D4.2 Second intermediate report on enabling technologies

CONTRACT NO	EESI2 312478
INSTRUMENT	CSA (Support and Collaborative Action)
THEMATIC	INFRASTRUCTURE

Due date of deliverable: 31/05/2014

Actual submission date: 31/03/2015

Publication date: 31/03/2015

Start date of project: 1 September 2013 Duration: 30 months

Name of lead contractor for this deliverable: PRACE-BSC, Rosa M. Badia

Name of reviewers for this deliverable:

Abstract: This is the first intermediate report of EESI2 WP4 Enabling Technologies WGs. The document reports on the initial findings of each of the WGs and present initial recommendations.

Revision: 1.0

Project co-funded by the European Commission within the Seventh Framework Programme (FP7/2007-2013)		
Dissemination Level to be filled out		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Table of Contents

Glossary

Abbreviation / acronym	Description
DAG	Direct Acyclic Graph
EMWG	Exascale Mathematics Working Group
FMM	Fast Multipole Method

1. Executive Summary

This document is the second EESI2 intermediate report on enabling technologies, corresponding to WP4

With regard WG4.1, Numerical Analysis is an enabling technology that underlies all numerical computation in all application areas. The efficient and reliable implementation of these core numerical algorithms is crucial and essential if we want to realise the potential of future Exascale systems.

With regard WG 4.2 Scientific software engineering, software eco-system and programmability, tackles the development, operation and maintenance of software. The challenges in this area come from difference sources, between them the long life of codes or the lack of high-level programming environments.

WG4.3 focuses on disruptive technologies in enabling technologies. The activity in the group has focused on identifying how the disruptions can be identified, and afterwards identifying technologies that will help handling the disruptions.

WG4.4 focuses on establishing and maintaining a global network of contacts with vendors in the HPC industry and to leverage this network to investigate the state of the art and trends related to the Exascale roadmap in the HPC hardware and software industry.

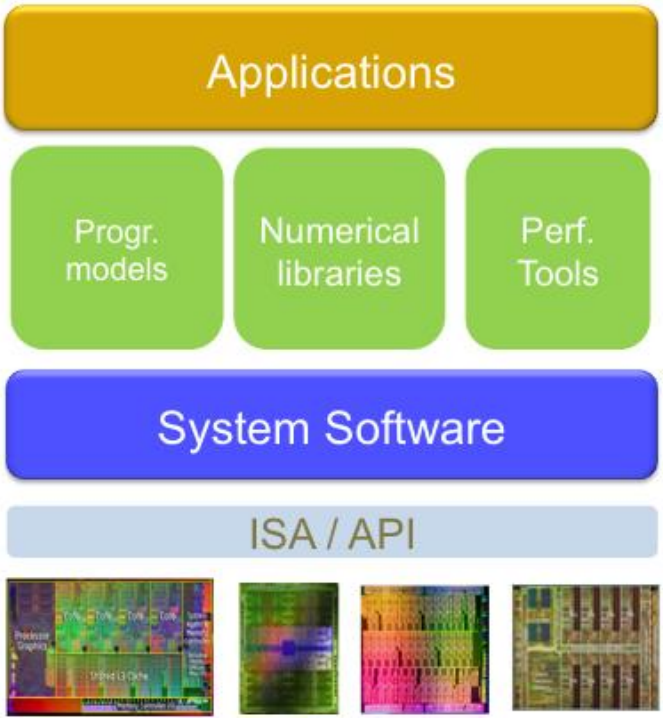
This deliverable presents an update with regard deliverable D4.1 for each of the WGs. The conclusion for the whole WGs is that in a year, there is no significant changes, although those that have been found out are reported in this document.

The deliverable also reviews the process of elaboration of three of the project recommendations:

- Algorithms for Communication and Data-Movement Avoidance
- High productivity programming models for Extreme Computing
- Software Engineering Methods for High-Performance Computing

2. WP4 Coverage

This is the second deliverable of WP4 "Enabling Technologies". This WP covers the different levels offered to the applications: programming models and performance analysis tools, numerical libraries, system software, and also the hardware level. Although organized in different WGs, a comprehensive view is necessary, since in order to achieve the exascale challenges a collaboration between all levels is necssary.



