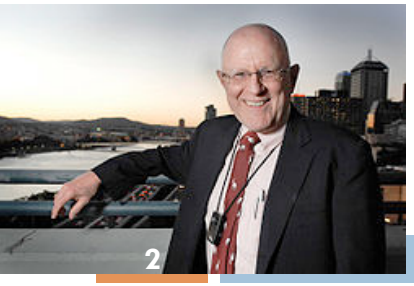
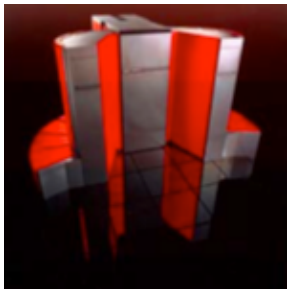
THE ROAD TO EXASCALE: HARDWARE AND SOFTWARE CHALLENGES

JACK DONGARRA
UNIVERSITY OF TENNESSEE
OAK RIDGE NATIONAL LAB



Looking at the Gordon Bell Prize

(Recognize outstanding achievement in high-performance computing applications and encourage development of parallel processing)

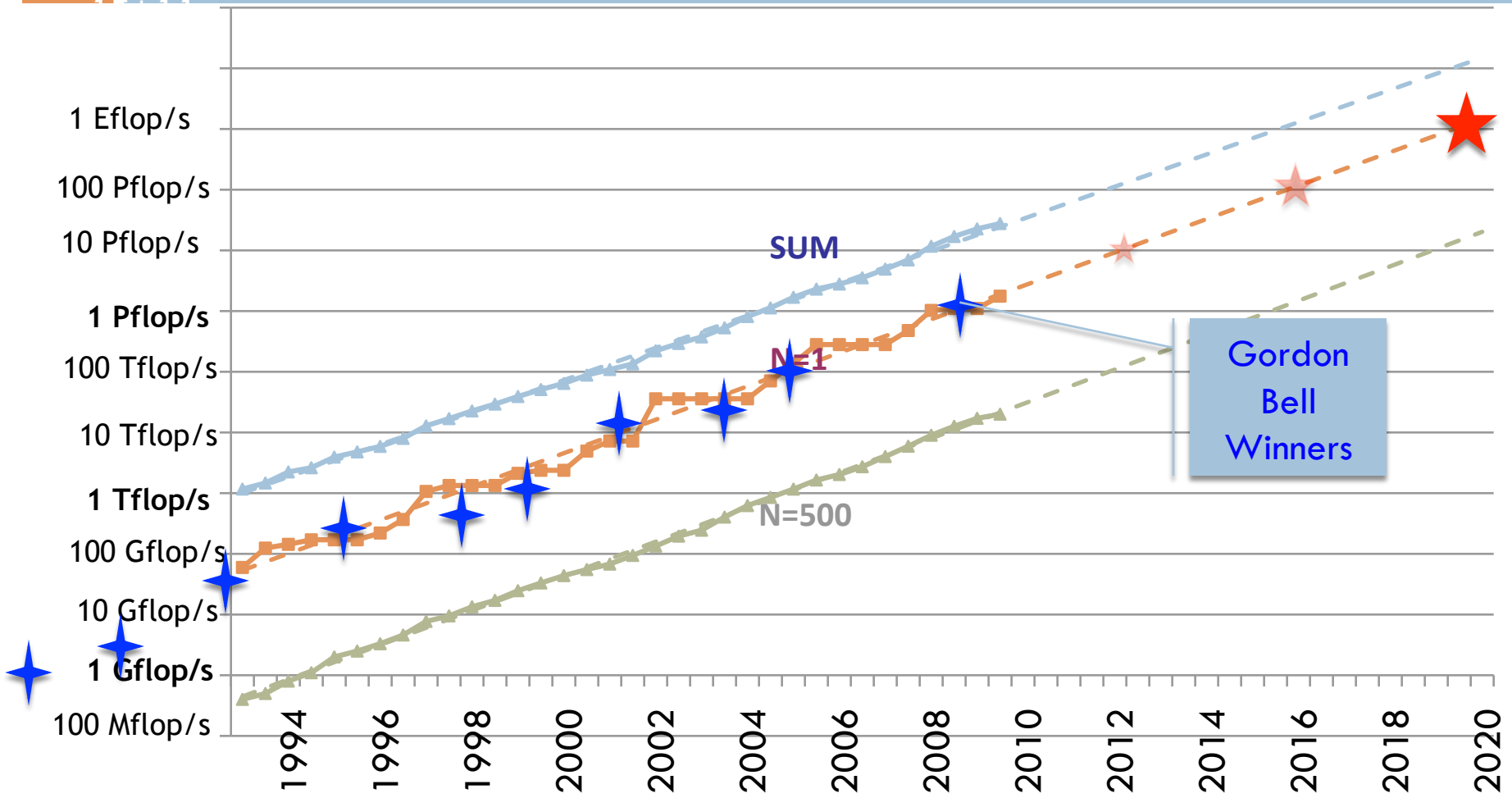


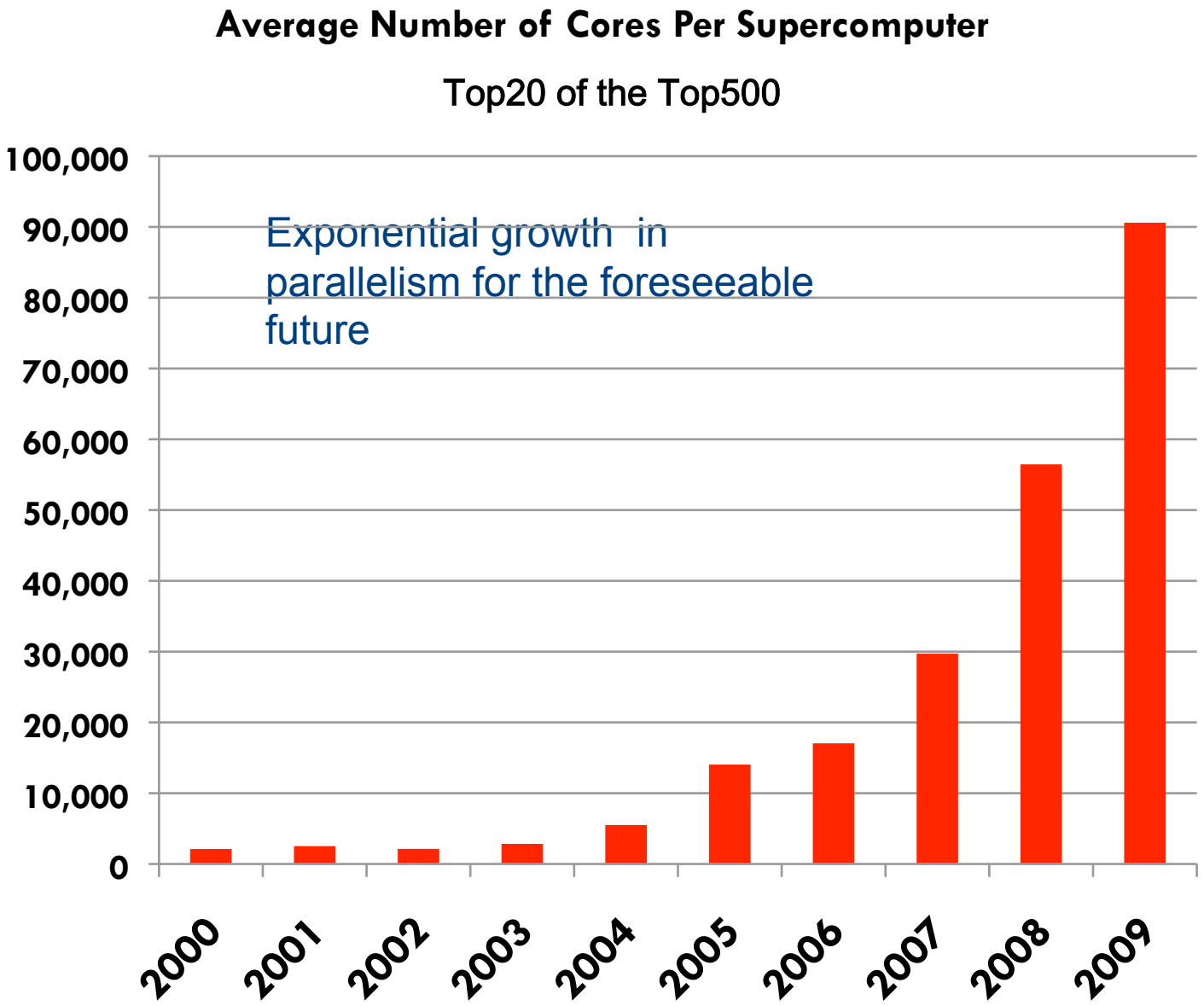
- 1 GFlop/s; 1988; Cray Y-MP; 8 Processors
  - Static finite element analysis

