthroughs to be critical?

Although thorough study of roadmaps might provide a better assessment of this, a useful early assessment can be established by studying the Top500 data. To do this, we look at the characteristics of first major deployments and compare these to the systems when the same performance dominated the bottom 100 of the Top500. For example, when multi-teraflops supercomputers first entered the Top500 (i.e. at the top few), perhaps June 2001, they were MPP systems, with O(1000) processors using multiple processors (cores) per node, and achieved around 7 teraflops HPL score from around 12 teraflops peak. Some 8 years later, the supercomputers in places 400-500 on the Top500 list achieved around 20 teraflops HPL score from a range of 20-50 teraflops peak, using a few thousand cores in an MPP/cluster configuration with multiple cores per node. This single spot case would seem to support the assertion that, at least at a very high level, the software requirements of near predecessors and successors of the first exascale systems will be broadly similar. Clearly, this needs to be investigated across further data points in addition to this spot case, in order to develop confidence in the assertion.

Collaboration Approaches

The group identified five major relationship models among vendors, national laboratories, universities and provider consortia, noting that multiple models could be used in any given program. Each model has degrees of overlap and interpretation. The group considered the following high levels models for interactions between the different roles in such a complex project. An important outcome of a model is they essentially set common expectations for all parties, in an explicit manner, early enough that all parties (resource providers, industry partners, stakeholders, the science community) know what to expect of the system.

- *Funded investigation.* This model defines a research exploration whose outcome is insight, knowedge and informed opinion. There is no expectation of an outcome other than the production of knowledge, possible with resultant licensable intellectual property (IP).
- *Fully defined purchase.* This model is a standard product acquisition, where a vendor delivers a working product, under terms of a purchase, with normal warranties and expectations.
- *Design and development (no product).* Creating and demonstrating prototypes of new approaches and technologies without a deliverable "product" is the hallmark of this model.
- *Co-design and co-development.* As will a purchase, this model results in one or more working solutions, but in this case the risk and the investment are shared by multiple parties, typically the user and the vendor.

- *Base plus value add.* The final model produces a solution with some baseline value, with additions whose development may be shared by multiple parties.

These approaches are discussed in greater detail below.

The development of an efficiently usable Exascale computer environment requires a tremendous effort of all parties involved: hardware and software developers, research labs, users at all scales and governments. But the benefits will serve all. The financial input to this endeavour, however large it may be, needs to be spent in the most efficient way. This requires best estimates for the total cost and the breakdown of the cost in segments, it requires planning and public reporting and coordination of efforts to guarantee continuity of process and prevention of capital destruction (due to bankruptcy, time gaps between efforts, accidental doubling of efforts, etc). At the same time an open eye must be kept on differences of interest, conflicts included, the balance between public and private, culture -if even between continents, as we address a global effort- and the balance between communality and competition.

Why would this effort to reach exascale in computing be different from the effort to get to the unimaginable petascale when that project started in 1994? There are a multiple reasons:

- The marketplace has completely changed in the past fifteen years:
  - other vendors;
  - less chip developers;
  - other margins on investment and in sales: more volume, less revenue per element;
  - HPC has become more democratised: more volume cycles concentrating in the middle layer;
- The balance between single processor performance and system performance has changed: more components involved, such as network, memory layers, software layers, etc.
- The component density on the chip seems to have reached the ultimate limits: we can forecast the real boundaries now as we approach the 100-atomic distances level.

All-in-all, we can expect that we may be better in estimating what we are up against, compared to 1994, but be less predictable in estimating how to solve the problems ahead of us. To give but one example: in 1994 the discussion was by far dominated by technical discussions. Seymour Cray even considered biological components necessary in the fabrication process of smaller devices. And yes, there was a discussion on software too. The situation today is that we do have petascale systems, we note that our software did not develop at the appropriate rate to fully exploit the newest technology at full scale, certainly not in all disciplines and now see the urge for investing in software. And we conclude that software will be the bottleneck for Exascale computing as well. And it may be true. But as we are getting so close to the limits of feature sizes on the chip, while at the same time reaching the boundaries of affordability if it comes to power consumption, it may turn out that hardware again be the hardest part after all.