3. WG 4.1 Numerical Algorithms

3.1 Scientific context and tasks of the working group

Numerical Analysis is an enabling technology that underlies all numerical computation in all application areas. The efficient and reliable implementation of this core numerical algorithms is crucial and essential if we want to realise the potential of future Exascale systems.

Our basic building blocks involve dense matrix kernels, the most ubiquitous being the multiplication of two dense matrices, a kernel that should be designed to attain the peak performance of the machine. This software may be used directly on extremely large problems or may itself be a building block for the factorisation of large sparse matrices and the solution of the corresponding set of equations which may come from the discretisation of a continuous problem, for example the solution of a three-dimensional PDE. An alternative for solving large sparse systems is to use iterative methods where the main kernel is usually a sparse matrix-vector computation.

Very similar iterative methods can be used in the solution of large eigensystem problems where only a subset of eigenvalues and vectors are required. More recently there have been advances in combining direct and iterative methods in so-called hybrid methods that can again be designed to exploit the hierarchical structure of the evolving hardware. We also discuss software that sits further up the stack including problems in control and the major area of optimization, both linear and nonlinear. A major tool for decomposing large problems for all these approaches is graph and hypergraph partitioning which we also discuss, including the parallel implementation of software in this area. We then comment on aspects of structured and unstructured grid calculations and parallel random number generation, particularly in the context of Monte Carlo methods.

As for EESI-1 we have found it useful to break down our area into the following subtopics: Dense linear algebra, Graph partitioning, Sparse direct methods, Iterative methods, Eigenvalue problems, Optimization & control, Structured & unstructured grids and Monte Carlo. These are listed roughly in the order they appear on the software stack, i.e. Dense Linear Algebra is the building block on which most other areas depend, and so on. Due to their increase importance we have decided to add the topics of Tensors and Fast Multipole Methods as separate items.

Topics such as Dense and Sparse Linear Algebra that are used in all other areas have always had the highest pressure to develop efficient implementation. As a result when it comes to exascale progress is more advanced and algorithms are more mature in these areas. On the other hand addressing the remaining gaps is also a prime priority.

This section presents an update with regard deliverable D4.1 for this WG.

3.2 Origins of Expertise

The working group consists of a chair and vice-chair and eleven experts chosen to cover the domains of interest. Their names and area of expertise are listed below:

Name	Organization	Area of expertise
Andreas Grothey	University of Edinburgh	Continuous & Stochastic Optimization
Iain Duff	STFC	Sparse Linear Algebra
Jack Dongarra	University of Manchester	HPC, Numerical Linear Algebra

Mike Giles	University of Oxford	GPU, CFD/Finance, Grids, Monte Carlo
Thorsten Koch	Koch Zuse-Institut Berlin	Combinatorial Optimization
Peter Arbenz	ETH Zürich	Eigenvalues, Iterative Methods
Bo Kågström	Umeå University	Dense Linear Algebra
Julius Žilinskas	Vilnius University	Global Optimization, Meta-heuristics
Salvatore Filippone	Università di Roma "Tor Vergata"	Numerical Software
Luc Giraud	INRIA Bordeaux	Iterative Methods, Multipole Methods
Patrick Amestoy	ENSEEIHT-IRIT, Université de Toulouse	Direct Methods, Solvers
Karl Meerbergen	K.U. Leuven, ExaScience Lab	Eigenvalues, Tensors, Model reduction
Francois Pellegrini	LaBRI Bordeaux	Partitioning

3.3 Key Challenges

The key challenges described in the previous deliverable are still of foremost importance in order to achieve exascale scalability. Namely the areas

- Dense Linear Algebra
- Tensors
- Graph Partitioning
- Structured and unstructured grids
- Sparse direct methods
- Iterative methods
- Eigenvalue solvers
- Fast Multipole Method
- Optimization
- Uncertainty Quantification

Approaches such as maximising useful calculations per memory access, synchronization avoidance, dynamic scheduling and mixed arithmetic calculations remain common paradigms to be explored.

