EESI2 First Intermediate Reports

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

Due date of deliverable: September 2013

Actual submission date: September 2013

Publication date:

Start date of project: 1 September 2012 Duration: 30 months

Name of lead contractor for this deliverable: TOTAL SA, Philippe RICOUX

Revision: V1.0

Introduction

The joined documents are the first intermediate reports of EESI2 initiative after one year work.
These reports present the gap analysis of recent actions on developments for Exascale applications.
An important focus was put on research of potential disruptive technologies.
End of May 2013, during a large technical workshop (D8.3), involving the experts (around 100) of all the working groups, all important results, findings, conclusions and recommendations were presented and discussed.

An update vision and roadmap of Exascale implementation is presented (D7.1) based on the EESI2 working-groups reports (D3.1, D4.1, D5.1, D6.1) and the international workshops conclusions (D2.1). The project proposes eight recommendations for eight R&D programs, five of them are absolutely critical: Ultra scalable algorithms, Resilience, Big Data, Couplers, High productivity programming models. The three others are of high priority: Mini apps, Software Engineering Methods for High-Performance Computing and Verification, Validation and Uncertainty Quantification.

In addition of these 5 crucial recommendations, this introduction presents one highlight of each report:

- ✓ D7.1 (Vision): Exascale imposes to do something different and differently, imperative breakthrough on hierarchical algorithms which reduce communications and tasks synchronizations
- ✓ D3.1 (Industrial applications): large petaflop computers are operational now in several companies; big data management is already a reality. A breakthrough on multi-physic coupling computing is necessary.
- ✓ D4.1 (Enabling Technologies): breakthrough(s) on Numerical Analysis , new paradigms for parallel programming, evolution of memory technologies, merge efforts with ETP4HPC
- ✓ D5.1 (Cross cutting): Big Data for extreme computing, Resilience, Uncertainties and Validation, are key issues on which working groups concluded to propose urgent recommendations (see D7.1)
- ✓ D6.1 (Exascale Code Maturity): First steps on methodology and “Maturity test” centre model.
- ✓ D2.1 (Education and Strategic Coordination): Organization of a series of three International workshops entitled Big Data and Extreme Computing (BDEC) that will be organized in USA (April 2013), Japan (February 2014) and Europe (December 2014).
- ✓ D8.1, 2, 3 (Dissemination): Dissemination plan, New Web site, Technical Meeting report

And Finally,

- ✓ D1.1 (Management): First Overview of EESI2 project Management.

All EESI information and documents are available on the new web-site: http://www.eesi-project.eu

Deliverable D7.1 First intermediate report on EESI2 Exascale vision, roadmap and recommendations

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

Due date of deliverable:	September 2013
Actual submission date:	September 2013
Start date of project: 1 September 2012	Duration: 30 months
Name of lead contractor for this deliverable: **Total – Philippe Ricoux** Name of reviewers for this deliverable: **Total – Jean-Yves Berthou, Thierry Bidot**	
Abstract: **The report provides a vision and a set of recommendations issued from the first year project**	
Revision: V1	

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

1. Introduction

The EESI2 initiative concerns recommendations for Europe on R&D projects for the implementation of efficient Exascale applications and the development of supporting software environments.
A first European initiative, EES1, developed a unique network of academic and industrial experts delivered a first worldwide cartography of developments in this area, proposed a global Exascale software stack roadmap, an Exascale application roadmap, including fundamental research and industrial grand challenge applications. EESI1 finally proposed general recommendations on the Exascale roadmap implementation agenda.

This EESI D5.6 Final report (end 2011) on roadmap and general recommendations development is available on the website http://www.eesiproject.eu/pages/menu/homepage.php
EESI experts also largely participated to the new (Oct-2012) PRACE Scientific Case Update for High Performance Computing in Europe (2012 - 2020), which is available on: http://www.prace-ri.eu/PRACE-The-Scientific-Case-for-HPC .

The EESI2 project is a step further and deeper in the ***periodic update*** *of* ***vision, roadmap and recommendations*** for Europe on Exascale effective implementation: state of art of world development, detection of ***disruptive technologies,*** addressing as well cross cutting issues, numerical processing and software engineering, with a *gap analysis*, orientations and monitoring of R&D actions and training.

One of the EESI2 main objectives is to provide recommendations on strategic European actions with a particular focus on software key issues improvement, cross cutting issues advances, and gap analysis. These periodic recommendations should be synchronized with EC R&D projects decision agenda.

This report delivers a vision and some recommendations after the first year of the EESI2 project.

2. Vision

A set of technical issues that are on the critical path for Exascale computing, including Extreme Computing and Big Data, has been identified by EESI1:

At the level of simulation environment:

- **Unified Simulation Framework** and associated services: CAD, mesh generation, data setting tools, computational scheme editing aids, visualization, etc.
- **Multi-physics simulation**: establishment of standard coupling interfaces and software tools, mixing legacy and new generation codes
- Common (jointly developed) **mesh-generation tool, automatic and adaptive** meshing, highly parallel
- Standardized **efficient parallel IO and data management** (sorting memory for fast access, allocating new memory as needed in smaller chunks, treat parts of memory that are rarely/never needed based on heuristic algorithms, …)

At the level of codes/applications:

- **New numerical methods, algorithms**, solvers/libraries, improved efficiency
- Coupling between **stochastic and deterministic methods**: Numerical scheme involving Stochastic HPC computing for uncertainty and risk quantification
- Meshless methods and particle simulation
- **Scalable program**, strong and weak scalability, load balancing, **fault-tolerance** techniques, multi-level parallelism (issues identified with multi-core with reduced memory bandwidth per core , **reducing drastically communications**, Efficient parallel IO)
- Development of **standards programming models** (MPI, OpenMP, C++, Fortran, …) handling **multi-level parallelism** and heterogeneous architecture

The objective of EESI2 is to propose, based on this first list, some concrete R&D actions which may allow removing some of the technological obstacles to Exascale computing.

This report is a first attempt for achieving this goal after a one year work done by 100 experts organised in small specialized groups and sharing activities in a large two days Technical Meeting in Le Tremblay (France) on 28th and 29th May 2013 (see report deliverable EESI2 D8.3 “Report on First Technical Meeting”).

The EESI2 Vision

The vision which is presented here is also inspired by some worldwide recent installations in Europe, Asia and in USA of 10 (and more) Petaflops computers and by the feedback of several applications and tests running on full configurations of these systems.

These tests have shown the ***extreme difficulty to get some acceptable results*** in term of performance on these computers. In particular the following points appear to be critical:

- Resilience
- Error propagation
- Reproductibility
- Data transfert, communication
- Task synchronization

As a consequence**,** Exascale applicative softwares appear to be a very difficult challenge and most worldwide experts consider that this challenge will not be solved **with existing algorithms.**

Therefore there is a necessary collective awareness to develop some R&D programs radically different from the usual ones, improving R&D technologies and algorithms**.** The following recommendations must be understood in this context.

What appears presently, shared by US, Japanese and European experts, is that:

- *Exascale technology will trickle down to every scale (architecture system as well physics and time)*
- *Exascale cannot be justified only if we are just planning to do the usual thing but bigger*
- *Exascale machines will be useless without algorithms that use their specific features*
- *Exascale imposes to do something different and differently*

The following points are on the critical path to Exascale Computing::

- *The use of hierarchical algorithms which reduce communications and tasks synchronizations*
- *The use of multi-physics methods which do not need or minimize data transfers and include multi scaling and parallel space-time methods*
- *The reshaping of operating systems and management tools such as MPI and OpenMP and mesh generation tools to the new developed algorithms*
- *The use of in situ data processing*

In term of organization, the EESI experts agree that ***multi disciplinary research*** teams approach enable the emergence of significant progress toward the implementation of Exascale applications. The best coupling of Architecture, Algorithm and Application (AAA) is the challenge of efficient Exascale software. That leads *to increase international collaboration*, international working teams.
But, even if all agree on these points*,* there is an on going debate on co-design centre structure. EESI2 will go on deeper on this item. The related EESI2 working group just begins its works and will give recommendation within the next deliverable in 2014. As a first recommendation, EESI claims that co-design centers/centers of excellence should conciliates scientific multi-disciplinarity, international dimension, critical mass of researchers working at the same place, the balance of vertical (specialty) and horizontal (transverse) scientific domain and the need to do things differently.

3. Recommendations

In relation with this EESI2 Exascale vision, EESI2 initiative proposes ***eight urgent recommendations*** to do "something different and differently" in Europe or internationally in order to tackle key issues of the Exascale vision.

These recommendations have been divided in two groups.

The first group includes five priorities which are considered as absolutely critical while the second group includes three of high priority.

The first group comprises the following issues:

- Ultra scalable algorithms
- Resilience
- Big data
- Couplers
- High productivity programming models

The second group is composed of:

- Mini apps
- Software Engineering Methods for High-Performance Computing

Verification, Validation and Uncertainty Quantification

The following chapters present the main objectives, type of actions and funding for each of the eight recommendations.

