Applications Breakout Group:

"What Apps want from HW and SW"

IESP Workshop

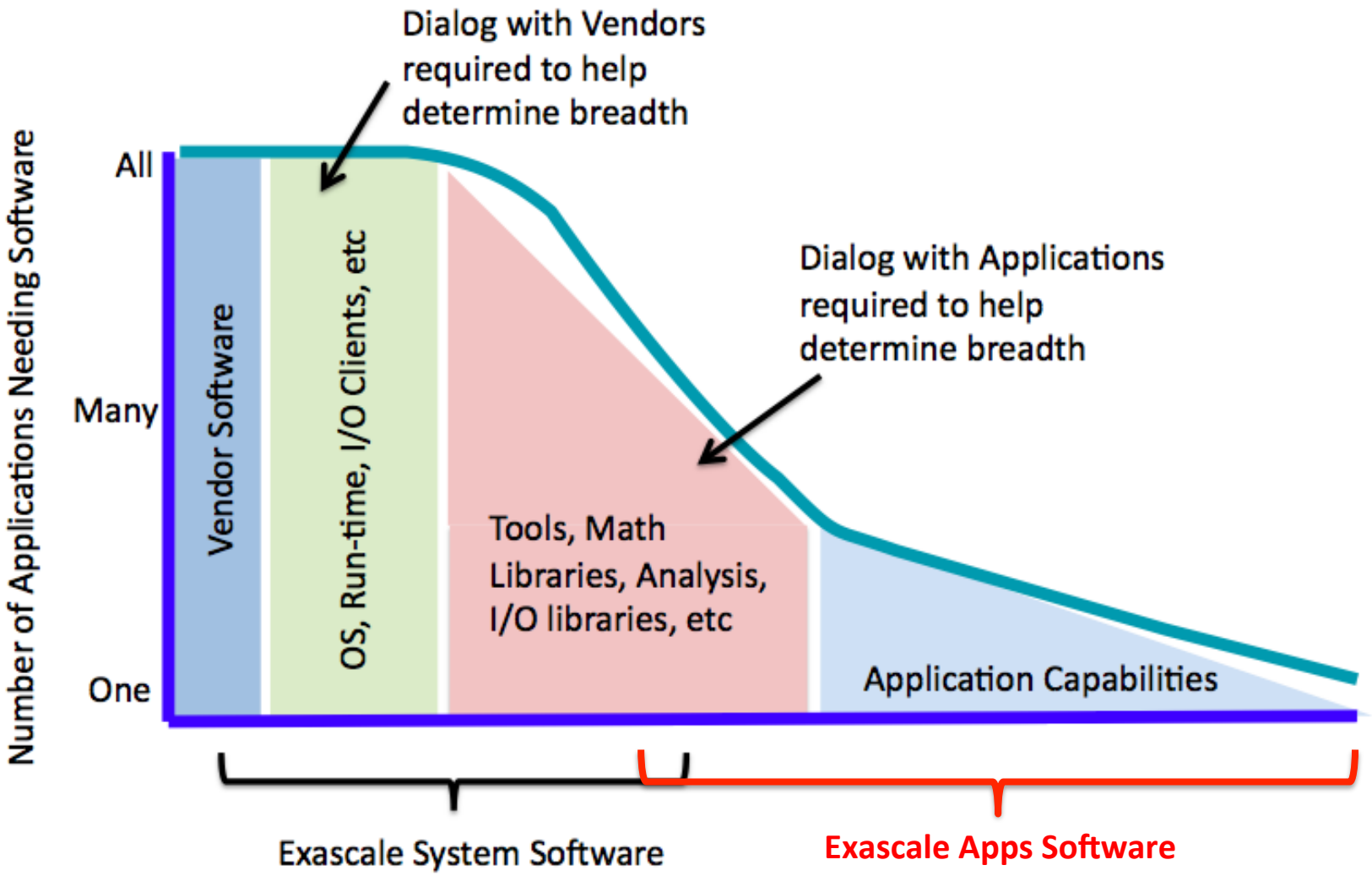
San Francisco, 6-7 April 2011

David Keyes, facilitator

Apps breakout participants

- Giovanni Aloisio, CMCC
- Jean-Claude Andre, CERFACS
- Michael Ashworth, Daresbury
- Jean-Yves Berthou, EDF
- Arndt Bode, LRZ Munich
- Riccardo Brunino, CINECA
- Jackie Chen, SNL
- Tim Cornwell, CSIRO
- Anshu Dubey, U Chicago
- Ramon Goni, BSC
- Bill Gropp, UIUC
- Stefan Heinzel, MPI DEISA
- Ryutaro Himemo, RIKEN
- David Keyes, KAUST/Columbia
- Alice Koniges, LBNL
- Paul Mackenzie, Fermi
- Paul Messina, ANL
- Kengo Nakajima, U Tokyo
- Hiroshi Nakashima, Kyoto U
- Stefan Requena, GENCI
- Andrew Siegel, ANL
- Godehard Sutman, Juelich
- Bill Tang, PPPL
- John Taylor, CSIRO

Apps scope


Dialog with Vendors
required to help
determine breadth
Dialog with Applications
required to help
determine breadth
Number of Applications Needing Software
All
Many
One
Vendor Software
OS, Run-time, I/O Clients, etc
Tools, Math
Libraries, Analysis,
I/O libraries, etc
Application Capabilities
Exascale System Software
Exascale Apps Software

History of Apps Charge

- Oxford, 13-14 April 2010: "Assess the short-term, medium-term, and long-term software and algorithm needs of applications for peta/exascale systems"
  - http://www.exascale.org/mediawiki/images/8/86/BlueDots13April2010.pdf
- Maui, 18-19 October 2010 : "What do the apps teams need from the software and hardware sides?"
  - http://www.exascale.org/mediawiki/images/e/ee/AppsBreakoutMaui.pdf
- We structured our response in both cases in terms of "Co-Design Vehicles" (candidate apps to be first on the exascale systems)

We mainly changed the question:

“What have we heard and what should HW and SW want from us?”

What have we learned so far?

- The hardware picture seems to have "converged" to a configuration in which (smearing swim lanes)