The underlying challenge across the whole area is the observation that with millions of cores computation is relatively cheap while memory access and communication are becoming increasingly expensive.

3.4 1.2 Developments of the past year

Progress on the key challenges is made on various fronts although as is the nature of research it is difficult to measure progress between one year and the next. It is likely that it will take a few years to recognize the impact that has been made.

On notable development has been a marked increase focus on Big Data and Data Science both in terms of research undertaken and funding calls. The focus of research in this are is (among others) on

- First order methods for optimization
- Compact representations/low-rank approximations/clustering
- Hierarchical algorithms
- Compressed sensing
- Randomised algorithms

We welcome this development. Although “Big Data” is not the only research challenge, many of the challenges involved are the same as for exaflop computing (this has indeed been already pointed out in the Deliverable 4.1).

We welcome the Horizon2020 call on “New Mathematics for Exascale” . This call has received strong interest from the scientific community including Numerical Agorithms and more than 80 applications have been received.

3.5 Gap Analysis

The gap analysis described in the previous deliverable is still up to date.

3.6 Project recommendation "Algorithms for Communication and Data-Movement Avoidance"

This recommendation aims at addressing one of the major challenges in high performance computing, the fact that there is an exponentially increasing gap between the time required to compute floating point operations and the time required to move data between different levels of the memory hierarchy or between different computing units. It takes as an example the recent development of communication avoiding algorithms in numerical linear algebra that aim at addressing this challenge. These algorithms are able to minimize data movement as well as the number of communication and synchronization instances in extreme computing, and as a consequence they also reduce significantly energy consumption.

From this example, the recommendation motivates the need for developing new numerical algorithms that are able to address this major communication problem and go far beyond overlapping communication with computation. These algorithms should be developed for numerical linear algebra, but also beyond, for all critical stages of computationally intensive applications, e.g. mesh generation algorithms, parallel in time methods.

This recommendation was done in collaboration with external experts (Laura Grigori, Hatem Ltaief, Ulrich Ruede).

4. WG 4.2: Scientific software engineering, software eco-system and programmability

This working group focuses on methods, processes, tools, and support structures required to create robust, correct, efficient, and maintainable code under economic constraints. In an adaptation of ISO/IEC/IEEE 24765:2010, we define "[scientific] software engineering as the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to [scientific] software". Our target software includes mostly highly scalable simulation codes but also other data-intensive applications such as graph analysis and is developed in both academic and industrial settings.

4.1 Members and their expertise

Name	Organization	Area of expertise
Felix Wolf (chair)	German Research School for Simulation Sciences, Germany	Parallel programming tools (performance analysis & modeling, parallelism discovery), parallel programming models
Matthias Mueller (co-chair)	RWTH Aachen University, Germany	Parallel programming models, correctness checking, runtime error detection, performance analysis, energy efficiency
Mike Ashworth	Science and Technology Facilities Council, UK	High-performance applications development, numerical algorithms, benchmarking, languages, software tools and environments
Achim Basermann	DLR (national aeronautics and space research centre), Germany	Parallel numerical algorithms and data structures, development of parallel applications, parallelization technology for modern computer architectures, Python for HPC
Vincent Bergeaud	CEA, France	Software engineering in scientific simulation, parallel programming and uncertainty analysis
David Brayford	Leibniz Computing Centre, Germany	Software design, software architecture, software development, low level (CPU, embedded, SIMD, drivers), high level (applications)
Jim Cownie	Intel, UK:	Parallel programming, Occam, MPI, OpenMP, Cilk, Fortran, UPC, parallel debugging, message passing hardware, computer architecture from the SW viewpoint

Alessandro Curioni	IBM Zurich, Switzerland	Scientific computing, parallel programming, computational sciences, algorithm re-engineering for massive scaleout, HPC applications design and maintenance
Torsten Hoefler,:	ETH Zurich, Switzerland	Parallel programming models, parallel library development and stacking, performance modeling, performance portability, message passing, RMA, scalable algorithms and runtime systems
Horst Lichter	RWTH Aachen University, Germany	Software architectures, metrics and measurement, requirements engineering, model-based development, test and validation, software development processes
Andrea Walther	University of Paderborn, Germany	Algorithmic differentiation (AD) including the development of an open-source AD tool, nonlinear optimization, high performance computing for the simulation of complex systems and their optimization

4.2 Key challenges

The key challenges described in the previous deliverable are still up to date.