Yet we will have to decide how to divide development cost estimates into hardware and software, if only because both sides of the coin will involve different types of parties and different types of funding. For the sake of further discussion, let's put hardware and software development costs for Exascale computing on equal footing: 50-50.

We addressed the following topics regarding funding and development:

- Funding and planning;
- Competition versus cooperation;
- Collaboration options and opportunities;
- Impact and implications of open source.

Funding and Planning

Basically the exascale endeavour is a man-to-the-moon (or Mars) type of effort. We know where we want to go, we may even know when, but not yet how. But it is also an effort the benefits of which lie well beyond the realm of the parties directly involved. It is a driver for further development in IT at large; it will result in side products for the broader masses and new industrial dynamics. This is why a serious public (financial) support can be applied for that will give the public return on investment. Funding parties in this case are:

- National labs and government departments (US mainly);
- National research councils and academies;
- National funding programs for economic, technology and scientific support and development;
- European Framework programs and beyond.

All continents involved, America, Europe and Asia, seek their involvement in all aspects of exascale computing: hardware development, system design, software development, industrial strengthening, and scientific research. The discussions on this topic so far have learned that in this stage of development all interests are common to all: the stage of competition at that level (continental or national) will come, but much later in the process.

Two elements to be coordinated:

- Contracts between hardware developers and developments (in the broadest sense) and the software developers (also at all levels of software);
- Tuning of funding schemes, including the issue of international cooperation.

All in all it is worthwhile to encourage all funding parties involved to collegially address the issue of collectively supporting a world scale effort, by at least creating awareness of the existence of the activity, but without directing the process any more than required to reach the goal. In other words it is not the intention to have everything coordinated in advance, because that also would be counterproductive.

The advise, however, is to organise or at least orchestrate a communication between all funding sources to make clear;

- What Exascale is about;
- What may be expected from such an effort;
- What the advantages may be, also in terms of real money, of a fair level of tuning of efforts, in focus, in timing and funding conditions, between the funding organisations themselves and between them and the Exascale project.

As part of trajectory –for the stakeholders to get matters more clear and for funding parties to know better what to focus on, the whole workload could be subdivided in *bundles* of coherent activities. For instance a single bundle could consist of compilers, libraries and tools and the development of new languages. Such bundles could be procured in a competitive manner to consortia of research groups and industrial partners (could be, but not necessarily limited to, manufacturers). The result of such procurements, for the total set of bundles defined would give an estimate for the total cost of development. The procedure could yield a healthy level of competition, and two or more consortia working on the same bundle would yield possible ways to the same goal. Per bundle one could establish or select a best funding method or source and per bundle one could decide whether this bundle should be handled by industry, by the scientific community or both. A condition from the onset would be the agreement on open standards, to avoid later problems to connect all the bundles or exchange best parts from each bundle into a final product.

Competition versus Cooperation

It was clear from discussions on this topic, that there will be a time where the collective interest in exascale computing will cease to exist and the market/scientific community is best served by competitive offers of systems. And in the meantime there will for sure be trajectories where the stage of competition will arise much earlier than for other parts. After all, many software improvements towards parallel computing will be beneficial for the community immediately, because they will work as well at the Petascale level. The conclusion is fairly safe that the path to Exascale computing will go a long way in a cooperative rather than a competitive approach if we consider the development from the point of view of the man-on-the-moon project. That is if we draw up the final product and then try to figure out what we need to do to make it. The route will be dominated by competition only, if the project is considered from the point of view of the sum of incremental changes.

It seems, however, that both industry and the scientific community better take the view that the design be largely from scratch, of course taking for granted what we know already and the development lines we can already estimate, because incremental development will only make us hit the wall harder when it comes. The efficient use of an exascale system will never come if through incremental adaptations of what we have today.