- 1 TFlop/s; 1998; Cray T3E; 1024 Processors
  - Modeling of metallic magnet atoms, using a variation of the locally self-consistent multiple scattering method.

- 1 PFlop/s; 2008; Cray XT5; 1.5×10^5 Processors
  - Superconductive materials
- 1 EFlop/s; ~2018; ?; 1×10^7 Processors (10^9 threads)

Performance Development in Top500




Average Number of Cores Per Supercomputer
Top20 of the Top500
100,000
90,000
80,000
70,000
60,000
50,000
40,000
30,000
20,000
10,000
0
Exponential growth in parallelism for the foreseeable future
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009

Factors that Necessitate Redesign

- **Steepness of the ascent from terascale to petascale to exascale**
- Extreme parallelism and hybrid design
  - Preparing for million/billion way parallelism
- Tightening memory/bandwidth bottleneck
  - Limits on power/clock speed implication on multicore
  - Reducing communication will become much more intense
  - Memory per core changes, byte-to-flop ratio will change
- Necessary Fault Tolerance
  - MTTF will drop
  - Checkpoint/restart has limitations

www.exascale.org

- **Software infrastructure does not exist today**

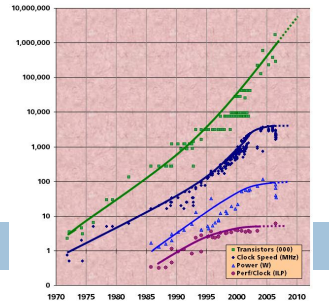
Major Changes to Software

- Must rethink the design of our software
  - Another disruptive technology
    - Similar to what happened with cluster computing and message passing
  - Rethink and rewrite the applications, algorithms, and software
- Numerical libraries for example will change
  - For example, both LAPACK and ScaLAPACK will undergo major changes to accommodate this

IESP: The Need

- The largest scale systems are becoming more complex, with designs supported by consortium
  - The software community has responded slowly
- Significant architectural changes evolving
  - Software must dramatically change
- Our ad hoc community coordinates poorly, both with other software components and with the vendors
  - Computational science could achieve more with improved development and coordination

A Call to Action



- Hardware has changed dramatically while software ecosystem has remained stagnant
- Previous approaches have not looked at co-design of multiple levels in the system software stack (OS, runtime, compiler, libraries, application frameworks)
- Need to exploit new hardware trends (e.g., manycore, heterogeneity) that cannot be handled by existing software stack, memory per socket trends
- Emerging software technologies exist, but have not been fully integrated with system software, e.g., UPC, Cilk, CUDA, HPCS
- Community codes unprepared for sea change in architectures
- No global evaluation of key missing components

International Community Effort

- We believe this needs to be an international collaboration for various reasons including:
  - The scale of investment
  - The need for international input on requirements
  - US, Europeans, Asians, and others are working on their own software that should be part of a larger vision for HPC.
  - No global evaluation of key missing components
  - Hardware features are uncoordinated with software development

IESP Goal

Improve the world's simulation and modeling capability by improving the coordination and development of the HPC software environment

Workshops:

Build an international plan for developing the next generation open source software for scientific high-performance computing

Key Trends

- Increasing Concurrency
- Reliability Challenging
- Power dominating designs
- Heterogeneity in a node
- I/O and Memory: ratios and breakthroughs