4.3 Developments of the past year

Although no new versions of the programming interfaces discussed in the corresponding section of the previous deliverable were published during the past year, several interesting developments occurred in the programming models area. One is the quick deployment and adoption of MPI-3's extensions. Especially the new RMA programming model, known from UPC and Fortran 2008, has been gaining attention and fast implementations are available. Furthermore, task-based programming models are looming at the horizon and may be adopted at large scale. All these affect software development methodologies in that we require new tools and development environments for these new programming models.

Intensive research was also done in the area of domain-specific languages (DSLs), e.g. for stencil codes and graph algorithms. It was shown that higher-level source does not necessarily contradict performance and in fact can improve performance and portability – even in real applications.

Another critical development of the past year was the growing realization that Moore's law will likely come to an end between 2020 and 2025, as feature sizes beyond 5 nm seem neither physically nor economically feasible. This will have a profound impact on the entire HPC landscape, whose precise implications are hard to predict. With increased integration density no longer available to create added value, design may emerge as the primary vehicle for generating profit. The resulting diversification would cause a huge challenge for the HPC software industry. It is unclear, however, how many new designs the HPC market can absorb. At some more distant point in the future, hardware may stabilize and we may see a shift of focus and resources away from hardware to software. With hardware becoming more standardized, investments in HPC software would become more attractive. Either way, further fostering Europe's strong position in HPC software seems to be an advisable proposition.

4.4 Gap Analysis

The gap analysis described in the previous deliverable is still up to date.

4.5 Proposition of R&D program for 2014 and beyond

Although the WG recognizes the support that will be provided through the recent FET HPC call in H2020, the proposed R&D program remains unchanged. However, the WG created a number of more detailed recommendations outlined in the section below.

4.6 Further recommendations

In addition to the recommendations related to training, education, networks of excellence, centers of excellence, and interdisciplinary support structures laid down in the previous deliverable, the WG formulated a number of more detailed recommendations concerning key aspects of Exascale software engineering. These recommendations are the result of in-depth studies conducted by WG experts during the past year.

Co-designing applications and the system is a powerful technique to ensure early and sustained productivity as well as good system design. In their early phases, such co-designs often rest on back-of-the-envelope (BOE) calculations. In general, such calculations allow problems in applications to be detected early on and their severity to be determined years before the machine is installed or the first prototype becomes available. On the system side, BOE calculations allow designers to adjust system parameters to target applications, for example, they can be used to determine the required bytes-to-flop ratio of memory, network, or even the file system. However, even such BOE calculations are very time consuming and also error prone. Especially projecting applications requirements for large workloads still poses a challenge. Therefore, tools are needed to automate such projections, making the co-design process more reliable and efficient.

Accelerators are widely used in large-scale systems, i.e., the top 10 systems of TOP500. At smaller scale, systems without accelerators dominate. Most vendors have an accelerator-based roadmap towards Exascale. Independent of the specific vendor implementations, accelerators share properties (sometimes known under vendor specific names) that are also envisioned for future general purpose CPUs (many cores, many threads with SMT, longer vector registers). Currently a broad range of programming models exists to program accelerators (CUDA, OpenCL, OpenACC, OpenMP 4.0). They all evolve at different speeds and are supported by different groups. While this is a normal situation for an evaluation phase of new hardware architectures, there clearly is a need for better support for programming accelerators in the long run. Otherwise the impact made at Exascale level will not trickle down to smaller scale. We therefore recommend to strengthen the EU participation in relevant vendor-neutral standardization efforts (e.g., OpenCL, OpenMP). In addition research to provide abstraction levels independent of the specific programming method and hardware implementation should be supported.

Domain Specific Languages (DSL) targeting specific application areas (e.g., climate, engineering, materials) offer a high-level abstraction enabling application development to be protected and insulated from architectural issues at Exascale. Development of applications using DSLs should take place using co-design principles with teams comprising science domain experts and computational technologists. With multi-disciplinary teams there is a great advantage in separation of concerns so that the science code can be developed separately from the computer science aspects. DSLs can do this, encapsulating the growing complexity of code required for multiple levels of parallelism and targeting a range of multi-petascale and Exascale architectures.