Each recommendation is described as following:

- Scientific perimeter of the recommendation
- Motivations of the recommendation
- Impact of the recommendation
- Level of innovation of the recommendation
- Coarse tasks of the program of the recommendation
- Type of recommendation

These recommendations, and at least the five first, should be examined for EC R&D programs funding as soon as possible.

3.1 Recommendation N°1 - Ultra scalable algorithms

Scientific perimeter

The domain of numerical algorithms 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 the potential of future Exascale systems is to be realized.

There are several kinds of challenges related to Exascale computing related to **numerical algorithms**

- some arise directly because of the architectural features in Exascale computers, such as hierarchical and heterogeneous processor layout
- some arise because of the increasing need for energy aware algorithms. Communication and memory accesses tend to be power hungry, an increase in computational speed is often in direct contrast to the low energy requirement
- others arise because of the increasing complexity of computations that application scientists and engineers

Algorithms must be designed to match the complexity of the Exascale architecture so, in most cases, a hierarchical algorithm will be required probably using combined communication protocols. Advances cannot be achieved by only restructuring existing codes but by redesigning numerical algorithms, and even developing new algorithms by reconsidering the mathematics behind, the numerical kernels, in agreement with computational platforms.

Motivations

The scientific perimeter concerns numerical analysis and computer science, but also some aspect of probability theory as far as fault tolerance and stochastic algorithms are concerned. So there is strong need for persons well trained in these three domains. The collaboration between experts in these domains must be promoted.

The project aims making simultaneous progress in:

- Identifying news application areas for Exascale computing in high dimensional multiscale/multiphysics modeling. Mathematical objects arising from these applications frequently possess different structural properties to the traditional application areas of numerical algorithms. A severe redesign, often on a quite fundamental level, is needed to deal with these structures efficiently and to incorporate a good handling system optimization, data uncertainties management and sensitivity analysis
- Find new algorithm that are better adapted to Exascale architecture. Those algorithms should push to their limit the combination of ideas that are retrospectively considered as breakthrough (such as hierarchical algorithms, domain decomposition, preconditioning, etc.) so as to better cope with the upcoming machine particularities. But significant effort as to be produced to find new mathematical ideas that will find the best tradeoff between speed, accuracy and reproducibility.
- Improve the appropriateness of algorithms to hardware characteristics by considering e.g. fault tolerance, energy aware algorithms, hardware aware data structures

Impact (scientific, technological, societal & environmental)

The simulation of complex systems is heavily based on the availability of algorithms adapted to massively parallel machine. This proposal is an important step to give to the application communities the fool they need to produce a sustainable research effort. Impact is expected on all fields where modelling and optimizing is the key: industrial design, climate modelling, energy production, etc..

Level of innovation

The project innovation is on two levels:

- Maturation of existing algorithms and software tools to optimize them for Exascale computing
- Development of new numerical techniques for handling very large problems

Meso scale detailed description of the proposal

SP1 – Improvement of algorithms

- ➔ Algorithmic advances. As much as possible, all the algorithms may have to be hierarchical, to avoid communication, use sophisticated task scheduling.
- ➔ Mathematical analysis of algorithms. It will be essential to improve the theoretical tool we use to analyse algorithms that use some part of randomness, truncation, asynchronicity. Error analysis, convergence theory in has to be considered in functional and probabilistic spaces.
- ➔ Validation: available software tests do not take into account the specific needs of Exascale computing. The proposal is to improve those tools to introduce new problem collections that will provide the suitable test-bed for new architectures and algorithms.

SP2 – Development of new techniques

- ➔ A key point of this program will be the development of new algorithms, implementing mathematical ideas that may be proved successful, because they are well adapted to emerging architectures. For instance, data compression in algorithms of model reduction ideas as they appear now in direct solvers are promising areas that need further exploration. Randomize algorithm is also an area where research has to be strengthen. The use of dynamic accuracy in codes, based on refined errors analysis could be expected to play an important role in the next generation of algorithms.

SP3 – Software and large scale applications

- ➔ The program we propose is targeting the solution is large scale real-life problem. So producing software for the application is essential. Such software, to be efficient on Exascale computers, is expected to focus on load balancing issues; this means that synchronisation points are expensive and asynchronous versions of existing algorithms need to be investigated. For instance, the algorithms and their implementation may have to be based on DAG representations of algorithms, that is better suited to exploit all possible concurrency in computations.
- ➔ Additionally, even in the traditional application areas Exascale will change the way that numerical algorithms are used. The solution of inverse problems rather than the direct forward problem, for example in seismic modelling and shape optimization, image reconstruction for 2D and 3D images, and parameter identification within an optimization loop is becoming increasingly important. Similarly the incorporation of data uncertainties and sensitivity analysis is becoming computationally feasible. For control problems, for example in the stabilization of a flow or a chemical process. The program should stimulate tight interaction between mathematician, computer scientists and persons from application areas.

It needs to be mentioned that very few areas within the scope of this working group have easily identifiable gaps that can be closed by concentrated effort alone. Often progress is slow but steady. Solutions may not appear where they were initially suspected. It is therefore paramount that many possible avenues are explored and not only the (initially) most promising ones. In many areas fundamental breakthroughs are needed and these require sustained funding over many years.

Type of recommendation/Proposal for the next step

The DoE decided to create recently a work group (EMWG) of mathematicians with the goal to explore and find new Exascale algorithms. A workshop has been organized in August 2013 and EESI2 was invited to participate. The DoE is largely opened to enlarge the group to European people.

We recommended to create a similar action at European level and to share this action with the US EMWG.This may be done through a 3 years CSA or similar program with a budget of 1M. It may be pre-organized by EESI2 consortium.

Then, An IP over 4 years could host this project. It should mean a 15 million Euros budget, 6M on SP1, 6M on SP2, and 3M on SP3. Deliverables will come mostly in the form of Open Source software, ready for use for the scientific and industrial community.

3.2 Recommendation N°2 - Resilience

Scientific perimeter

The recommendation presented here covers two topics of resilience: tolerance to fail stop errors and tolerance to silent data corruptions.

Fail stop errors (application execution crashes) have multiple origins: Power outage, Hard errors (broken component: memory, network, core, disk, etc.), Detected uncorrectable soft errors (bit flip in memory, logic, bus), OS error (buffer overrun, deadlock, etc.), System Software error (service malfunction, time-out), Application bugs, Administrator error (Human), User errors (Human). In all cases, at Exascale, they will lead to massive resource usage and energy losses. Silent data corruptions are more insidious. Executions go to completion and may produce unnoticeable incorrect results. Taking into account these two problems is essential, especially when decision taking depends on the obtained results.

The resilience challenge cannot be addressed in isolation looking at a single software or hardware component. Resilience needs to be addressed considering the whole system: all layers of the software stack, all hardware components constituting the Exascale system and all usages of this system. This is clearly a multifaceted problem that needs experts from multiple disciplines.

Motivations

Resilience is considered as a “black swan”: the most difficult under addressed issue facing HPC (Advanced Scientific Computing Advisory Committee –ASCAC-, 2011). It is consistently presented as one of the top problems to solve to make Exascale computing a reality (IESP, EESI, ICIS reports to cite a few). The difficulty comes from the fact that current approaches will not work at Exascale. This forces the community to elaborate new effective and efficient resilience solutions. The situation could be compared with the transition from unicore to multicore forcing programmers to switch abruptly to parallel programming. However in the case of resilience, there is no middle ground: either the execution complete with correct results or it does not produce any usable result.

Europe has a long history of research in fault tolerance for distributed system and is at the origin of several of the most influential resilience research groups in the context of HPC. However, nowadays the main resilience forces in Europe are either not focusing on HPC anymore or are located in non-European countries. A large resilience project is an excellent opportunity for Europe to motivate its resilience experts to focus on HPC and to start rebuilding leading research groups in resilience for HPC.

The project aims at:

- Finding relevant effective and efficient solutions for Exascale resilience, addressing both fail stop errors and silent data corruptions, taking into account the multifaceted aspect of this problem
- Redeveloping in Europe a solid leadership in resilience for HPC by attracting resilience experts to the HPC domain

Impact (scientific, technological, societal & environmental)

Resilience to fail stop errors and data corruption is mandatory to obtain correct trustable results beyond the HPC context. Scientific progresses in resilience to fail stop errors may be applied in context of non-scientific applications (fast context checkpointing/restoration, failure prediction, state compression), Scientific progresses in data corruption resilience requires techniques that could be applied more broadly in the HPC context to debugging (the source of corruption being then the application code itself) and validation. Better resilience also means lower performance overhead, which means more science productions for a given time frame. Resilience to data corruptions also translates to better-informed decisions. Concerning the environment, fault tolerance already consumes a significant portion of the power used by production systems. Improving resilience will also contribute to reduce energy consumption by reducing failure free overhead and the amount of rework after failures.

Level of innovation

The project innovation is on two levels:

- Improvement of existing techniques for resilience to fail stop errors and development of innovative techniques
- Development of efficient techniques for resilience to silent data corruptions.

