EESI WG4.4
Scientific software engineering

Chair: Mike Ashworth, STFC
Speaker and Co-Chair: Andrew Jones, NAG
# List of experts

<table>
<thead>
<tr>
<th>Name</th>
<th>Organisation</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mike Ashworth (Chair)</td>
<td>STFC</td>
<td>UK</td>
</tr>
<tr>
<td>Andrew Jones (Co-Chair)</td>
<td>NAG</td>
<td>UK</td>
</tr>
<tr>
<td>Sebastian von Alfthan</td>
<td>CSC</td>
<td>FI</td>
</tr>
<tr>
<td>John Biddiscombe</td>
<td>CSCS</td>
<td>CH</td>
</tr>
<tr>
<td>Ian Bush</td>
<td>NAG</td>
<td>UK</td>
</tr>
<tr>
<td>Iris Christadler</td>
<td>LRZ</td>
<td>DE</td>
</tr>
<tr>
<td>David Emerson</td>
<td>STFC</td>
<td>UK</td>
</tr>
<tr>
<td>Rupert Ford</td>
<td>University of Manchester</td>
<td>UK</td>
</tr>
<tr>
<td>Yvan Fournier</td>
<td>EDF</td>
<td>FR</td>
</tr>
<tr>
<td>Chris Greenough</td>
<td>STFC</td>
<td>UK</td>
</tr>
<tr>
<td>Joachim Hein</td>
<td>Lund University</td>
<td>SE</td>
</tr>
<tr>
<td>Mohammad Jowker</td>
<td>BSC</td>
<td>ES</td>
</tr>
<tr>
<td>Peter Michielse</td>
<td>NWO</td>
<td>NL</td>
</tr>
<tr>
<td>Stephen Pickles</td>
<td>STFC</td>
<td>UK</td>
</tr>
<tr>
<td>Stephane Ploix</td>
<td>EDF</td>
<td>FR</td>
</tr>
<tr>
<td>Felix Schuermann</td>
<td>EPFL</td>
<td>CH</td>
</tr>
<tr>
<td>Ash Vadgama</td>
<td>AWE</td>
<td>UK</td>
</tr>
</tbody>
</table>
Typical HPC hardware and software universe

Scientific Applications

Numerical Libraries, Solvers and Algorithms

System Software
Eco-system

Parallel Programming Environment and Tools
Operating System, File System, System Mgmt.

HPC System (including compute nodes, interconnect, storage)
Identify the application development roadmap to achieve the Grand Challenges built on the emerging software, hardware and library infrastructure

- Coordinate closely with WGs 3.1-3.4 which define the Grand Challenges
- Coordinate closely with WGs 4.1-4.3 which set out the hardware, software, library and algorithm roadmaps

What is needed is to bring together the scientific and implementation aspects in reliable and sustainable codes
WG4.4: Specific challenges

### End-to-end simulation
- Discretisation (affects scalability and load balancing)
- Data management
- Visualization

### Complexity of software
- Multi-physics, multi-scale
- Multiple components
- Frameworks and workflows
- Multi-disciplinary teams

### Complexity of hardware
- Billion-way parallelism
- Heterogeneous processors
- Complex multi-level hierarchy
- Limited memory, bandwidth etc
WG4.4: Themes

- Application frameworks & workflows
  - End-to-end solutions; coupled models; etc.

- Visualisation & data management
  - Data issues; in-situ visualisation; etc.

- Fault tolerant algorithms
  - Build on API from OS; interface with WG4.2

- Application design
  - Build on prog lang & env; interface with WG4.2

- Software engineering
  - Software quality; collaborative working
11 priority research milestones in scientific software engineering

- Development of **Domain Specific Languages (DSL)** targeting specific application areas, e.g. climate, engineering, materials. High-level abstraction enables application development to be protected and insulated from architectural issues at the Exascale.

- Development of **open standards in coupling software and data exchange**, with training/education.