  - we have 100-1000 threads per processor-memory unit
  - we lose, in a relative sense, factors of 10-100 in memory per thread (capital limit) and in memory bandwidth per thread (operational limit)
  - we give up floating point performance reliability, putting codes that synchronize frequently at risk for scaling
- The software picture has evolved to a set of development plans that are less revolutionary and more evolutionary, after all
  - combination of weak scaling through MPI, then strong scaling through threads
- Probably both pictures are ultimately optimistic, but they provide targets for apps planning today

How does this affect apps outlook?

- The software projects (ESSI, ESC, PP) will provide a layer above the hardware *and* a set of open source examples of how to code effectively for it in the form of numerical libraries
- Apps will be far more conscious of the unreliability and unreproducibility of hardware than ever before
- Performance-seeking apps (namely *all* exascale apps) will be substantially rewritten, *but* much of the hard work will be done and demonstrated by SW development experts

How does this affect apps outlook (cont)?

- Programming model of "MPI+X" (or "MPI⊙X") is comforting
  - new tools like "blameshifting", etc. will be very useful
- We will still have to recode almost all our algorithms to reduce their synchronization and increase their locality of data, to avoid an Amdahl bottleneck in the apps-authored part of the overall executable
  - but we can use a familiar hybrid programming model in which to implement the new code

What we need to provide is "exaskeletons":

sample implementations of basic "dynamical cores" (a step up from the 13 "motifs")

Skeletons do not have to give the right physical answers, but they should stress the HW and SW representatively; on the other hand, they should also be representative with respect to issues beyond floating point performance, like checkpointing