Mesoscale detailed description of the proposal

SP1 – Extend the applicability of Checkpoint/restart and migration

Checkpoint/restart form the basis of current resilience arsenal in production center. Its current implementation relying only on remote file system will not scale. The required improvements are well identified and the following points present the most critical ones:

➔ Improve checkpoint/restart performance
  - Understand memory access patterns for various application classes and derive properties that can enhance checkpointing asynchrony,
  - Explore and potentially leverage redundancy across multiple processes data structures at application level in order to identify applications classes that can benefit from specific optimizations like deduplication,
  - Improve Multi-level checkpoint/restart by decoupling checkpointing from the failures of storage levels.
  - Minimize the overhead of copying checkpoint images between the different storage levels of multilevel checkpoint restart environments
  - Understand how to make the best usage of non-volatile memory

SP2 - Improve system efficiency and execution recovery in presence of fail stop errors through better fault tolerant protocols

  - Understand the sensitivity of simulation codes to state inconsistency. If simulation codes tolerate by themselves minor state inconsistency, how this could be leveraged to design better resilience approaches.
  - How to leverage partial restart to improve system efficiency. All resilience researches in HPC so far seek to improve the efficiency/reliability a single execution. In some situations, the system efficiency/reliability is more important than the one of a single execution.
  - Understand how message logging can accelerate recovery.
  - Understand fault tolerance of MPI applications in the context of MPI3. In particular investigate if the sparsity, persistence and determinism of neighbor collective operations can be used for advanced message logging or other fault-tolerance protocols. Understand message logging and recovery schemes for RMA that enable transparent uncoordinated checkpointing and recovery schemes.

SP3 – Investigate alternatives to checkpoint/restart: tasks based checkpoint/restart, migration and redundancy

Several alternatives to checkpoint/restart have already been identified. However these solutions are less mature and need more study

  - Tasks based checkpoint/restart: Tasks are easy to migrate to a different processing unit in the event of a fault, as well as being easier to schedule. Dataflow based runtimes offers efficient localized checkpoint restart. We expect these runtimes to be effective for Exascale fault tolerance. A task is typically fired only when all its inputs are ready, the programmer annotates task directionality information as well as task inputs and outputs; both checkpointing and recovery are asynchronous since independent tasks that are not affected by the error could continue to execute. While the principles are well understood, research is still needed on efficient implementation and optimization techniques.
  - Migration: Process migration is known to be complex and resource consuming. Like for checkpointing, generic system migration is not necessary required. Migration could be made less costly if it minimizes the data movements and programmer could help to identify the relevant data to migrate. More research is needed to understand how live migration techniques can be adopted at application-level (i.e. what memory content

needs to be moved? In what order? How to minimize amount of transferred data? etc.).

- Redundancy: Replication is tempting because it provides protection against fail stop errors and data corruption (by comparison). However it is also extremely expensive: full replication requires twice resources and adds an overhead on executions. More research is needed to understand where replication can be used efficiently and how to reduce replication costs.

SP4 -Fault aware software stack

When multiple levels of the software stack, the hardware and potentially even multiple software at the same level are capable of detecting and correcting failures, the consistency of the resilience actions can be impaired.

- All software involved in the resilience should be fault aware and a notification/coordination infrastructure should guarantee relevant and consistent notifications/decisions/actions between these software.
- At low level, firmware and OS should be enhanced to handle RAS features.
- At higher level, more efforts should be put on understanding how to leverage and control hardware resilience features at runtime level
- At the top level, the application should be informed of failure and coordinate with other software layers to decide resilience actions to take.

SP5 Develop failure prediction

Failure prediction is an important highly speculative approach. If successful it can change drastically the way fail stop errors are tolerated. Progresses have been made in the understanding of the impact of failure prediction on execution performance in presence of failures (predicted or not). Another important progress is the understanding that failure prediction cannot handle 100% of failures and this technique should be coupled with some preventive techniques like checkpointing or replication.

- The main objective of failure prediction now should be on performing actual prediction, online, on real production systems. Prediction performance in this context is currently low: 30% of recall (30% of failures that happen are actually predicted). An Important research effort should be made to increase the recall to value around 80%.
- The second objective should be to design failure prediction workflow to work with extremely large and growing data sets. Very large systems are generating 1GB of information about their state per day (this is the data used for prediction). This is 100 larger than what researchers are using to develop failure prediction approaches. So failure prediction research should integrate a Big Data angle.

SP6 Resilient algorithms

For fail stop errors, some numerical kernels can provide alternatives to checkpoint/restart. Research has started mainly in the context of linear algebra (primary dense linear algebra) based on ABFT approaches. A few fault-oblivious linear equation solvers have been designed that have no overhead in fault free calculation and increasing penalty cost when the fault rate increases.

- The performance crosscutting between algorithm specific checkpointing and their fault-oblivious counterpart needs to be investigated to possibly decide at runtime what alternative to select (interactions with the runtime may be needed).

For data corruptions, much less works exist, often based on a checksum mechanism that enables to possibly detect a silent error but does not necessary permit to recover the corrupted data.

- Adapt numerical algorithms to re-compute or recover the lost piece of data.
- Exploit data redundancy exhibited in many parallel numerical algorithms to recover lost or corrupted data

The current research on resilient algorithms for fail stop errors and data corruption is limited to few linear algebra kernels.

- Extend study to all linear algebra kernels and other widely used numerical kernels.
- Extend research beyond the library context and study how to reconstruct state by combining recovery from partial checkpoint with recovery from data available on remaining processes of the execution.

One of the limitations of this approach is the lack of composability with the rest of the execution. These resilient algorithms are generally integrated in libraries and protect only the data passed and managed by them. However HPC applications are using data that are not passed to all calls of numerical libraries and these data need to be protected too.

- Investigate the composability of the above-mentioned techniques with other fault recovery solutions.
-

Type of recommendation/Proposal for the next step

An IP over 4 years could host this project. It should mean a 15 million Euros budget, 3M on SP1, 2M on SP2, 3M on SP3, 1M on SP4, 3M on SP5, 3M on SP6. Deliverables will come mostly in the form of Open Source software, ready for use for the scientific and industrial community.

3.3 Recommendation N°3 – Big Data

Scientific perimeter

The nature of science is changing – new scientific discoveries and socio-economical innovation will emerge from the analysis of large amounts of complex data generated by high-throughput instruments, observational systems, extreme-scale computing, and public World Wide Web.

In many domains – such as physics, earth sciences, environmental sciences, genomics, health sciences, financial and social sciences, etc. – our ability to acquire and generate data outpaces our ability to manage, explore, analyze and valorize them both technically and socially.

Big Data challenges conventional data storage and management, computation and data analytics, execution and communication models. This data deluge is not simply a hardware architecture and infrastructure problem but rather requires a new, holistic approach in order to extract the wealth of information hidden in those data, and to valorize the infrastructures that generate them.

Data-intensive analysis and Extreme-scale computing have evolved somewhat independently of each other. Synergistic challenges today in Data-Intensive science and Extreme Scale Computing Big Data require the development of transformational multi-disciplinary DataScope instruments, specifically designed to enable new data-intensive analytics, exploration, and valorization that are simply not possible today.

DataScope must take into account huge variations in full-service data lifecycle and commitments, encompassing data acquisition and generation, data-intensive processing and analytics from on-the fly analysis to public reuse, long term archival and curation, sharing, metadata and provenance chain, permanent identification – DOIs – security and privacy, etc.

Recommendations & Strategy

High-priority should be given in supporting an R&D initiative that can benefit both data-intensive science and extreme-scale computing so as to leverage their synergies:

- For science domains that need extreme-scale simulations, commensurate investments in extreme-scale computing capabilities and data infrastructure are necessary.
- For science domains that rely on high-throughput instruments and observational systems, commensurate investments in instrument/observation components and extreme-scale computing capabilities for continuous data stream analysis and reduction and other tiers of data analysis and processing.

The main objectives of this R&D initiative should focus on:

- Research in transformational algorithms to address fundamental challenges in extreme concurrency, asynchronous parallel data movement and access patterns, new alternative execution models, beyond the CSP and the embarrassingly parallel execution models, supporting asynchronous irregular applications and resilience, will benefit data analytics and computational methods for both data-intensive and extreme-scale computing.
- Research in disruptive I/O and in scalable multi-tiered data storage and parallel data management system technologies that can bring high throughput under heavy concurrency and supporting highly parallel data workflow, encompassing I/O middleware and scientific data formats supporting high-level data objects and data access patterns, scientific database technologies and indexing methods.
- Research in advanced data analytics algorithms and techniques, adopting new disruptive methodologies, to face the analysis of the big data deluge advancing in different scientific disciplines. This research should also promote and support the adoption of efficient metadata specification, management and interoperability in different scientific disciplines, as a key element to govern the scientific discovery process.