Software Development Processes (SDP) heavily impact the quality of the developed software. As there are several challenges (e.g., developers are domain experts not software engineers, high developer fluctuation in scientific projects) the SDP needs to be customized in order to effectively support Exascale development projects. Furthermore it is not known which important technical characteristics of Exascale development projects should be supported by a dedicated Exascale SDP. Hence, existing best process practices in developing Exascale applications have to be collected and analysed to define a lightweight SDP framework. The process framework should be extensible and customizable to the specific needs of Exascale development projects.

Software Product Lines (SPL) are sets of software systems for a particular market or domain developed from a common set of core assets in a prescribed way. In short, a SPL potentially decreases development time and cost, improves productivity, and increases the quality of software. Although SPLs have been successfully implemented in several domains, it is an open question whether existing product line techniques can be reused or adopted in the context of HPC and Exascale computing. Hence, intensive research is needed regarding e.g., which technologies support the development of highly reusable components or which kind of software architectures are needed to build Exascale applications on top of a reusable platform.

Static program analysis is a means to prove whether the behavior of a program matches its specification. **Software testing** is a dynamic analysis process to detect errors or to identify the difference between actual and expected result or behaviour. Behaviour may cover non-functional properties like performance and scalability, which are especially relevant for HPC. The overall goal is measuring software quality. There are challenges both for the static analysis and the dynamic test of highly scalable codes. While the low cost of static analysis make it extremely powerful, code complexity and the lack of runtime information make the static verification of code properties very hard. On the other hand, the infrastructure needed for dynamic debugging of high numbers of processes is very expensive. Effective software testing for highly parallel software can make software development more efficient, productive, and stable. Another impact is cost reduction through cheaper regression tests, an advantage especially important for industry. Topics of future research should therefore include innovative verification strategies, advanced tool support for quality tests, and new test strategies for extremely parallel codes.

4.7 Project recommendation "High productivity programming models for Extreme Computing"

The motivation for this recommendation is based on one hand, on the evolution of the architectures for exascale (with heterogeneous nodes with GPUS or MICs, big-little architectures, etc) and on the other, on the requirements of large scalability to large number of nodes. While productivity and portability has been an important issue in the past, with these new architectures is now a must.

While the state of the art on programming models presents some approaches that can be applied towards this direction of high productivity in scientific codes, more is needed to enable the requirements of the exascale architectures and to motivate the application developer to port or develop their codes with these new programming models.

The recommendation "High Productivity Programming Models" is born with this motivation and it is described in detail in the document "D7.2 EESI2 Second Annual Report 2014 Update Vision & Recommendations", and it is also attached as an annex to this deliverable.

The specific objectives of the recommendation are to explore:

- New approaches in task-based asynchronous execution models, being able to hide the details of the HW platform
- Automatic exploitation of parallelism enabling scalability at very large number of nodes

- Communication hiding programming in heterogeneous architectures
- Embedded Domain Specific Languages to improve productivity of HPC heterogeneous environments (prototyping programming languages or scripting languages)
- Tools for automatized detection of data- and task-dependencies for multi-threaded task based programming
- Programming environments where it is possible, for example, to interplay between compute platforms and data-bases, to design and execute workflows for high-throughput computing or multi-stage computational refinements.
- Intelligent runtimes that perform efficient resource management, exploit data locality, automatic load balancing, and that are energy aware, between other features.

4.8 Project recommendation "Software Engineering Methods for High-Performance Computing"

The decision to elaborate this recommendation was motivated by a lack of attention to software engineering issues in the HPC community. So far, the needs of HPC were mostly analyzed from the perspective of domain science, programming techniques, or numerical analysis. Although together offering already a very broad perspective, they generally do not encompass people and processes. Other software domains have developed proven software engineering approaches, whose suitability and adaptability for HPC needs to be investigated. Moreover, the special requirements of HPC like performance and scalability need to be addressed from a software engineering perspective. Given the high development cost of HPC software and the assets of existing software infrastructure, neglecting these aspects could lead to suboptimal software quality and would pose the risk of squandering huge saving potentials.