See co-design centers' skeleton apps

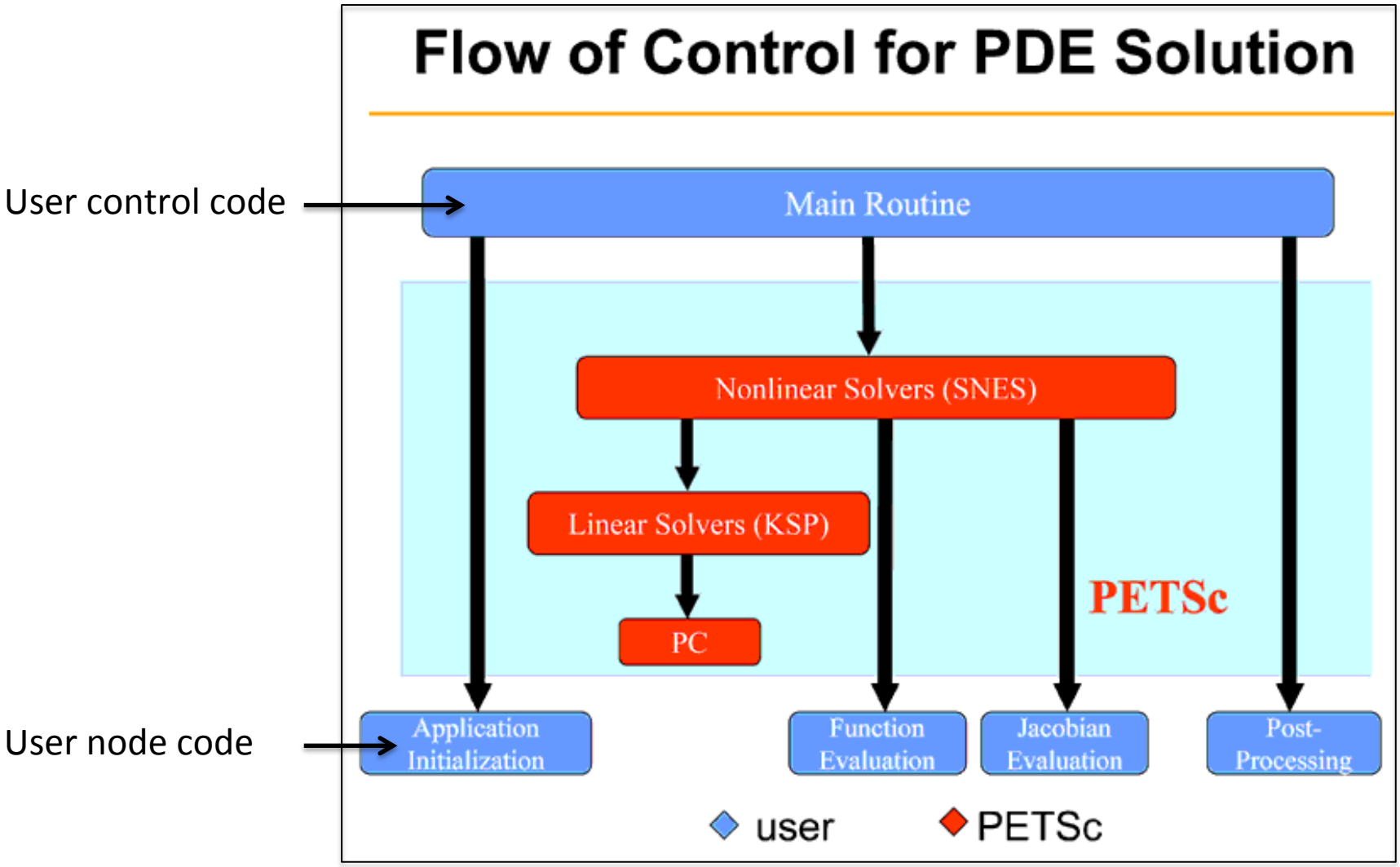
The 13 algorithmic motifs*

- **Dense direct solvers**
- **Sparse direct solvers**
- **Spectral methods**
- **N-body methods**
- **Structured grids / iterative solvers**
- **Unstructured grids / iterative solvers**
- **Monte Carlo ("MapReduce")**
- **Combinatorial logic**
- **Graph traversal**
- **Graphical models**
- **Finite state machines**
- **Dynamic programming**
- **Backtrack and branch-and-bound**