High priority in this R&D initiative should be given to research and other investments that simplify the end-to-end science workflows, the scientific data management, and improve the productivity of

scientists involved in extreme-scale and data-intensive computing. The main recommendation is to focus on:

- Research to simplify human-in-the-loop workflows for data-intensive science with virtual data facility and for heterogeneous full-service data lifecycle encompassing the full data provenance chain and reproducibility.
- Research to develop libraries of scalable data analytics and data mining algorithms - and software components for use in workflows - encompassing data abstractions, in-situ and out-of-core data processing modes, approximate data analytics, indexing-topological-statistical methods, uncertainty quantification.
- Research to support full-service data lifecycle management systems – and their wide diversity – supporting large distributed teams and internally organized scientific communities, encompassing on-the fly analysis, private and public reuse, archival and curation, together with metadata and metadata interoperability, data annotation and permanent identifiers, data and secondary data provenance chain, security and privacy, etc.

Mesoscale detailed description of the proposal

The proposed strategy is to **create and support a task force** to perform a multi-disciplinary analysis of these different objectives and substantiate them.

A key objective of this task force should be the **creation of a set of multi-disciplinary proxy applications** (**or mini applications**) and scientific use cases to capture the combined characteristics and needs of orchestrated data-intensive simulation and analytics applications that will feed into algorithmic, methods, software development activities and potentially drive future design/co-design strategies.

The duration of this task force should not exceed 6 months, and the task force must include a multi-disciplinary committee involving scientific disciplines that have already well structured around combined extreme-scale computing and data-intensive analytics. Among others, High Energy Physics, Astronomy & Astrophysics, Climate Sciences, Seismology, Genomic and Health sciences are key components. This committee could also embed representatives from the EUDAT FP7 project.

As already stated the outcome of this task could lead to focused calls for proposal for addressing on a prioritized way the different R&D topics listed in this proposal.

A final recommendation is to **bridge the gap between the number of current computational and computer scientists trained in both extreme-scale and data-intensive computing** and the needs for this combined expertise in support of the scientific and socio-economic challenges in Europe. The task force should look how fellowships, career awards, and funding grants increase can be targeted at increasing the pool of computer and computational scientists trained in both Exascale and data-intensive computing.

Type of recommendation/Proposal for the next step

A possible funding scheme could be a NoE scheme or a CSA with a combined STREP activity.

3.4 Recommendation N°4 - Couplers

Scientific perimeter

This text describes a topic which should be included in future strategies for HPC developments in Europe: the passage from a paradigm where one solver is run on one parallel system (a task which can be daunting in many cases) to a new paradigm where multiple solvers have to be run and coupled simultaneously on one or many parallel systems (a task which will be much more difficult and requires specific attention). This evolution is required to address multiple scientific challenges (climate change for example) but also to attract industrial interest for massively parallel systems because many industrial problems include multiphysics studies and therefore simulations where multiple solvers must be coupled. The following text describes the context for this study, the possible solutions and the reasons why efforts are required at the European level to prepare this work. The idea of developing common coupling software which can be used by all communities is pushed forward as an interesting path.

Simulation is now an essential part of scientific research and actively participates to design phases in industry, complementing the traditional approaches of theory and experimentation. Ensuring that simulations execute efficiently on next generation supercomputing platforms is essential for maintaining productivity in many domains. Recent workshops have identified areas of scientific research that demand computational power beyond even that provided by the fastest petascale computing resources available today. This includes applications in climate science, high-energy physic, fusion energy, computational biology, environmental sustainability, and combustion science. These and similar applications are of vital societal importance and continue to push the boundaries of computing.

Some of the most complex scientific simulations requiring extreme scale computing are based on multi-physics models that simulate multiple interacting physical processes by combining independently developed modeling components into a single model. Development of coupled models is an inherently multi-disciplinary effort, requiring scientific expertise from multiple domains as well as technical expertise in developing high performance software. One source of uncertainty is whether the state-of-the-art software libraries used today for model coupling will scale efficiently on Exascale platforms.

In many communities (aerospace, aeronautical, climate, automotive, astronomy, electronics …), access to greater computing power allows to increase spatial and temporal resolutions of the models as well as increase geometrical complexities of the simulated systems to capture fine-grained physical phenomena. More direct computations are performed with less modeling closures, which can be incomplete, inadequate or erroneous. Consequently, important phenomenon can be accurately captured such as convective clouds in the tropics, precipitations, flame description in combustors, wall shear stress and heat transfer in turbine blades … Increasing computational power is also a path for longer time simulations in these different sciences in order to study long period phenomenon and accumulate accurate statistics, investigate climate evolution on a long term, etc. Finally, huge computing powers help to increase number of simulation runs for uncertainty quantification (model parameterization, boundary conditions) and ensemble simulations.

Motivations

Dealing with multi solver simulations, the scientific communities are faced with the challenge of running coupled model simulations efficiently on Exascale machines with highly loaded models that exchange data with a high frequency. Today's climate or Computational Fluid Dynamics models are the results of decade of work adding up hundreds of person-years and millions of lines of codes. It would be extremely time consuming and expensive to completely rewrite and validate these models. Instead, efforts have certainly to be done to modularize and restructure them in a scalable, portable and maintainable way. Coupled models are challenging to develop due to the need to coordinate execution of the independently developed model components while resolving both scientific and technical heterogeneities. The component models (i.e., the constituents that are linked together) often utilize different spatial representations, requiring interpolation from one model's output to another model's input, evolve at different timescales, requiring temporal averaging or statistical sampling, and use incompatible data structures, requiring on-the-fly data structure conversion. Moreover, the

resolution algorithms as well as the computer implementation (type of parallelism, scalar or vector processing) for each physic have been optimized for years and are not always similar. A classic example in combustion science is the coupling between radiation, which is usually parallelized on frequencies and is solved efficiently on GPUs, with reactive fluid dynamics in gas turbine parallelized by sub-domains running most of the time on CPUs. The efficiency of these different coupling requirements in conjunction with the load balancing of the component models is therefore crucial for Exascale computing. The main design constraints underlined for efficient Exascale computing are of course mandatory for coupled applications: use more parallelism, less memory and less communication.

The current coupling technologies used for multi-physics and multi-components simulations can roughly be split into two main categories. The first one is designed to offer faster implementation while the second one focuses on high performance via architectural conformity.

On one hand, *direct coupling,* using a fully concurrent multiple executable approach, is somewhat less flexible regarding the execution of the component models and does not allow component models to share data directly through memory (i.e., data exchange requires interprocess communication). However, it is relatively straightforward to implement. In this case, the original components are run as separate concurrent executables, and their main characteristics, such as memory management or internal parallelism, remain practically untouched with respect to their standalone mode. The exchange of coupling data is performed through "put" and "get" instructions implemented in the component codes. It is the user's responsibility to ensure that the component models coherently define some global parameters such as the total run duration, the meeting points, etc. The main advantage of this approach is that it requires minimal intrusion into, or restructuring of, existing legacy codes. The drawback is that it is less flexible, and in some cases, less efficient than the coupling exchanges as it necessarily imply additional data transfer between the solvers. Typical libraries that implement this type of paradigm include for example OASIS, Open-PALM, MpCCI, Distributed Coupling Toolkit (DCT) and Typed Data Transfer library (TDT).

On the other hand, *coupling via top-level interfaces* within one integrated application (such proposed by the American coupling infrastructure Earth System Modeling Framework (ESMF), Community Climate System Model (CCSM), CPL7 or Flexible Modeling System (FMS)) requires some standardization around high level interfaces, design, and datatypes, in order to provide opportunities to run models in more flexible and efficient configurations. In this case, the model source code is decomposed into init, run, and finalize units and the interface must match the standard expected by the coupling layer. The coupling data are made available as input and/or output at each calling interface. Following the coupling infrastructure coding standards, a new code is then rebuild based on a standard wrapping layer assembling the elementary units and ensuring the targeted exchange of data. This approach forces the components to expose both data and control interfaces; it is the wrapping layer that executes the units and this "inversion of control" automatically ensures consistency of global parameters across component models (e.g., run duration). This approach increases flexibility in how components models are mapped to computational resources and is, in some cases, more efficient as the component models can be executed concurrently, sequentially, or in some hybrid mode and as coupling exchanges can be in some cases optimized as shared memory accesses. A drawback to this approach is that it requires a deeper level of change to legacy codes in order to take maximum advantage of the shared infrastructure (e.g., splitting them into init, run, and finalize units, which might be difficult to achieve) and imposes the use of the coupling layer standard-calling interface and data structure.