In order to get a better grip on this matter it is advised to deepen this topic further. The complicating factor is that all other “normal” developments will continue as well and that many results on the path to exascale computing will find their way in the daily market as soon as possible.

Collaboration Options and Opportunities
From actual practice in the world of HPC up to today five major types of distinguishable relationship models can be found. These will be compactly elaborated on.

- *Funded investigation.* This model defines a research exploration whose outcome is insight, knowledge and/or informed opinion. This can be either fundamental or applied research, since there is no concrete expectation of time frames for breakthroughs, schedules or performance. Although prototypes or one-of-a-kind systems are sometimes created, there is no expectation of an outcome other than the production of knowledge, possible with resultant licensable intellectual property (IP). While it is always hoped that some significant, usable outcome is possible, this model's typical product is reports, prototypes, analysis and papers, with no guaranteed product. In this model, the funding is paid as work progresses rather than by periodic achievements. Risks to the investigators are minimal and are really only to reputation (if things do not work) or follow-on opportunities. Examples of this approach are the base R&D funding for NSF and DOE/Office of Science, the series of Illiac systems and the Multics Project.

- *Fully defined purchase.* This model is at the other extreme from funded investigation. It is what is thought of as a standard product acquisition, where the resource provider or stakeholder issues a request for a system to meet a set of requirements. Then, one or more industry partners and/or resource providers bid a solution implementation to meet the requirements and the resource provider selects a system they believe will prove the best solution. Once selected, an industry partner delivers a working product, under terms of a purchase, with normal warranties and expectations. This model works best for products that have a broad market and are well defined.

  It should be noted that over the past several years, many HPC acquisitions following this model have moved from the traditional "lowest common denominator" procurement to a "best value" method, which is more like commercial best practice and provides the bidding industry partners and selecting resource providers more flexibility. In some cases, there is an intermediate party – an integrator – that works to put all the parts of a solution together – possibly from multiple industry partners.

  Nonetheless, this model results in a completely defined set of contract terms industry partners must meet, with most of the funding going to the industry partner only upon delivery of a fully working system. It should be noted here that some view this as transferring *all* risk to the industry partner. Indeed, the financial risk is often all on the industry side, but it is important to acknowledge the resource providers and stakeholders bear substantial programmatic risk, as we have seen at the John von Neumann Computing Center and other sites when industry partners failed to produce working systems. Indeed, for many resource providers – they "bet the farm" with each major acquisition. Examples of this model include the major ASCI procurements and the DOD Modernization Program Technology Insertion process.

- *Design and development efforts only (no product).* Creating and demonstrating prototypes of new approaches and technologies without a deliverable "product" is the hallmark of this model. While still implemented as a contract, it is more a "best effort" than a guaranteed system, where the resource provider is requesting the development of technology. The industry partner(s) then creates the technology and demonstrates its potential to achieve the goals desired. Sometimes a system is produced that is used some period for science or engineering applications, but often the technology is not "production" ready and/or only in existence for a short time. The funding strategy for this model typically has "milestone" payments as well as a payment associated with the final demonstrations. In some cases, the final demonstration may be only a small fraction of the overall cost. The cost of the investment depends heavily on how much other development funding can be "leveraged" or justified for other purposes (e.g. other potential markets). Risks to the investigators are modest and are to reputation and follow-on opportunities or some part of the final demonstration costs. The current best example of this approach is the DARPA High Productivity Computer System project and the ASCI Pathforward awards.

- *Co-design and co-development.* Similar to normal purchase, this model results in one or more working solutions, but in this case the risk and the investment are shared by multiple parties. Often invoked in situations where no single organization can afford to do something by themselves, in this model, multiple parties share the effort, cost and risk to design, develop and possibly use a solution. There can be many partners or only a few, and the roles may be assigned according to the strengths and/or needs of the partners. The roles are defined in an official manner as are the responsibilities. There is a co-dependence that must be taken into account and carefully managed in order to assure all organizations are delivering on their responsibilities. This model has been used successfully, and also has had very notable failures. On the success side, Sandia's ASCI Red – primarily a co-design between SNL and Cray – produced not just a working system, but an entire product line; and the development to the Cell Processor with IBM, Sony and Toshiba. On the failure list one might list the Burroughs's Scientific Processor, the American SST program and the joint deployment of a system by Sony and Virginia Tech.

