



FP7 Support Action - European Exascale Software Initiative

DG Information Society and the unit e-Infrastructures



Software Ecosystem

Franck Cappello, INRIA and University of Illinois

Bernd Mohr, JSC

François Bodin, Marc Bull, Toni Cortés, Torsten Höfler, Jesus Labarta, Jacques C. Lafoucriere, David Lecomber, Arnaud Legrand, Thomas Ludwig, Simon McIntosh-Smith, Jean-François Méhaut, Matthias Müller, Raymond Namyst, Peter Niessen, Olivier Richard, Pascale Rossé, Karl Solchenbach, Vladimir Voevodin, Felix Wolf



Software Ecosystem Scope

Scientific Applications

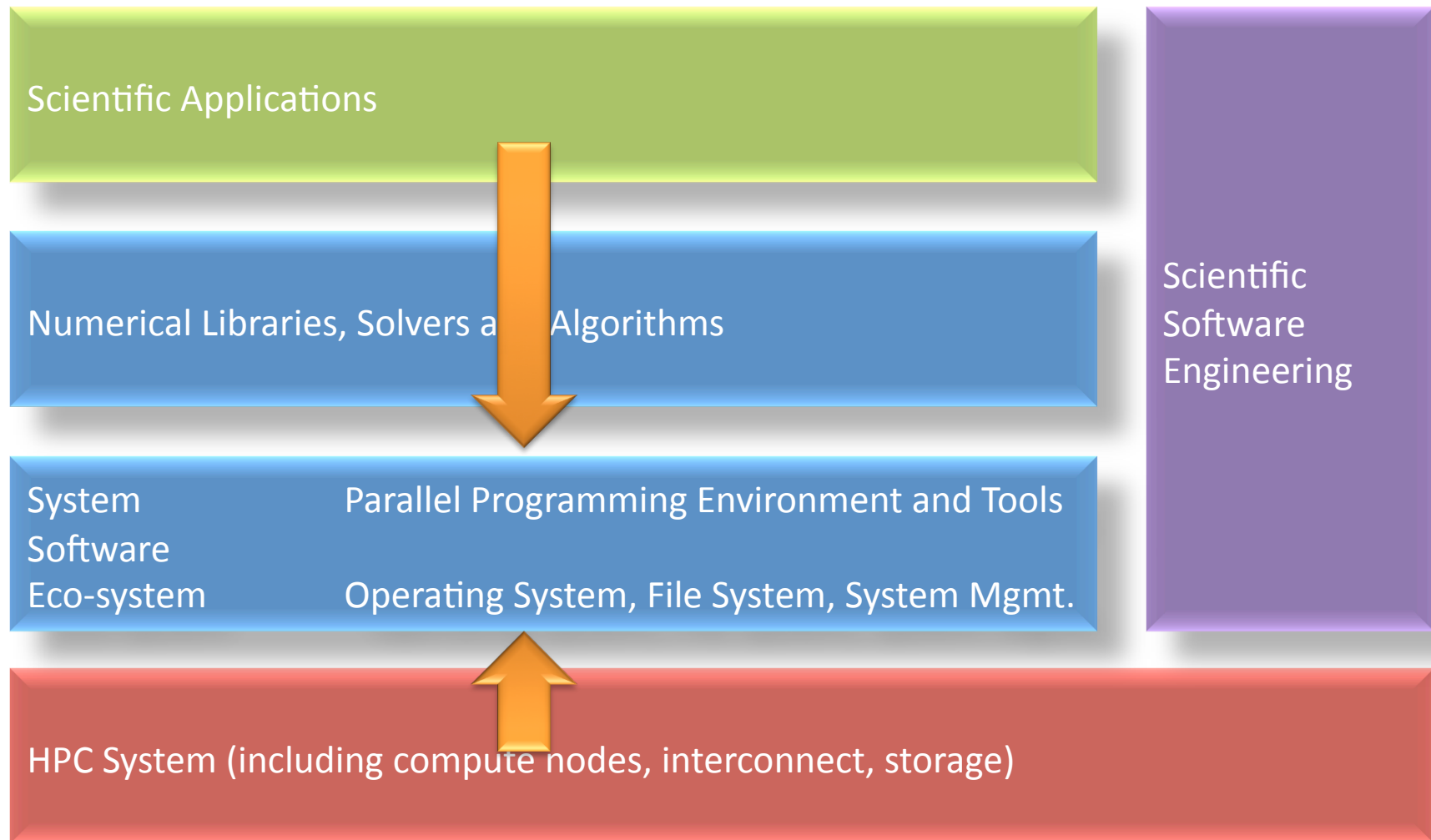
Numerical Libraries, Solvers and Algorithms