There is not a clear view of which coupling architecture will survive to Exascale computing, *ie*, the multiple or single binary. Both are submitted to almost the same challenges (use more parallelism, less communications and less memory) and offer alternative solutions to address them. Current performance optimization methods for coupled systems are relatively simple heuristics based on empirical scalability measures of the component models. In both the multiple-binary and single-binary approaches, model throughput is dependent on efficient overlapping of the component model computations in order to minimize processor idle time. Because a majority of coupled applications are computation dominant, this procedure has been sufficient to achieve adequate throughput of coupled models executed on thousands of cores. At today's typical processor counts, the coupler overhead does not introduce a significant performance penalty. A study by the Earth System Modeling Framework shows that for one typical configuration of the Community Climate System Model (CCSM), time spent in the coupler is less than 3% of the total running time. The same tendency are observed in

conjugate heat transfer simulations of gas turbine combustion chamber with OpenPALM on 12 000 cores. However, it has been showed in this last study that the scalability of the coupling depends on communication scheme between the models. For example, increasing the number of cores of just one model can create communication bottlenecks. Moreover, at very large processor counts, the number of messages required for both intra- and inter-model communication increases dramatically such that optimization of the coupler will become important to maintain high model throughput. A recent set of performance experiments on a usage of CCSM based on data centralization supports this prediction: For the highest resolution runs, the percentage of time spent inside the coupler increases dramatically when compared to typical resolution runs. At processor counts over 32 000, the coupler accounts for nearly half of the execution time. The authors conclude that future optimization efforts must focus on improving the efficiency of the coupler itself, not just the component models. Furthermore, since even incremental improvements to scientific codes can take years to implement, it is critical to identify the most serious performance bottlenecks for today's highest resolution coupled models and begin to address them now. Independently of the coupling architecture, it appears crucial to design the component codes targeting a priori coupled set-up instead to trying to assemble a posteriori codes that have been designed to run stand-alone. Indeed, as the mesh partitioning is of primary importance for application scaling in a standalone computation, the way the data are split in the different component models will determine the locality of the data in memory as well as the complexity of the communication pattern between the component models.

Impact

In communities where massively coupled applications have a large scientific, technological and societal impact (such as Climate and Combustion science), several initiatives exist in order to gauge existing framework and identify bottlenecks.

Such work has to continue in order to clarify the current status of coupling and provide solutions for future problems. Based on the existing conclusions, the recommendations to improve actual coupled simulations and to investigate solution for Exascale computing concern the improvements of the coupling architectures as well as of the coupled applications. To do so, the research groups have to be composed of inter-disciplinary specialists such as computers scientists, applied mathematicians, scientists…

Recommendations on improvement of the knowledge on coupled application scalability

Empirical performance measurements have to be performed and analyzed in more generic ways on existing applications as well as on toy models that reproduce different coupled architectures and applications of single and multi-executable.

Complementary to empirical measurements on actual systems, performance modeling of coupled applications has to be investigated. Such modeling enables predictions of model throughput from an analytical formula based on application and system parameters. A performance model is an analytical formula that describes salient performance characteristics of applications running on existing or future computing platforms. While performance modeling of scientific applications is not new, to our knowledge the development of a coupling-aware performance model is novel. Existing approaches derive a performance model for an application as a whole—i.e., they do not separately model performance of individual components within an application. A coupling-aware performance model is a composite of multiple component performance models—one for each of the participating simulations—as well as an additional performance model for the coupling software itself and for the fact that the components run in a coupled environment.

Recommendations on the coupler improvements:

Based on recent experiences on actual computers, several essential improvements of coupling paradigms have to be investigated:

- It is of primary interest to design a standard coupling API in order to enable interoperability, ease the integration of new models and cross disciplinary exchanges and sharing of performance analysis,

- The methods implemented in the software have to avoid centralization of data and prefer distribution even for high order interpolation methods,
- The performance of localization process (needed for the weight and address calculations required for the interpolation of the data between the components meshes) at the beginning of the coupled computations, and optimization of this process for geometrical or mesh changes during the simulations as well as of communication performances between the coupled models have to be increase by use of asynchronous processes, hide of operations by computations, intelligent search algorithms…
- As for the next generation of codes, it is important to investigate new programming models and systems for parallelism (PGAS, hybrid MPI/OpenMP…) at the level of the component models and of the coupler,
- Finally, in order to benefit from scalability performances of component models on different hardware, investigations in the possibility to couple efficiently component models running on heterogeneous architectures have to be achieved.

Recommendations on the coupled models improvements:

From actual performance measurements, an important path to efficient Exascale coupled simulations is to integrate deeper the coupling process inside the component model environments:

- An advanced comparison between single and multi executables have to be done on several component models types in order to extract the pro and cons of each methodology in the context of Exascale computing.
- It is essential to improve the coupling algorithms in order to hide communications with computations in the models, exchange only relevant information with reasonable frequency …
- The communication costs between the models have to be reduced by optimizing the balance between processes that are concerned by the coupling and other ones that are not, as well as the corresponding data distribution, optimizing the communication schemes between the coupled models via co-partitioning methods …

Recommendations on the software environment:

Finally, going to Exascale simulations brings new challenges in setting up multi-physics and multi-components on complex geometries in order to produce accurate and exploitable results. Hence, tools dedicated to ease to set up coupled computations have to be proposed (mesh connection between model, quick verifications of conformity, evaluation of physical quantities during computations …).

Type of recommendation/Proposal for the next step

An IP over 4 years could host this project. It should mean an approximate 15 million Euros budget.

3.5 Recommendation N°5 - High productivity programming models

Scientific perimeter

This recommendation covers the topic of High productivity programming models (eDSL, rapid prototyping programming languages…) with support to parallelization in heterogeneous architectures that are able to exploit aspects such as locality, load-balancing, etc. The objective is to tackle different fronts:

- To ease the development of applications, decoupling the logic of the applications from the actual resources, even enabling rapid prototyping
- Enable exploitation of the heterogeneous and parallel platform from runtimes that are able to exploit aspects such as locality or load balancing

Motivations

Innovation cycles in high performance computing platforms are characterized by moving towards more and more complex architectures accompanied with a crucial need for efficient programmability. Just for computing resources, aspects such as larger number of resources organized in different hierarchies (nodes, sockets, cores, …) of heterogeneous nature (different ISA, different variability in the availability of these resources, different performance/frequency, faults, …), made difficult the programming of such machines. However, it is not sustainable to write and re-write applications for new infrastructures. In this sense, programming models that enable to abstract the logic of the applications from the actual infrastructure details are necessary. Examples are: approaches that enable sequential programming with annotations, embedded Domain Specific Languages (eDSLs) or programming languages/scripting languages that enable rapid prototyping (i.e., python).

Also with larger number of nodes and core count, global synchronicity is not acceptable anymore and programming models and runtimes that support asynchronous synchronizations, such as task-based approaches guided by data dependences where the typical fork-join scheme is avoided are a must. In terms of communications, the programming model should be able to support and efficiently manage non-blocking and asynchronous collective operations. The programming model would optimally go with intelligent runtimes that are able to take into account aspects such as resource management for tasks, exploitation of the data locality, moving computation to the data and enable other optimizations such as automatic load balancing between different processes. A crucial aspect to be considered is also the inclusion of energy efficiency criteria both at the programming model and runtime decisions.

The programming environment would optimally come with a set of tools that support the development of the applications. For example, analysis tools for determining inter-dependent program parts, facilitating taskification of code, performance analysis and simulation tools to predict the behaviour of the application and debugging tools.

While the programming model is a key element for Exascale, no specific project focused on the programming model has been funded in the last calls. Also, with the recent extensions of OpenMP and MPI to better support aspects such as asynchronicity and with Europe being very well positioned in programming models, it is the right moment to fund such an initiative which can unify good ideas that are in Europe, and get a strong European programming model.

It is not sensible to continue writing applications just with Fortran + MPI, approaches that are able to offer a higher level of abstraction and rely on powerful runtimes to ensure efficient execution should be promoted.

The effort should be done together with applications, especially for approaches such as embedded Domain Specific Languages (eDSLs) where a co-design approach is necessary.

For the scientific community it would also be of high importance to have available programming systems where it is possible, for example, to interplay between compute platforms and data-bases is possible, to design and execute workflows for high-throughput computing or multi-stage computational refinements. Fields where this can have impact are materials design, chemical engineering or whether/climate simulations.

Impact (scientific, technological, societal & environmental)

The impact is at different levels.

Europe is very well positioned in programming models. If a European programming model is promoted it can have impact at worldwide level in the way applications are developed.

At scientific levels, it will enable a faster development of applications that will be easier to be maintained and extended, and ported to other infrastructures.

At technological level, it will enable applications to scale to larger number of core-counts, while being able to exploit the underlying heterogeneous infrastructure.

Also, there will be a reduction of cost of development of the applications together with the achievement of more energy-efficient applications.

Level of innovation

One goal of the current proposition is to work on high productivity programming models and eDSLs, in order to match the ever increasing complexity of HPC architectures with the community of scientific programmers, who eventually map complex algorithms to the architectures. To get out the maximum of performance from an architecture, the programming environment has to greatly support efficient programming, for not increasing the gap between actually reached program- and potential peak-performance on a machine. A solid support from software and eDSLs to reach performance is essential for an economic use of HPC architectures and will become crucial for high investments like Exascale machines.

Level of innovation and need is therefore high.