The recommendations were derived from the experience and knowledge of WG experts, from the review of relevant literature and through reasoning. The recommendation document went through several internal revision cycles with feedback from WG experts before being submitted to the coordinator.

5. Working Group 4.3 “Disruptive technologies”

The objective of this working group is to tackle specific disruptive technologies in the field of software eco systems and numerical analysis. Finding disruptive technologies is not an easy task, since it is only known that a given technology is disruptive when it is already main stream. However, the WG has been able to devise some conclusions.

5.1 Members and their expertise

Due to the nature of the group, its formation has been slightly slower than others. Right now is composed by the following members:

Name	Organization	Area of expertise
Iain Duff	STFC	Sparse Linear Algebra
Serge Gratton	ENSEEIHT	Optimization, data assimilation
Jesus Labarta	BSC	Programming models, Performance analysis, System software
Mike Giles	Oxford	CFD, MC, Financial Maths, GPUs
Hatem Ltaief	KAUST	Sparse Linear Algebra, GPU and heterogeneous
Rosa M Badia	BSC	Programming models, Heterogeneous programming, distributed computing

5.2 Identified disruptions and technologies that enable to cope with them

In deliverable 4.1 a set of disruptions and technologies originate disruptions. The disruptions identified continue to be the same list, since no major changes have succeeded in such a short time. The identified disruptions were:

1. Variability in a dynamic world: in resource performance, resource availability
2. Abstraction: from low level device features to high level specification
3. Asynchrony
4. Bottleneck shift from computing to data transfer
5. Imprecise computations
6. Power constraints

With regard technologies that originate disruptions, the following were identified:

- Hardware techonogies
  - Memories: PCM, memristor, NVRAMs
  - Packaging
    - 2.5D
    - 3D stacking
    - Optic communication between devices
  - Multicores, manycores, accelerators & SoC
  - Storage
    - SSDS
- Software technologies
  - Virtualization
  - No SQL: Key-value storage
- New Paradigms
  - Quantum Computing
  - Bio-inspired

5.3 Reacting technologies

To cope with these disruptions and new technologies, the WG has identified some reacting technologies, which appear as a result to them. The list of reacting technologies is the following:

- Load balancing
- Asynchronous programming models and system software
- Communication reducing, communication hiding, synchronization reducing algorithms
- Power aware schedulers
- Mixed precision computation, low rank compression
- Hybrid algorithms and solvers
- Stochastic PDEs
- New techniques: tensor calculus, novel algebras, stochastic programming
- New algorithms: Chaotic relaxation, contour integration, Monte-Carlo techniques, vectorization
- Hierarchy (i.e., nesting) – it is a need for new algorithms: Fast multipole methods, dense eigenvalue, H-matrices

Most of them were already described in deliverable 4.1. The new element added to the list is the ability to express new algorithms (Fast multipole methods, dense eigenvalue, H-matrices) with ***hierarchy***. The concept of hierarchy enables the algorithm to be organized in different levels, or even recursively. For example, this can be done with some programming models with task nesting (OpenMP tasks or OmpSs). The hierarchical implementations will improve aspects such as synchronization and communication (vertical and horizontal), data reuse, resiliency (local failures), power, etc. One good and concrete example in this area is the H-matrices, which are a compressed format of dense matrices using low rank representations. Not all matrices can be H-matricized but many from engineering applications can.

6. Working Group 4.4 “Hardware and Software Vendors”

One of the main objectives of this working group is to establish and maintain a global network of contacts with vendors in the HPC industry and to leverage this network to investigate the state of the art and trends related to the Exascale roadmap in the HPC hardware and software industry. A group of 13 experts, mostly from the HPC hardware industry, have agreed to contribute to this working group and to share their insights into the industries R&D roadmaps:

Expert	Email	Organization	Position
David Lecomber	david@allinea.com	Allinea	CTO and founder
Chris Adeniyi-Jones	Chris.Adeniyi-Jones@arm.com	ARM	R&D engineer
Jean-Pierre Panziera	Jean-Pierre.Panziera@bull.net	Bull	Director of Performance Engineering
Alex Ramirez	aramirez@ac.upc.edu	BSC	Computer Architecture Research Manager
Francois Bodin	Francois.Bodin@caps-entreprise.com	Caps Entreprise	CTO
Giampetro Tecchiolli	giampietro.tecchiolli@eurotech.com	Eurotech	CEO/CTO
Ulrich Brüning	ulrich.bruening@ziti.uni-heidelberg.de	EXToLL	Director
Luigi Brochard	luigi.brochard@fr.ibm.com	IBM	Distinguished Engineer
Karl Solchenbach	karl.solchenbach@intel.com	Intel	Director Exascale Labs Europe
Axel Koehler	akoehler@nvidia.com	nVIDIA	Senior Solution Architect HPC
Matthias Müller	mueller@rz.rwth-aachen.de	RWTH Aachen	Head of Compute Center
Kai Dupke	kdupke@suse.com	SUSE	Senior Product Manager Server
Malcom Muggeridge	Malcolm_Muggeridge@xyratex.com	Xyratex	VP of Emerging Technologies

In the previous Deliverable D4.1, we identified a list of challenges that need to be tackled on the way to Exascale. In terms of hardware, the most pressing challenges were energy efficiency, the capacity, and bandwidth of the memory and storage subsystems, as well as reliability and resilience. This deliverable will give an overview on actual product announcements of the hardware industry and assess whether they address these challenges adequately.

6.1 Hardware challenges

6.1.1 Energy efficiency, Power Wall, Power Density

The power consumption of HPC systems is largely dominated by CPUs and GPUs, and to a lesser extent by memory, network, and storage. However, the energy efficiency of an HPC system is not only determined by its hardware but also by the software running on it and how efficiently it exploits the hardware. Hence, energy efficiency is not only a matter of the power consumption of individual components but also a matter of the balance of their performance: a system that is built from low power components but only runs at 1% of its theoretical peak performance due to poor network performance can hardly be considered as energy efficient.

As Moore's law still holds thanks to ever finer semiconductor manufacturing processes (14nm being the state-of-the-art in 2014 and 10nm being predicted for 2016), future microprocessors automatically become more energy efficient. As leakage currents fall with smaller transistor sizes, they deliver more compute performance within the same power envelope of their predecessors. However, as it becomes harder and harder to put these additional transistors to good use, the degree of parallelism within a single CPU or GPU increases steadily. They feature more cores and employ ever wider instruction sets that require parallel code even at the instruction level. Consequently, it becomes more and more difficult for programmers to exploit these systems efficiently.

More and more system integrators offer not only air-cooled systems but also direct-liquid cooled or immersion cooled solutions. As these approaches allow for much more efficient waste heat removal, they facility high density form factors and therefore increase the power density of HPC systems.

6.1.2 CPUs, GPUs, and Accelerators

For HPC systems, the most established providers for CPUs are Intel, IBM, and AMD. As IBM sold its x86 business to Lenovo, they concentrate on its Power architecture in terms of HPC and also founded the OpenPOWER foundation that will develop the Power architecture in the future. ARM started moving from the mobile market to the server market and recently introduced its 64bit architecture ARMv8 that seems to be suitable for HPC as well: first OEMs like AMD, AppliedMicro, and Cavium announced ARMv8-based server SOCs that feature wide memory buses, high-bandwidth interconnects, and 3rd generation PCIExpress for extension cards like Infiniband. AMD also offers HPC-grade GPUs and is the main contender of NVIDIA in this market. Besides its Xeon line of server CPUs, Intel also offers and further develops its accelerator product line Xeon Phi.

The CPU roadmaps of the established vendors look very similar in terms of technology: the number of cores keeps on increasing, DDR4 becomes the standard memory interface, the cache hierarchies become deeper, the instruction sets wider.

The coupling between CPUs and GPUs becomes closer, with faster busses, uniform memory access, and cache-coherency: technologies like AMD's Heterogeneous System Architecture (HSA) and OpenPOWER's Coherent Accelerator Processor Interface (CAPI) may have the potential to simplify the programming models for heterogeneous architectures by removing the need for explicit memory transfers between the host CPU and accelerators. However, Intel has announced no such technology for their Xeon and Xeon Phi processors.