- *Base plus value add.* The final model produces a solution with some baseline value, with additions whose development may be shared by multiple parties. This model is in some ways an adjustment to the fully defined purchase model above. It is possible to have a purchase of more conservative, base functionality that will partially meet the desired requirements, but then provide additional incentives for partners to add more value or exceed the base specifications. The base can be set at such a level that it reduces risk for partners and still provides some value by meeting some of the objectives. Then, if the base is exceeded (for example the performance is better, the system operates for a lower cost of ownership, or the system is delivered sooner) there is added financial reward to the partners. The logic in the system is more value to the science community since it is more powerful or available sooner.

This approach is seldom used in HPC system acquisition, except in a reverse manner where contracts have allowed industry partners to deliver less equipment than originally planned if the performance of the building blocks exceeded requirements (examples include at least one contract at NCAR). However, it is often used in other government and commercial contracting – in particular in important, time critical or large construction contracts. The challenges in this model include government budgeting issues, defining a sufficient enough base level, determining the relationship between the added value and the added costs and determining what the constraints are out of the control of the partners. One notable success of this approach was the reconstruction of the fire damaged I-80 overpass coming off the SF Bay Bridge. In this case, the repair was completed 33% faster than the best expected schedule and the contractor was provided a bonus based on how many days early the project was completed.

The following table is a summary of the five models.

Model	**Vendor Risk**	**Provider Risk**	**Stakeholder Risk**	**Payment**
Funded investigation	Reputation	Reputation	Nothing useful created	As work is done (Level of Effort)
Fully defined purchase	All Financial + reputation	All Programmatic + reputation	Mission	Almost all at the end after a completely working solution is delivered
Design and	Some	Some	Science	As work is done

development efforts only (no product).	Financial + reputation	Financial + All Programmatic + reputation	mission not accomplished	(Level of Effort) with some after demo
Co-design and co-development	Co-mingled risk	Co-mingled risk	Co-mingled risk	Not specified
Base plus value add	Financial Risk for base function + upside potential for value add income	All Programmatic + reputation	Risk only base function achieved + risk of unexpected budget swings	Almost all at the end after a completely working base solution is delivered + value add payments as appropriate

Software and the Implications of Open Source

Today, the software domain can be fairly split into the following components:

- System software (such as the basic OS and system operation overview)
- Compilers, debuggers, performance analysis tools;
- General software libraries;
- Special functions libraries and system calls;
- Third party application codes;
- (Individual) user codes;
- Special elements, such as visualisation (interfaces).

It has been noted that this subdivision may not survive in this very form as a result of future developments. For instance the role of the OS could theoretically change so dramatically that this layered table has to change accordingly. But let us then consider this list as our time zero measurement of the present situation and see how it develops.

Scientific and technical research requires unambiguous backtracking possibilities and complete reproducibility. Software for critical systems (such as airplanes) has to be certified and systems hardware and software cannot be excluded from such certification. This is for obvious and mainly practical reasons. But the validity of science as such also critically depends on reproducibility. Such reproducibility should go beyond just the notion that it would be *theoreticall*y possible. It should be *practically* possible to do. As a consequence all software involved in the production of a scientific result should be verifiable as should be the formal functioning of the computer system used.

Insights such as these are increasingly becoming commonplace and the development of the “open” community cannot be ignored. And it makes sense, starting a new era of development towards the exascale goal, to set an example and base all developments on the “open” concept. This is not the same as suggesting that all delivery of software related products should be free of charge or all use should be free of charge. Such demands will depend on company business models, on grants conditions etc.

Intellectual property issues are not really different in developing exascale computers as they are in any other cooperatively and/or funded effort. And as usual the intellectual property issues have to be settled before any agreement between a funding party and the grantees is made or before an agreement between a vendor and a cooperative party is set up.