Remark: there was a study by Nvidia that after optimizing the code layout/structure for GPUs, codes were often already 10 faster on the CPU than before optimization. The GPU story 'NVidia says large GPGPU speed up claims were due to bad original code'

http://www.theinquirer.net/inquirer/news/2227038/nvidia-says-large-gpgpu-speed-up-claims-were-due-to-bad-original-code

Mesoscale detailed description of the proposal

(If EC retains the EESI R&D proposal then a detailed proposal will be drafted)

High Productivity Programming Models for Heterogeneous High Performance Computing Architectures

Proposals should address one or more of the following issues

- Support task-based asynchronous execution model, hiding the details of the HW platform
- Automatic exploitation of parallelism enabling scalability at large number of nodes
- communication hiding programming in heterogeneous architectures
- embedded Domain Specific Languages for high efficient implementations in 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 is possible, to design and execute workflows for high-throughput computing or multi-stage computational refinements.

Tools and implementations should demonstrate their applicability in numerical kernels, mini-apps and large scale simulations.

This recommendation is linked with the Mini Apps one.

Type of recommendation/Proposal for the next step

The recommendation is to fund 1 IP + 2 Streps with duration of 30 - 36 months with a budget of 15 MEuros.

3.6 Recommendation N°6 – Mini Apps

Executive summary

Traditionally, co-design applies to the process by which processor manufacturers, software developers, domain scientists, computer scientists and applied mathematicians work together to collaborate on the design of the hardware, software and underlying algorithms. This “virtuous cycle” operates as an iterative approach, during which the study alternatively focuses on one component while all the others are kept fixed.

In the past decade, the shortening of the cycles of new architectures combined with the complexity of HPC applications (multi scale, multi physics, massive data analysis, more interactive features, etc…) have been the base for research in mini applications development (see for example [1], [2], [4]), computational reference kernels (TORCH [5]), or synthetic performance modelling [3].

Mini applications as lighter versions of complex HPC applications attract growing interest as a flexible test bed to facilitate software development. They aim to project how future more complex applications will interact with future computing hardware and are highly valuable to system designers and architects as long as they represent the behaviour of the complete workload with sufficient fidelity. They should capture a code’s essential features - core data structures, algorithms, parallel constructs- while demonstrating state-of-the-art implementations relying on modern programming paradigms. Two ways to build them: either to start from the real application and to apply the needed changes or to write a new code breaking a specific barrier to extreme scale, constructing the proto-application around it. Both approaches, evolutionary vs. revolutionary are valuable but not interchangeable. The need to maintain proto-applications up-to-date with the major evolution of the reference code is critical for their long term usability.

Scientific perimeter and methodology

The present recommendation concentrates on the scientific studies to be undertaken at the crossing point between domain scientists, computer scientists and applied mathematicians in order to develop a library of mini applications capable of enabling scalability and performance prediction for component base design.

When developed with the qualities requirements and features, mini applications can render unique services for architectural simulation framework, pre-production HW prototype, scalable Exascale software, component base systems, and for the HPC providers if they can predict scalability and performance. Final prototyped applications, based on mini applications and representative of realistic workloads, can be derived for which specific auto tuning optimizations and node architecture or communication network simulators can be evaluated. The methodology based on mini applications coupled with tools to assess scalability and performance are central to the co design process, key for the HPC application developer facing crucial choices when addressing extreme parallelism and heterogeneous architectures, and an excellent vehicle for outreach and dissemination.

Motivations

Too often, each HPC domain conducts its research independently without benefitting from findings of other domains sharing a number of similar or related problems: numerical kernels, data movement, programming models are good examples of such topics, among many others. Even though significant efforts have been initiated by academic or government research around mini applications, ISV and codes used in industry in particular has not been the object of focussed efforts in this area.

As shown in the following picture mini applications (or compact applications in the US Co Design taxonomy) are just on top of kernels (also called proto applications) and skeletons and allow capturing the applications essentials.

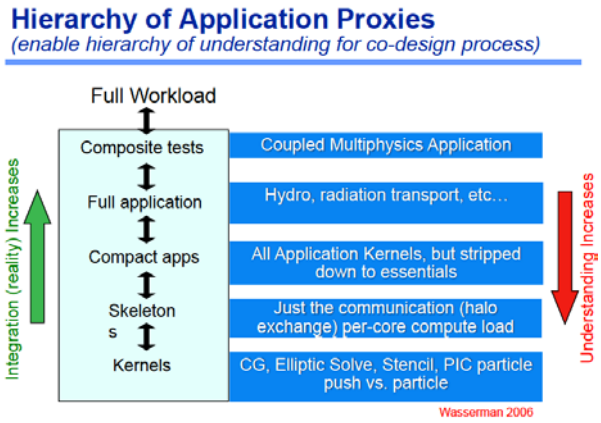


Figure 1 - Co Design hierarchy of applications proxies

The following picture illustrates the existing US CoDex methodology used by the DoE Centers of Excellence in fusion, advanced reactors, materials or high-energy physics.

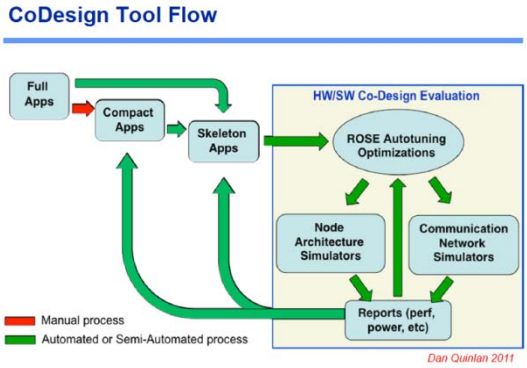


Figure 2 - Overview of the US CoDex co design tool chain

Similar methodologies are starting to be deployed as well in Europe; this is in particular the case for the Intel European Exascale Laboratories in Paris (with the Codelet Tuning Infrastructure and its associated tool chain) and Leuven (with the development of multicore simulators). Such initiatives developing European wide methodologies and software need to be amplified in the future for tooling the co design effort of European applications and technology.

Impact

When designed with the qualities requirements and features, the mini and proto applications will render valuable and often unique services for architectural simulation framework, pre-production HW prototype, scalable Exascale software, component base systems, and for the HPC providers by providing prediction scalability and performance for relevant workloads.

Based on mini applications its possible via automatic tools for features generation and extraction to derivate proto apps and final kernels in which specific auto tuning optimisations and node architecture or communication network simulators (which allow to simulate hardware before it is built) could be used for co designing computational kernels on novel architectures. Such outcome could then re re-propagated back from kernels to proto apps and then to mini applications and finally full applications.

Level of innovation

Short report about the state-of-the -art for mini- and proto apps between the US, Europe and Japan
One of us (MC Sawley) held a Bird-of-feather session on the topic during ISC13 in Leipzig, co-organized with Satoshi Matsuoka from Riken and Tokyo Institute of Technology, and with Jim Ang, from Sandi NL. Here is a summary of the messages delivered on this occasion.

i) From Sandia SNL, we heard that the idea that changing software is as costly, if not more as changing hardware is counterintuitive; however this fact has been the driver for the Mantevo project since its start in 2007, pursuing the goal to influence the hardware design. The initiators defended the need to sustain the effort around the Mantevo Library for a minimum of 5 years and possibly much longer, Ref [1], in order to be effective. Sandia NL sees an impact of the mini applications in multiple dimensions of Co-design:

- HW: Node and System architecture
- SW: Application and System Software

They also hope to be able to use mini applications to reexamine conventional ideas on COTS and system balance for Exascale design, such as

- Component performance
- Investment
  - Platform costs
  - R&D investments

The Mantevo community is active on the www.mantevo.org website, where papers are regularly posted. Following SC12, a Virtual box with some of the most recent mini applications can be downloaded. The efforts are also moving in the direction of assessing the predictive performance of the mini applications, Ref [2].

ii) Japanese HPCI Exascale Feasibility Study-Mini Applications
In Japan the group lead by Satoshi Matsuoka and Hirofuji Tomita is driving the feasibility studies on application software for the Exascale ([6]), collaborating with the 3 teams that are developing the hardware and system software. Such studies must be finalized by the end of this year, after which the R&D program for the Exascale will unfold. S. Matusoka and his colleague Tomita are focused on writing a report based on 3 main topics:

- Write “Computational Science Roadmap” and show performance requirement for Exascale era
- Select characteristic applications and assemble into Mini-App benchmarks
- Create benchmarking repository and performance models for Mini-Apps (planned for March 2014)

23 applications have been selected according to their societal impact, adequacy to Exascale requirements, impact on academic and industrial usage, and to the commitment of the main developers. Collaborations with ASPEN team of J. Vetter at ORNL, Ref [3] and with T. Hoefler at ETH Zurich have been established for the performance modelling activities,

iii) Intel European Exascale Labs
Developments made in two Intel European Exascale joint laboratories were presented during this BoF. The first is the work done on a 3DFD kernel, stencil code, inspired by a full RTM seismic application, and CnC, Concurrent Collections. This work, performed at the Exascale Computing Research in Paris in 2012, gave rise to an Open Source Release of the mini-RTM. The second, Helsim, is inspired by the full PIC code for Space weather, and is being actively used at the Exascience lab in Leuven for the purpose of comparing profiling tools, identifying bottlenecks, HW/SW co-design and to study the impact of new Exascale algorithms.