* *The Landscape of Parallel Computing Research: The View from Berkeley*, UCB/EECS-2006-183

Example of a SciDAC "petaskeleton"

- PETSc for PFLOTRAN http://ees.lanl.gov/pflotran/



Apps issues/responsibilities (algorithmic)

- Synchronization
  - Locate all synchronization points
  - Separate work between synchronizations into critical path and non-critical path parts
- Locality
  - Reduce data motion, both vertically and horizontally
- Arithmetic intensity
  - Consider high-order methods to swap many small blocks for fewer big blocks
  - Consider recomputation to avoid storage-retrieval bottlenecks
- Precision
  - Consider how to work with the deltas rather than the whole quantities
- I/O
  - Implement data triage

Apps issues/responsibilities (programming models)

- "Domain specific" interfaces for source-to-source transformations
  - Not exactly the right word
  - Prefer "data structure/algorithmic specific" interfaces
  - E.g., one for explicit PDEs, one for implicit PDEs, integral equations, particle methods, kinetic equations, data mining, etc.

Apps issues/responsibilities (political/social)

- Determine qualitative thresholds of scientific discovery in performance/capability that are crossed from peta to exa, for motivation and lobbying for resources
  - E.g., no. of particles, full dimensions, high resolution, no. of energy groups, etc.
- Help police false promises, for credibility of the campaign
  - Will entire workflows and multiphysics combinations really work from the data motion viewpoint?
- Critique completeness and relevance of IESP's apps inventory for own domain
- Provide compact/skeletal apps

What apps needs to provide to SW/ HW (through the exaskeletons?)

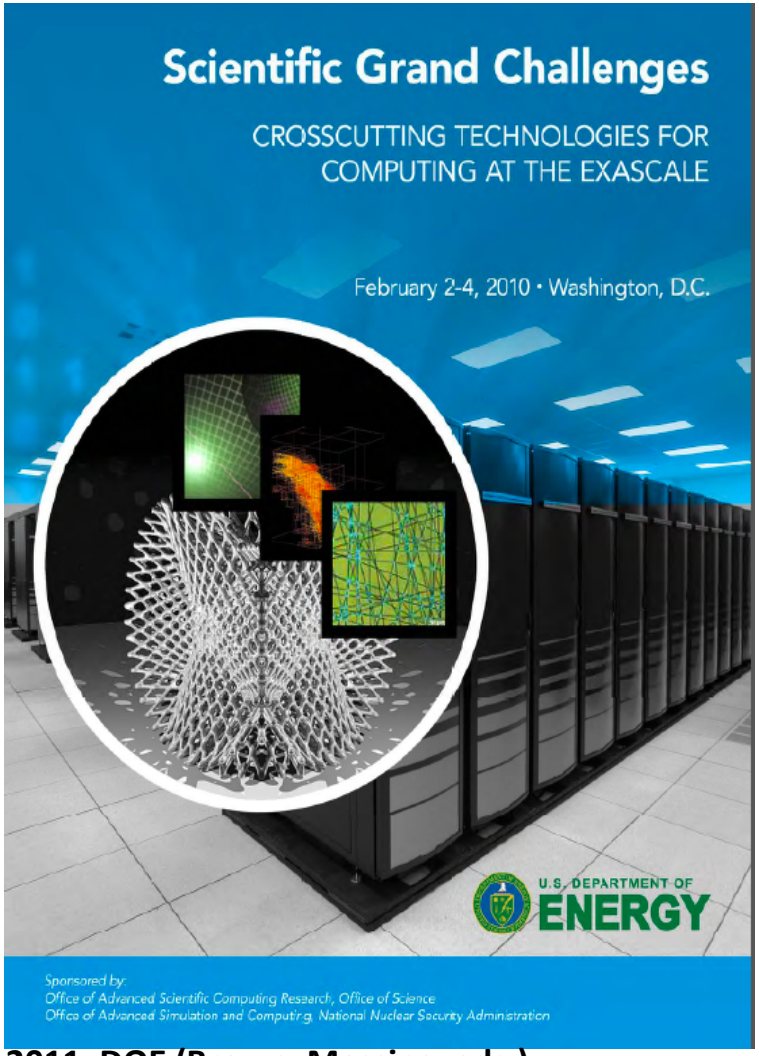
- Quantitative ideas about relative rates, sizes, frequencies, and granularities
- Not just flops and loads/stores, but things like branching frequency, what sorts of random accesses