System	Parallel Programming Environment
Software	Tools
Eco-system	System

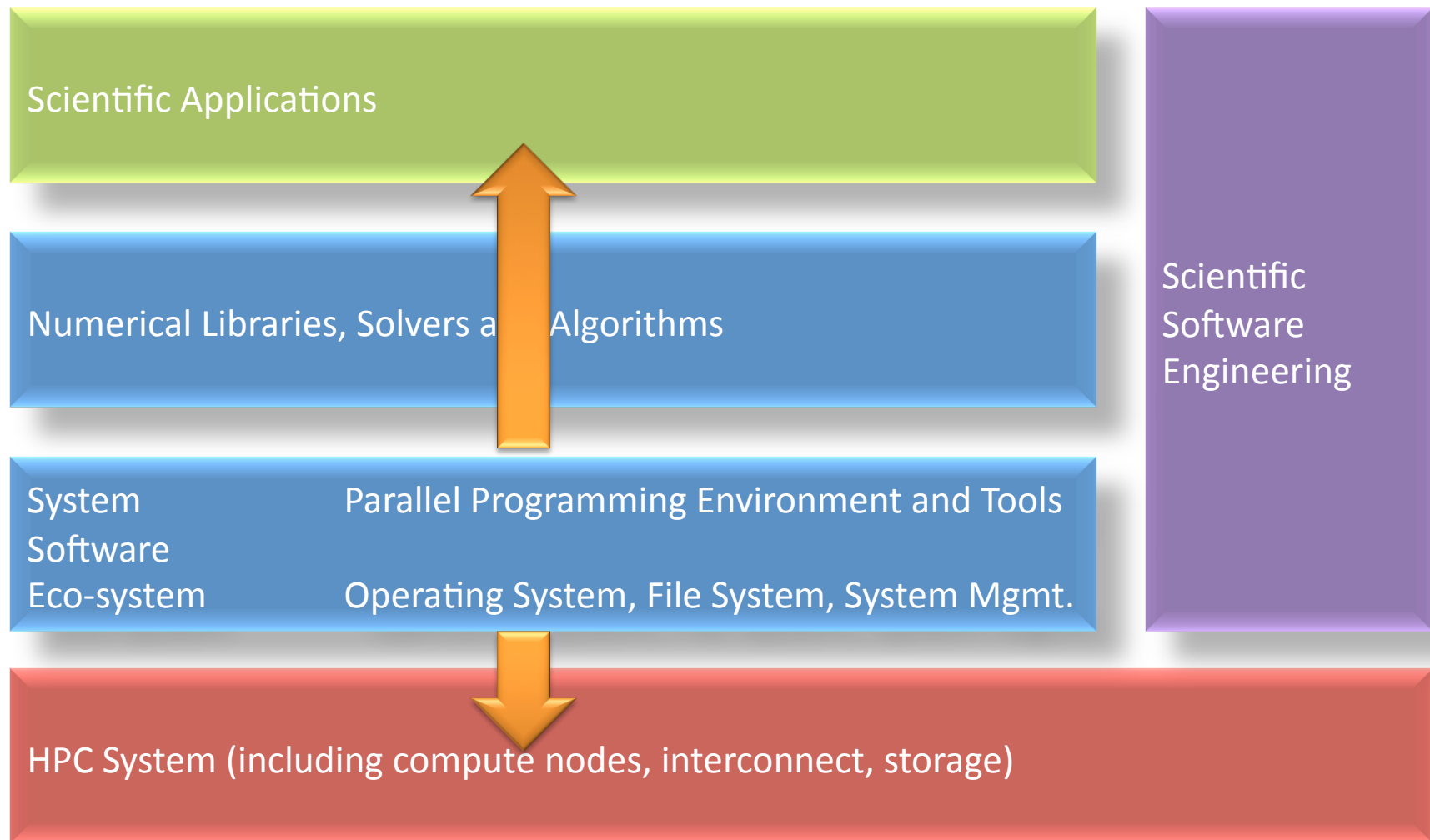
Scientific
Software
Engineering

HPC System (including compute nodes, interconnect, storage)