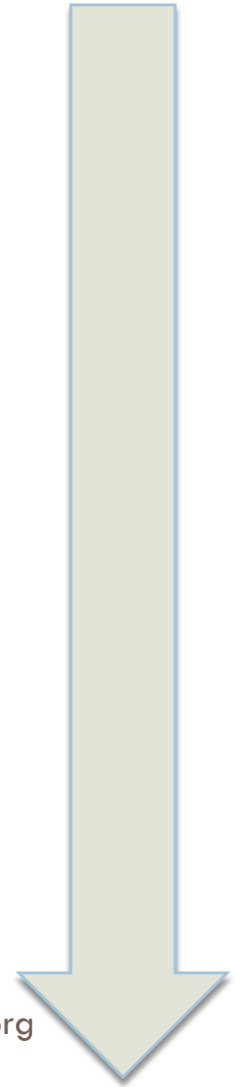
Requirements on X-Stack

- Programming models, applications, and tools must address concurrency
- Software and tools must manage power directly
- Software must be resilient
- Software must address change to heterogeneous nodes
- Software must be optimized for new Memory ratios and need to solve parallel I/O bottleneck

www.exascale.org Roadmap Components

Where We Are Today:

- SC08 (Austin TX) meeting to generate interest — **Nov 2008**
- Funding from DOE's Office of Science & NSF Office of Cyberinfratructure and sponsorship by Europeans and Asians — **Apr 2009**
- US meeting (Santa Fe, NM) April 6-8, 2009
  - 65 people
- NSF's Office of Cyberinfrastructure funding — **Jun 2009**
- European meeting (Paris, France) June 28-29, 2009
  - 70 people
  - Outline Report
- Asian meeting (Tsukuba Japan) October 18-20, 2009 — **Oct 2009**
  - Draft roadmap
  - Refine Report
- SC09 (Portland OR) BOF to inform others — **Nov 2009**
  - Public Comment
  - Draft Report presented



- www.exascale.org

Jack Dongarra
Pete Beckman
Terry Moore
Jean-Claude Andre
Jean-Yves Berthou
Taisuke Boku
Franck Cappello
Barbara Chapman
Xuebin Chi
Alok Choudhary
Sudip Dosanjh
Al Geist
Bill Gropp
Robert Harrison
Mark Hereld
Michael Heroux
Adolfy Hoisie
Koh Hotta
Yutaka Ishikawa
Fred Johnson
Sanjay Kale
Richard Kenway
Bill Kramer
Jesus Labarta
Bob Lucas
Barney Maccabe
Satoshi Matsuoka
Paul Messina
Bernd Mohr
Matthias Mueller
Wolfgang Nagel
Hiroshi Nakashima
Michael E. Papka
Dan Reed
Mitsuhisa Sato
Ed Seidel
John Shalf
David Skinner
Thomas Sterling
Rick Stevens
William Tang
John Taylor
Rajeev Thakur
Anne Trefethen
Marc Snir
Aad van der Steen
Fred Streitz
Bob Sugar
Shinji Sumimoto
Jeffrey Vetter
Robert Wisniewski
Kathy Yelick

SPONSORS

4.2.4 Numerical Libraries

- Technology drivers
  - Hybrid architectures
  - Programming models/ languages
  - Precision
  - Fault detection
  - Energy budget
  - Memory hierarchy
  - Standards
- Alternative R&D strategies
  - Message passing
  - Global address space
  - Message-driven work-queue
- Recommended research agenda
  - Hybrid and hierarchical based software (eg linear algebra split across multi-core / accelerator)
  - Autotuning
  - Fault oblivious sw, Error tolerant sw
  - Mixed arithmetic
  - Architectural aware libraries
  - Energy efficient implementation
  - Algorithms that minimize communications
- Crosscutting considerations
  - Performance
  - Fault tolerance
  - Power management
  - Arch characteristics

Priority Research Direction

Key challenges

- Adaptivity for architectural environment
- Scalability : need algorithms with minimal amount of communication
- Increasing the level of asynchronous behavior
- Fault resistant software– bit flipping and loosing data (due to failures). Algorithms that detect and carry on or detect and correct and carry on (for one or more)
- Heterogeneous architectures
- Languages
- Accumulation of round-off errors

Summary of research direction

- Fault oblivious, Error tolerant software
- Hybrid and hierarchical based algorithms (eg linear algebra split across multi-core and gpu, self-adapting)
- Mixed arithmetic
- Energy efficient algorithms
- Algorithms that minimize communications
- Autotuning based software
- Architectural aware algorithms/libraries
- Standardization activities
- Async methods
    - Overlap data and computation