Conclusions

The group concluded that funding profiles (distribution of funds across time) and the mix of hardware and software investments had substantial effect on the types of collaborations that were feasible and n their sustainability. Such collaborations could range from a tightly coupled collaboration with well defined milestones and international governance though confederation and information sharing to relatively independent developments and *ad hoc* interactions. There was general feeling that the lowest levels of software (i.e., systems issues) were the most difficult avenue for collaboration simply because they were so dependent on hardware and vendor-specific issues, necessitating clear involvement by the vendor(s).

IESP Paris June 2009 Participants

Giovanni	Aloisio	University of Salento, Lecce	Italy
Patrick	Aerts	NWO	NL
Jean-Claude	Andre	CERFACS	France
Mutsumi	Aoyagi	U Kyushu	Japan
Mike	Ashworth	Daresbury	UK
Franck	Barbier	ANR	France
David	Barkai	Intel	US
Sanzio	Bassini	CINECA	Italy
Kyriakos	Baxevanidis	EU	EU
Pete	Beckman	ANL	US
Jean-Yves	Berthou	EDF	France
Taisuke	Boku	U of Tsukuba	Japan
Bertrand	Braunschweig	ANR	France
Franck	Cappello	INRIA	France
Barbara	Chapman	U of Houston	US
Alok	Choudhary	NWU	US
Iris	Christadler	LRZ	Germany
Guillaume	Colin de Verdière	CEA	France
Jack	Dongarra	U Tennessee	US
Sudip	Dosanjh	SNL	US
Hugo	Falter	ParTec	Germany
Teresa	Finchum	UTK	US
Fabrizio	Gagliardi	Microsoft	Italy
Alan	Gara	IBM	US
Luc	Giraud	CERFACS	France
Kostas	Glinos	EU	EU
Jean	Gonnord	CEA	France
Bill	Gropp	UIUC	US
Stefan	Heinzel	Max Planck DEISA	Germany
Mike	Herioux	Sandia	US
Yutaka	Ishikawa	U of Tokyo	Japan
Jean-Pascal	Jégu	Teratec	France
Zhong	Jin	CAS	China
Fred	Johnson	DOE	US
Andrew	Jones	NAG	UK
Laxmilkant	Kale	UIUC	US
David	Keyes	Columbia U.	US
Kimmo	Koski	CSC	Finland

Bill	Kramer	NCSA	US
Jesus	Labarta	BSC	Spain
Jean-Francois	Lavignon	Bull	France
Alain	Lichnewsky	Genci	France
Thomas	Lippert	Juelich	Germany
Bob	Lucas	ISI	US
Barney	MacCabe	ORNL	US
Satoshi	Matsuoka	TiTech	Japan
Paul	Messina	ANL	US
Peter	Michielse	NWO	NL
Bernd	Mohr	Juelich	Germany
Terry	Moore	UTK	US
Hervé	Mouren	Teratec	France
Wolfgang	Nagel	Dresden	Germany
Hiroshi	Nakashima	U Kyoto	Japan
Jean-Philippe	Nominé	CEA	France
Abani	Patra	NSF	US
Serge	Petiton	CNRS	France
Claude	Puech	INRIA	France
Tracy	Rafferty	UTK	US
Dan	Reed	Microsoft	US
Catherine	Rivière	GENCI	France
Christian	Saguez	Teratec	France
Mitsuhisa	Sato	U of Tsukuba	Japan
Ed	Seidel	NSF	US
John	Shalf	LBNL	US
David	Skinner	LBNL	US
Thomas	Sterling	LSU	US
Makoto	Taiji	Riken	Japan
Anne	Trefethen	Oxford	UK
Mateo	Valero	BSC	Spain
Jeffery	Vetter	ORNL	US
Vladimir	Voevodin	Moscow State U	Russia
Peg	Williams	Cray	US
Kathy	Yelick	LBNL	US