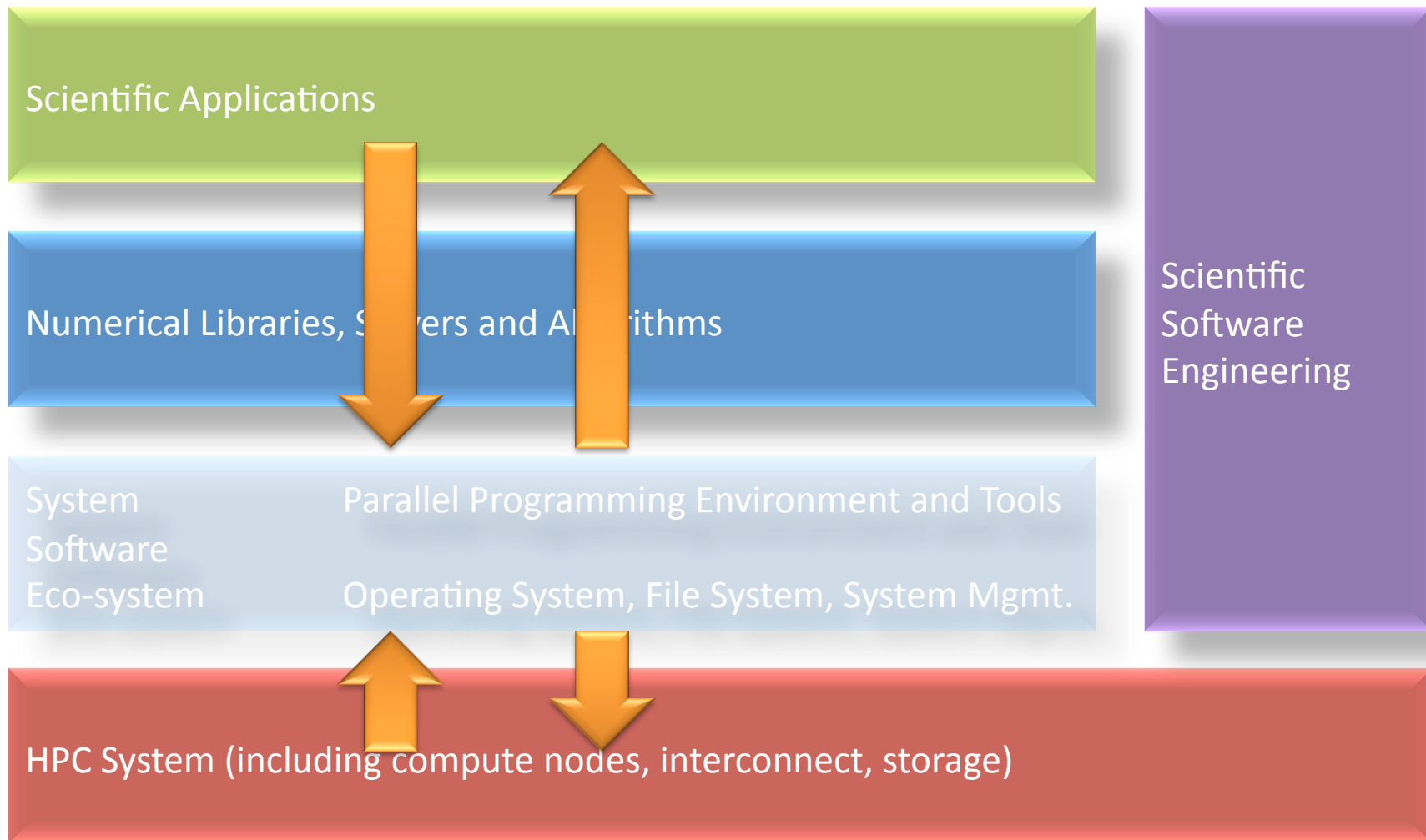
Software Ecosystem Criticality



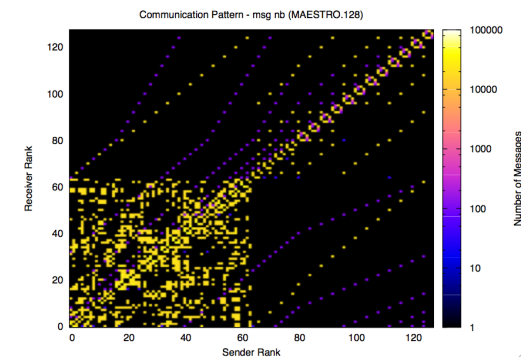
Software Ecosystem Criticality



Software Ecosystem Criticality



Software Ecosystem Topics



Software Ecosystem Challenges

- ❑ Exascale systems → very different than current HPC systems,
- ❑ Building, Operating and using Exascale systems will face severe challenges.
- ❑ Challenges cannot be addressed only at the hardware level.
- ❑ System software developers have to address these challenges, the most important ones being:
 - ❑ Scalability
 - ❑ Heterogeneity
 - ❑ Energy
 - ❑ Resilience
 - ❑ Performance
- ❑ A trend toward community software