In addition to the activities on mini applications , the Paris and Leuven labs conduct complementary work on tools and simulator: at the University of Ghent, partner to the Leuven lab, developments are being undertaken to further enhance Sniper, a X86 multi core simulator; in Paris, the UVSQ, central to the Exascale Computing Research Lab has developed its trade mark around highly profile tools (the MAQAO suite and the Codelet Tuning Infrastructure) for application characterization and optimization, that are being integrated with open source tools such as Scalasca and TAU. We envision that such tools and methodologies will play a major role in profiling the minim applications library and pave the way to a *software simulation* framework

These two labs are part of a consortium (10 partners, 6 countries) which filed a proposal to FP7 Call 10, EXA2CT. The essentials of the project which will start in September, are summarized :

- Goals: Creating breakthrough innovative algorithms and new programming models, paving the way to Exascale for leadership class codes

- Strategy Start from existing, leadership class applications, with a major industrial impact and extract knowledge about them to create the appropriate proto-applications.
- Enhanced Numerical Algorithms (Scalable, Pipelined, Robust Numerical Solvers) that scale up to Exascale performance that survive hardware failures
- Enhanced Programming Models (GPI/GASPI, Shark, Irregular Stencil Code, PATUS) communication model
    - to deal with platform heterogeneity
    - to deal with resilience
- Outcome: The results of the project will be made available
    - Proto-Applications
    - Programming Models & Runtime Systems
    - Libraries

Connections with the other Exascale projects

In the CRESTA FP7 project (http://cresta-project.eu) which aims to enable a key set of co-design applications for Exascale and to build and explore appropriate system software for Exascale platforms specific research is currently performed for deriving from the current 6 co-design vehicles (biomolecular systems, fusion energy, the virtual physiological human, numerical weather prediction and engineering) computational kernels, and helping partners develop quantitative assessments of the applications performance.

In the Mont Blanc FP7 project (http://www.montblanc-project.eu) which aims to assess the potential of low power devices as possible building blocks for future Exascale systems, a specific on-going activity is relative to the extraction of pertinent kernels and mini applications from the initial set of 11 applications in order to assess the programmability and the performance of the OmpSs programming model provided by BSC.

The DEEP FP7 project (http://www.deep-project.eu/deep-project/EN/Home/home_node.html) takes the concept of compute acceleration to a new level: instead of adding accelerator cards to standard Cluster nodes, a Cluster of accelerators, called Booster, complements a conventional high-performance computing (HPC) system and increases its compute performance and scalability. Heterogeneity at the Cluster level enables applications to run parts with high scalability on the Booster and parts with complex control flow and low scalability on the Cluster, using both sides of the system at maximum efficiency.

Mesoscale description of the proposal

1- Kick off efforts with 4 to 6 experts from EESI2 WP3 in order to:

i) Review carefully what applications are of relevance for the scientific and industrial eco-system in Europe that are impacted by the Exascale ramping up; bring forward clearly societal and economic impact→ around 20 to 30 applications

ii) Connect with the efforts in parallel on programming models, large data analysis, performance and scalability prediction tools; the idea is to make use or adapt the most advanced existing developments whenever possible.

iii) Map them with the existing Exascale projects, assess gap; establish priority list

iv) Propose methodology, metrics, roadmap and needed resources; establish the gap, if any, for software simulators

v) Connect with relevant international efforts to explore possibility of synergy

2- PoC

i) Build the community of the Exascale givers (main developers i.e. INRIA or Cerfacs, main users i.e. AIRBUS or BMW, PRACE centres) by creating the motivation factor → student competition, annual EU PRACE challenge, defining common metrics, etc...

ii) Establish the map of commonalities/ differentiators with similar initiatives in US and Japan; organize a common workshop end of 2014; fund student /fellowship exchange

iii) Build the link with ETP4HPC by inviting the members to test, assess and use the library of applications to validate the new, emerging technology for new HW and System SW

3- Launch a call for proposal in order to fund a targeted effort of Exascale proto-application design, validation, demonstration for the top 15 applications for 2014-2015, then the next 15 (the more complex or less ready one) for 2016-2017. Each batch should have a clear demonstration step.

Type of recommendation, type of funding

CSA; 6 month duration.

References

[1]: Improving Performance via Mini-applications, SAND2009-5574, Printed September 2009

[2]: Assessing the Predictive Capabilities of Mini Applications, *Richard Barrett et Al*, SC12 Research Poster.

[3]: Aspen: A Domain Specific Language for Performance Modeling, *Kyle L. Spafford Oak Ridge National Laboratory spaffordkl@ornl.gov Jeffrey S. Vetter Oak Ridge National Laboratory Georgia Institute of Technology vetter@computer.org*

[4]: http://exactcodesign.org/proxy-app-software/

[5]: http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-144.html

[6] http://www. Exascale.org/mediawiki/images/1/14/Talk-4-maruyama.pdf

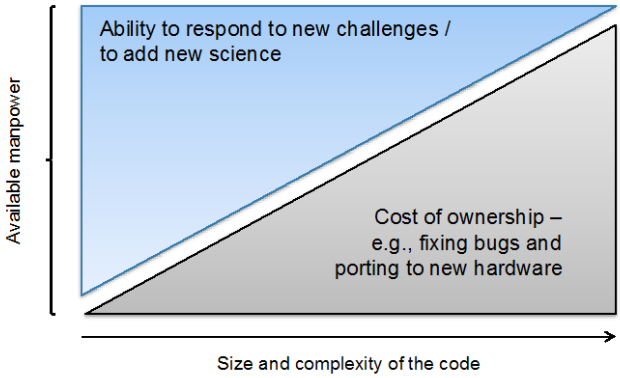
3.7 Recommendation N°7 - Software Engineering Methods for High-Performance Computing

Scientific perimeter

According to ISO/IEC/IEEE 24765:2010, software engineering is defined as “the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to 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. In contrast to other application areas, a distinctive property of this class of software is the high demand for computational resources, a major factor in the mix of resources needed to arrive at an answer to the scientific questions this software is supposed to address. Performance thus emerges as an essential non-functional requirement.

Starting from very vague high-level requirements, high-performance computing (HPC) applications are often developed incrementally, evolving through many versions, and rarely reaching a final version as the frontier of sciences in the field and in computing hardware is constantly advancing. The software process is often described as informally agile. Many codes begin their life in academia as the product of a single PhD student or lone researcher. As the functionality and the demand grow, the team becomes bigger. Initially, users and developers are usually identical. Once the code has grown to reach a certain level of maturity, scientists or engineers external to the development team start joining the user base. Successful codes are often commercialized or continued as community projects, especially if the resources and the expertise of a single organization do not suffice. The larger the user base and the more critical the application, the more important become other non-functional requirements such as performance on high-end parallel platforms, robustness, and ease-of-use. Correctness is always the most important requirement, but the effects of errors are more disastrous if more people depend on correct results. In general, the lifespan of successful HPC applications can be in the order of multiple decades, typically surviving several generations of hardware.

Motivation



In the past, the availability of fast hardware has been seen as the primary factor in the mix of resources needed to arrive at a solution, neglecting other important aspects such as development speed and cost. Ill-equipped low-level programming environments have been regarded as merely inconvenient, something that can still be used effectively given enough training and time. Development would just take longer but ultimately reach the desired objective. However, in addition to the cost of adding new features, software has also a *cost of ownership* – even if the functionality remains stable. Bugs have to be fixed and the code has to be ported to new platforms at least before the current ones are decommissioned, which usually happens every few years. The less maintainable an application is, the higher its cost of ownership becomes and the less manpower is available for the creation of new scientific capabilities. New developments also become more challenging and thus more expensive the more complex the original code base is. Just like a major portion of the operational costs of hardware have been distilled into the term *power envelope*, it is helpful to think of software as having a *manpower envelope* that places a limit on its size and complexity. If size and complexity exceed the limit of what the manpower envelope can support, the code is destined to stagnate (see figure). The objective must therefore be to lower the cost of ownership and utilize the remaining manpower as efficiently as possible to maximize the potential of the code.

Given the long lifetime of HPC codes, development rarely means development from scratch. Instead, it more than often means extending or restructuring existing code, which was often written by people different from those who apply the changes. This is true in particular for codes being developed in academic institutions with their natural staff fluctuation in the form of consecutive generations of PhD students. For a grown code base, even prototypical changes are expensive and require careful planning. Especially for large industry and community codes, regression testing emerges as a major cost factor. Regression testing has to cover functional as well as non-functional requirements such as performance. Given that simulation codes often aim at scenarios that are hardly accessible to experiments, the generation of test cases itself can become challenging. In industries such as automotive and aerospace, testing may have legal implications where safety certificates can only be issued if simulation software has been verified. Any changes to the software can therefore be particularly time-consuming and costly.