- Integration of **coupling and workflow technologies** with the development of efficient and generic coupling/workflow software for Exascale architectures.
Development of **advanced mesh generation, mesh partitioning and load balancing software** for billion-cell meshes targeting industrial problems.

Development of a **flexible generic I/O layer** that can be used by applications to interface with either the storage system or the data analysis system. This layer should then be extended with advanced data reduction techniques to carry out in-situ domain-specific data reduction and feature extraction.

Development of **advanced computational steering technologies** and their widespread integration into applications in order to improve resources management and time-to-solution.
Local checkpoint/restart to fast memory should be sufficient for some time on multi-Petascale systems; we need developments to bring this into common usage. Beyond that applications will need to become intrinsically fault tolerant.

Development of a global open standard for an API for error signalling and reporting – Europe should take a lead in this e.g. through collaboration of application development groups with European vendor R&D Labs

Development of fault tolerant applications across a range of key scientific areas, using MPI-3 initially, but adopting new APIs as they become available
Development of **Exascale applications**, new codes as well as evolution of existing codes, should be carried out by multi-disciplinary teams; collaborating in a process known in the USA as *co-design*. Europe needs to build the structures and encourage a culture for establishing cooperative working and critical masses of expertise in key domain sciences.

Development of **Advanced Integrated Development Environments** for HPC systems at all Tiers, incorporating capabilities in editing, code-checking, compilation, debugging, performance analysis, testing, verification, documentation-generation and collaborative working.
WG4.4: Costs

- Detailed costs have been estimated for each of the 11 recommended R&D activities
- Some, e.g. development of standards, should be complete by 2015 so that they can be exploited in Exascale application development
- Total from 2012-2020 is around €200M
Scientific Software Engineering is crucial for the efficient and effective exploitation of investments in HPC at all levels, especially Exascale, e.g.:

- **Coupling**: single apps may not be able to exploit Exascale; we need coupled apps or ensembles
- **Visualization and data management** are key for handling the data deluge from Exascale systems
- **Fault tolerant applications**: without them simulations will waste time resulting in very poor effective cost
- **Application design and software engineering**: complexity of apps and need for new apps makes these key issues through the software lifetime
Europe has a strong track record of innovation and excellence in scientific software R&D

- **Open source software** will be an important factor in future competitiveness (requirement in EC calls)
- Organisation in **multi-disciplinary teams**: strengths e.g. in PRACE, HP2C, Hartree, Jülich Sim Labs, Maison de la Simulation
- **Industry commitment** patchy and limited to traditional sectors
- **Limited prototype hardware access**: limited influence on design of next-generation hardware (participation in the co-design process)
Multi-disciplinary teams for new app R&D are crucially dependent on availability of the necessary skills and expertise in many areas

- A young, and growing, community of researchers who are able to exploit current and future leading-edge computing to the fullest possible extent
- Training in use of current leading-edge computers; and in writing new codes for new architectures
- A new generation of researchers: to realise the potential of HPC in a research environment that is increasingly interdisciplinary and multi-scale
WG4.4 Collaborations outside Europe

- Much software R&D is already international
  - European-led codes are developed with contributions from the US and elsewhere and vice versa
  - **Globalisation** of software development: Europe should embrace and participate in, contributing from a position of strength
  - An increasingly **international focus** is particularly relevant for global industries
  - **Open standards**: standardisation must be an international process; Europe should have a strong voice (already seen in MPI and OpenMP)
In addition to the 11 recommendations:

- There are significant potential societal and economic impacts in the successful adoption and exploitation of High Performance Computing and Computational Science and Engineering.

- It cannot be over-emphasised that the expected impact and the return from investment in Exascale systems will not be realised without a corresponding and appropriate level of investment in the enabling and supporting software technologies described here and in the reports from Working Groups 4.2 and 4.3.
THE END

IESP Meeting, 5-7 Oct. 2011, Cologne