Software Ecosystem Chairs, Leaders and Experts


Chairs:

□ Franck Cappello	INRIA&UIUC,	1 FR,	Resilience and FT
□ Benrd Mohr,	JSC,	1 DE,	Performance tools
□ Jesus Labarta	BSC	1 ES	Programming models
□ Marc Bull	EPCC	1 UK	OpenMP / PGAS
□ François Bodin	CAPS	2 FR	Programming/Compiler for GPUs
□ Raymond Namyst	LABRI Bordeaux,	3 FR	MPI&OpenMP Runtime , GPUs
□ Jean-François Méhaut	INRIA Grenoble	4 FR	Performance Modeling/Apps.
□ Matthias Müller	TU Dresden	2 DE	Validation/correctness Checking
□ Felix Wolf	GRS	3 DE	Performance Tools
□ David Lecomber	ALINEA	2 UK	Parallel debugger
□ Simon McIntosh-Smith	U. Bristol	3 UK	Computer Architecture & FT
□ Vladimir Voevodin	MSU	1 RU	Performance tools
□ Thomas Ludwig	DKRZ	5 DE	Power management
□ Olivier Richard	INRIA	5 FR	Job and Resource Manager
□ Jacques C. Lafoucriere	CEA	6 FR	I/O, File system
□ Toni Cortés	BSC	2 ES	I/O, Storage
□ Pascale Rossé	BULL	7 FR	OS, System management
□ Karl. Solchenbach	Intel	1 BE	All
□ Torsten Höfler	NCSA	1 US	Performance modeling
□ Peter Niessen	ParTec	4 DE	Batch systems


Software Ecosystem roadmap

- A 50 pages documents
- Written by top European experts/leaders
- Starting from IESP roadmap, Exascale studies, Soft. Used in HPC centers
- Listing Challenges
- Exploring Solutions
- Considering Crosscutting issues
- Investigating Social benefits, environmental and economical impact
- Estimating time line and needed MenMonths efforts

EESI Final Conference,
10-11 Oct. 2011, Barcelona



European Exascale Software Initiative



D4.4 Working Group report on software ecosystem

CONTRACT NO
INSTRUMENT
THEMATIC

EESI 261513
CSA (Support and Collaborative Action)
INFRASTRUCTURE

Due date of deliverable: 31 May 2011
Actual submission date: 9 September 2011
Publication date: N/A

Start date of project: 1 JUNE 2010
Duration: 18 months

Name of lead contractor for this deliverable: GENCI
Authors : Franck Cappello (INRIA-UIUC) and Bernd Mohr (JUELICH)
Name of reviewers for this deliverable: Bernd Mohr (JUELICH)

Abstract: This report is a compilation of main challenges and a development roadmap of the system software stack for an exascale system covering OS, runtime environments, I/O systems, system management, programming models and languages, Framework, compilers, libraries and tools.

Revision R1.1

Project co-funded by the European Commission within the Seventh Framework Programme (FP7/2007-2013)		
Dissemination Level		
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)	

Software Ecosystem Key Findings

- ❑ Most HPC system software components have to be **substantially adapted or newly developed** to address Exascale challenges
 - A 1000 MY investment in system software development is necessary: 100 persons during 10 years.
- ❑ Application developers (in research and in industry) are reluctant to adopt new, not yet fully accepted and standardized programming models
 - **A transition path for HPC applications** to Exascale programming models is required
- ❑ The different Exascale system architectures should not be visible at the level of the programming model
 - **The SW ecosystem must be portable** to support various Exascale hardware
- ❑ Collaboration between HPC hardware, system vendors, European R&D laboratories and Academic researchers has to be ensured
 - **Co-design processes must be established**

Software Ecosystem Key Findings

- Significant funding is essential in R&D areas where Europe has technology leadership and critical mass: programming models & runtime, performance, validation and correctness tools.
 - The key players should form alliances to define standards.
- Establish cooperation programs with technology leaders in areas where European technology is not leading: I/O, File systems, OS, etc.
 - To ensure that system components and standard can be influenced from a European perspective
 - To revitalize research in these important areas in Europe
- Consistency and completeness of the software stack as well as portability, robustness, and proper documentation have to be ensures
 - **European Exascale Software Centre** should be established to coordinate research and development of HPC Exascale software ecosystem components developed in National and EU projects,