To reach Exascale, applications will have to address numerous technical hurdles simultaneously, including (i) scalable and energy efficient algorithm design, (ii) hardware faults which may compromise scalability, (iii) soft errors which may compromise correctness, and (iv) the pre- and post-processing of vast amounts of input and output data. Moreover, the problems Exascale applications are supposed to solve are also more complex from a scientific viewpoint, often requiring the coupling of methods across many length and time scales. As a consequence, the development costs of applications that strive to run at this level will rise significantly, creating a need to reassess software processes from an economic perspective. Otherwise, the capability of our software base will be left way behind the potential of our hardware.

Impact

Improved software engineering methods have the potential to lower development costs and make development more sustainable. In this context, sustainability means the ability to grow or change. This is necessary to preserve the value of our investments in software. If development is not sustainable, an application may never reach Exascale. Thus, proper software engineering goes beyond budgetary considerations, and indeed it is almost certainly an essential ingredient on the path towards Exascale. While not at the frontier of domain science, software engineering methods should be regarded as a critical link of the supply chain without which this frontier cannot advance. In this sense, its contribution will significantly help justify the investments in Exascale hard- and software.

Level of innovation

The required level of innovation has two dimensions – one cultural and one technological. Changing the culture means to move away from a field split into the sub-disciplines domain science, numerical analysis, systems architecture, and programming towards a more integrated view that also encompasses people and processes. From a technological viewpoint, the required methods and tools will face the same two categories of challenges that application developers face. There is a genuine methodological challenge of solving the problem at hand and there is the Exascale challenge of making the solution work with hundreds of millions of threads.

Proposal

In this recommendation, we concentrate on design, quality management, and reuse aspects with the goal of providing appropriate methods and tools to support them. Furthermore, a smaller but still non-negligible topic within the scope of this recommendation is keeping track of development practices beyond anecdotal evidence.

Because separate recommendations are devoted to (i) couplers, (ii) verification, validation, & uncertainty quantification, (iii) high-productivity programming models, and (iv) resilience, we do not cover those topics, even though we deem them relevant in the context of software engineering. However, we mention them where they fall into a broader problem area we address.

1. Develop **predictive methods and tools to assist software re-design and co-design** of scientific application software for future platforms. They need to predict the effects of changes

in the software, especially with respect to performance, scalability, energy efficiency, and fault tolerance. We further require lightweight predictive tools that estimate the effects of changes of the hardware as a basis for the co-design of future platforms. While such predictions are already being made, the methods being applied are still ad-hoc and tools, where they exist, lack the desired level of automation and robustness.

2. Develop **scalable debugging and performance analysis tools**. While tools for correctness and performance analysis exist, they lack the required scalability and offer only limited insight. Both debuggers and performance analyzers need to be enabled to cope with even more massive amounts of information in a scalable manner. Moreover, modeling tools are needed that anticipate performance bottlenecks before they become manifest at the target scale. Such tools will also be useful in assessing the potential of existing codes as a prerequisite for planning and prioritizing changes. With their strong record in tools, the European community is optimally prepared.
3. Support the **transformation of codes into resilient software**. As systems grow in size and in complexity both with respect to hard- and software, applications become more susceptible to both hard errors, which cause programs to fail outright, and soft errors, which adversely affect correctness of results and reproducibility. For this reason, numerous legacy codes will have to switch to fault-tolerant algorithms and programming models, which will enable them to cope with hard and soft errors. Options include exact error recovery and fuzzy algorithms, which adapt to failures while maintaining the desired outcome. Methods and tools are needed to analyze the fault-tolerance requirements of an application and to guide the associated re-design process. The primary objective is to strike a balance between the desired degree of resilience and the fault-tolerance overhead in terms of time and space.

Funding instruments

The above funding objectives are best implemented as a collection of Specific Targeted Research Projects (STREPs) running for three years with a total funding volume of €8M, with about €2.3M per STREP and ~€0.4M for the accompanying survey.

Consortia should gather expertise from domain science and/or numerical analysis, HPC-oriented computer science, and classic software engineering. The collection of projects should be loosely coordinated through a series of joint workshops. The STREPs should also include an educational component intended to disseminate the newly developed technologies.

3.8 Recommendation N°8 - VVUQ

Scientific perimeter

The recommendation presented here covers two neighbouring topics: VVUQ (Verification, Validation and Uncertainty Quantification) and optimisation.

The uncertainties in the numerical simulation process can arise from different sources:

- Lack of knowledge on a physical parameter (epistemic uncertainty),
- Parameter with a random nature (aleatory uncertainty),
- Uncertainty related to the model (model error),
- Uncertainty related to the numerical errors (numerical errors).

Taking into account these uncertainties is essential for the acceptance of numerical simulation for decision making. These uncertainties must be integrated in the verification and validation process of the simulation codes. This process is now commonly called VVUQ (Verification, Validation and Uncertainty Quantification). Verification consists in checking that the equations underlying the code are correctly solved. Validation is the stage during which the predictive capability of the numerical model is checked against experimental data or a reference model. Uncertainty quantification consists in defining the uncertainties that taunt the output of the simulation code.

Optimization is strongly linked to the VVUQ process because it is also a tool adding value to the simulation process for decision making, and also because the uncertainty analysis relies on optimization for many of its steps (model calibration, definition of surrogate models). The two subjects also meet on the field of stochastic optimisation.

The two topics also share the same position at the top of the software stack: their usage can be achieved with little or no impact on the codes that are used for the modelling. They therefore share similar issues in terms of deployment on Exascale machines.

Motivations

The mathematical background behind uncertainty analysis is very strong and comes from the field of statistics. Europe can claim world leading experts on these topics, but the link between this community and the HPC community must be strengthened.

The project aims at:

- Preparing VVUQ and optimisation software for Exascale computing by identifying and solving problems limiting the usability of these tools on many-core configurations,
- Facilitating access to the VVUQ techniques to the HPC community by providing software that is ready for deployment on supercomputers,
- Making methodological progresses on the VVUQ and optimisation methods for very large computations.

Impact (scientific, technological, societal & environmental)

VVUQ and optimisation give immense added value to the simulation process in terms of decision making. Verification and validation must definitely be addressed in order to define precisely what is the domain of predictability of a simulation tool. Uncertainty quantification gives the estimation of the error on the simulation results and points to the input uncertainties that should be reduced in order to improve the results, thus focusing the need for finer reference simulations or experimental data. Optimisation is obviously a major stake for the design of new systems.

The expected impact of the project is therefore to enhance the scientific recognition and societal acceptance of Exascale simulation by defining the predictability of the tools.

Level of innovation

The project innovation is on two levels:

- Maturation of existing VVUQ and optimisation tools for taking into account the constraints of Exascale computing
- R&D on numerical techniques for handling very large computations

Mesoscale detailed description of the proposal

SP1 – Ultrascalable tools for VVUQ and optimisation

➔ Optimisation
Linear Programming (LP) being the main building block for more complex problems such as nonlinear programming or mixed integer programming, a particular effort should be made on the research of LP scalable algorithms.
Optimisation under uncertainties will also be investigated: while promising results exist on linear problems, much work remains for nonlinear and mixed integer problems.

➔ Verification tools
Verification procedures cover two aspects: software unit tests and mathematical verification of the numerical resolution. For unit tests, the software tools that are used mostly come from the broad software engineering community, and do not take into account the specific needs of Exascale computing. The proposal is to improve those tools to make them suitable.

➔ Validation and UQ
The VVUQ tools should evolve to make use of the different levels of parallelism in a smooth manner: batch scheduling system, distributed parallelism, in-node parallelism should combine to deploy easily VVUQ methodology on codes that have different parallelism strategies, and on coupled simulations involving several codes.

SP2 – Accessibility of the software

➔ Numerical/Software improvements on VVUQ & optimisation tools to facilitate 'black-box' usage and therefore facilitate the dissemination of techniques on all software, be it academic or industrial, open source or commercial.

➔ Setting up a European portal for VVUQ and optimisation software, with deployment of the software on the PRACE infrastructures and efficiency benchmarks.

SP3 – Methodological progresses

➔ Model errors in the validation process
Most of the Validation and Uncertainty Quantification techniques make the assumption that numerical models are perfect and propagate parametric uncertainties, be they of aleatory or epistemic nature. Emerging techniques provide solutions for improved predictability of the simulations taking into account the existence of model errors.

➔ Surrogate models and reduced basis models
These models used to replace actual codes with less computationally intensive numerical models are essential elements for uncertainty analysis and optimisation of large simulations. The work will consist in improving the learning stages of these models by taking into account the objective (finding an extremum, defining a failure point, evaluating uncertainty) at the learning stage.

Type of recommendation/Proposal for the next step

An IP over 4 years could host this project. It should mean a 15 million Euros budget, 6M on SP1, 6M on SP2, and 3M on SP3. Deliverables will come mostly in the form of Open Source software, ready for use for the scientific and industrial community.