What we want that we didn't hear (enough)

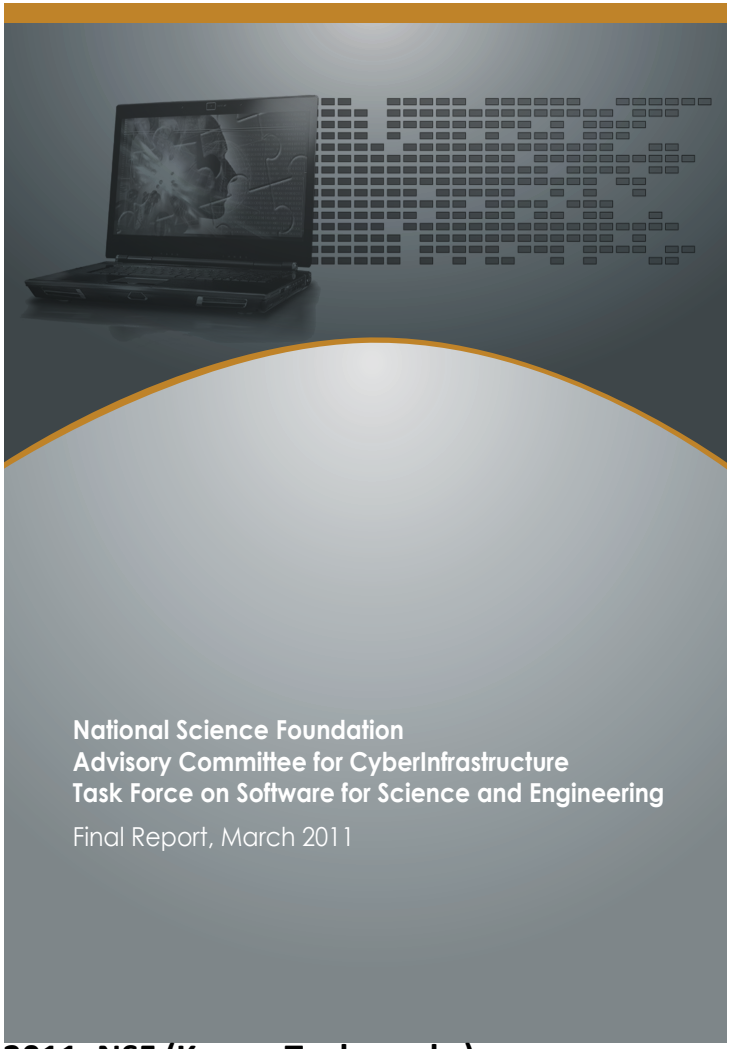
- Tools for development, not just production
- And the ability to turn them off ☺
- Tools for dynamic processor allocation (e.g., in multiphysics applications)
- Need to get more data-intensive applications into the application inventory
  - Need to hear more about exascale challenges from observational sciences, biosciences, etc.
  - Need to focus more on the entire vertically integrated process including input/output staging and pre/post-processing (not just simulation, but UQ, *in situ* analytics, etc.)
- A development environment that helps us make trade-off decisions about coding alternatives

For IESP audience interest …

New reports of interest to IESP people



2011, DOE (Brown, Messina, eds.)



2011, NSF (Keyes, Taylor, eds.)

Meeting of likely interest to IESP algorithms people

- Brown University ICERM workshop:
- *Synchronization-reducing and Communication-reducing algorithms and programming models for large-scale simulations*
- 9-13 January 2012
- Co-organizers: J. Hesthaven, D. Keyes, M. Knepley, K. Yelick