Key findings: Testbed needs in 2015

	< 100 Petaflops enough	≥ 100 Petaflops needed
Dedicated/ reconfigurable	<ul style="list-style-type: none"> -Prog. Models and Runtime -Performance tools -OS -I/O-File system -Job and resource manager 	<ul style="list-style-type: none"> -Performance tools -Prog. Models and Runtime
Production	<ul style="list-style-type: none"> -Compilers -Programming models -Performance tools -Power management -Validation and correctness checking 	<ul style="list-style-type: none"> -Resilience -Parallel debuggers -Performance tools -Runtime -Prog. Models -Performance modeling at scale

IESP Additional slide: Recommendations By topics

- **Programming models** that provide the right balance between expressiveness, portability, performance and transition path for applications
- **Unified runtime system**, providing a unified API to deal with threads and lightweight tasks (together with their integration with MPI/PGAS communication systems)
- Vertical integration of **validation and correctness** at all layers of software stack
- Many of **the performance tools** used in production today are made in Europe, sustainability of funding must be ensured to maintain leadership
- EU must be a major contributor/partner to **Open Source file system**
- Hardware initiative in Europe around ARM, AMD Fusion, Nvidia Denver, etc. should be complemented by an **OS initiative**
- Leverage European leadership in energy efficient hardware design and Know-how from embedded system to develop **power management techniques**
- Hardware and Applications will not **mask/tolerate all faults** efficiently. Developing/maintaining system software for fault tolerance is critical

IESP Additional slide: Testbed needs in 2015



< 100 Petaflops enough

≥ 100 Petaflops needed

Dedicated/ reconfigurable	<p>-Programming models and Runtime: Interaction of runtime with kernel scheduler</p> <p>-Performance tools: Interaction with resilience / power components</p> <p>-OS (Scheduling, memory mgnt.): Measure noise generation (could also apply to other parts of the system software)</p> <p>-I/O-File system: May be done at lower scale, but needs root access for reconfiguration</p> <p>-Job and resource manager: Needs also root access for reconfiguration</p>	<p>-Performance tools (system level): Measure and minimize overheads induced at the full scale by low-level performance monitoring infrastructure tightly integrated with OS</p> <p>-Programming models and Runtime: Interaction of runtime with kernel scheduler</p>
	<p>-Compilers: Node level</p> <p>Programming models and Runtime: node level API</p> <p>-Performance tools: Node level</p> <p>Power management</p> <p>Validation and correctness checking: Node level</p>	<p>-Resilience: FT protocols, ABFT, NFTA, execution state storage</p> <p>-Debuggers, Validation and correctness checking: scalability tests</p> <p>-Performance tools: Scalability of data collection and analysis</p> <p>-Programming models and runtime: System level scalability</p> <p>-Performance modeling at scale</p>

IESP Additional slide: Development/maintenance



- Having an eco-system is essential to develop relevant system software:
 - How to initiate and operate interactions between users, application developers, hardware providers and system software developers?
- Software engineering:
 - Do we need to establish a set of key applications that will serve as references for system software certification
 - Do we need to establish procedure for system software integration / retirement?
 - What kind of hardware systems are needed to develop/assess/certify/maintain the system software (in addition to production ones)?
- Funding:
 - What level of funding is needed for software maintenance? (EESI: 1000MY for development until 2018 for a stack)
 - What is the right funding model? A mix of secured funding for key components and competitive funding for others?

IESP Additional slide: About International relations



- One potential scenario is one Exascale stack per major country. Each stack would probably needs to be consistent and complete
- What is the right cooperation model?
- How a local stack would integrate software modules made by others?
- Shall a country delegate or replicate the software development effort in domains where it is not leading?
- What is the right cooperation framework from the developer point of view? forums like the MPI one? International funded projects like the G8, other? Mix of them?
- How developers can actually develop system software compliant with more than one stack? Do we need to define APIs?



Questions?

Franck Cappello, INRIA and University of Illinois

Bernd Mohr, JSC

François Bodin, Marc Bull, Toni Cortés, Torsten Höfler, Jesus Labarta, Jacques C. Lafoucriere, David Lecomber, Arnaud Legrand, Thomas Ludwig, Simon McIntosh-Smith, Jean-François Méhaut, Matthias Müller, Raymond Namyst, Peter Niessen, Olivier Richard, Pascale Rossé, Karl Solchenbach, Vladimir Voevodin, Felix Wolf