As the coupling of CPUs and accelerators becomes tighter, the next logical step seems to be moving the network interconnect closer to the CPU as well. All major CPU vendors also have high-performance network technology available in their portfolio: Intel has Omni-Path (derived from Cray's Aries interconnect), AMD the SeaMicro Freedom Fabric, and the OpenPower consortium has

Mellanox with its Infiniband technology as a member. In the long term, the interconnect may very well move to the CPU die, allowing for higher bandwidth and lower latency.

6.1.3 Memory/Storage capacity, packaging, bandwidth

The advancement due to Moore's law also applies to DRAM technology. Thanks to shrinking manufacturing processes next generation DRAM will operate at higher frequencies and lower voltages which eventually translates to higher bandwidth and lower energy consumption. New technologies like 3D-stacked memory where multiple layers of DRAM are stacked and connected using through-silicon vias, allow for denser packaging. Additionally, developments like High-Bandwidth Memory (HBM) that put stacked DRAM on the same package as the CPU will eventually also allow for significantly higher memory bandwidth and closer coupling of memory and CPU. New protocols like DDR4 and Hybrid Memory Cube (HMC) also contribute towards higher memory bandwidths. Upcoming CPU generations will provide memory bandwidths in the order of 250 GB/s which will help to keep the balance between CPU and memory performance at least at current levels.

In terms of storage, spinning magnetic disks remain the "work horses" for data-intensive applications. However, the capacity and density growth we have seen in the previous decade seems to have come to a stop for a couple of years now. Where bandwidth and latency are important, non-volatile memory technologies like NAND flash already provide the better price/performance ratio and continue to close in on harddisks in terms of price per GB. New NVRAM technologies like Magnetoresistive RAM (MRAM) and Phase-change RAM (PRAM) have supposedly been developed by multiple vendors but no actual products have been announced yet. The same holds true for storage based on memristor technology.

6.1.4 Reliability/Resilience

As the number of individual components in current and future HPC systems continues to grow, the failure rates of these systems as a whole will increase dramatically. Therefore, total breakdowns and transient errors of each single component will have to be anticipated and accommodated at the hardware level. While everybody seems to talk about reliability and resilience features, actual products that implement such seem still to be missing and do not appear on the hardware roadmaps of the mainstream hardware vendors. Although being recognized as an important topic, research in this field only appears to be taking place in the academic sphere.

6.2 WG4.4 summary

In terms of new hardware, no revolution appears to be taking place. The gains in performance and power efficiency to be expected in the foreseeable future are mostly due to advancements in manufacturing processes. The general trend is tighter coupling of individual components, be it CPUs, accelerators, memory, or network interconnects. There is no new technology on the horizon that has the potential to be a game-changer. However, at the same time it also does not seem that there were any game-stoppers on the way to Exascale. The current hardware roadmaps of the established vendors may pave that path. The only topics of concern appear to be hardware reliability and resiliency. While academic research suggests that these topics will have to be addressed, hardware vendors appear to be ignoring them for the time being.

7. Conclusions

This document is the second intermediate report of the EESI2 WP4 Enabling Technologies. The WP is organized in four WGs: Numerical analysis; Scientific software engineering, software eco-system and programmability; Disruptive technologies; and Hardware and operating software vendors.

The deliverable presents an update of the first report and also reviews the process of the project recommendations derived from this WP.

Between the topics that the experts highlight as challenges, we find:

- Approaches such as maximising useful calculations per memory access, synchronization avoidance, dynamic scheduling and mixed arithmetic
- The fact that millions of cores computation is relatively cheap while memory access and communication are becoming increasingly expensive
- Research in the area of domain-specific languages (DSLs)
- Availability of the notion of hierarchy in programming models (i.e., nesting) – it is a need for new algorithms: Fast multipole methods, dense eigenvalue, H-matrices
- Reliability and resiliency

In summary, however, this report has minor updates since the period between both reports D4.1 and D4.2 has been relatively short.