Potential impact on software component

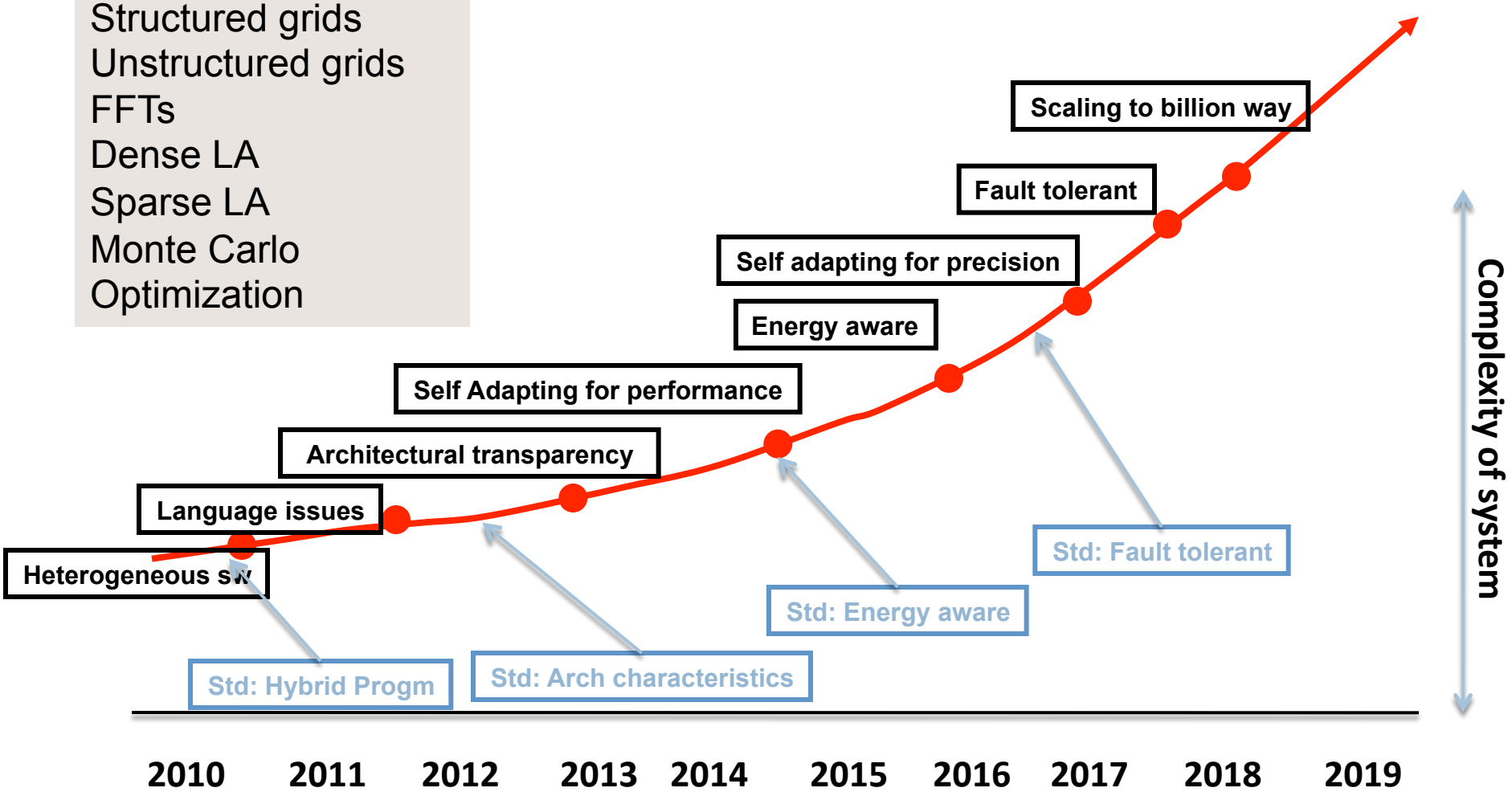
- **Efficient libraries of numerical routines**
- **Agnostic of platforms**
- **Self adapting to the environment**
- **Libraries will be impacted by compilers, OS, runtime, prog env etc**
- **Standards: FT, Power Management, Hybrid Programming, arch characteristics**

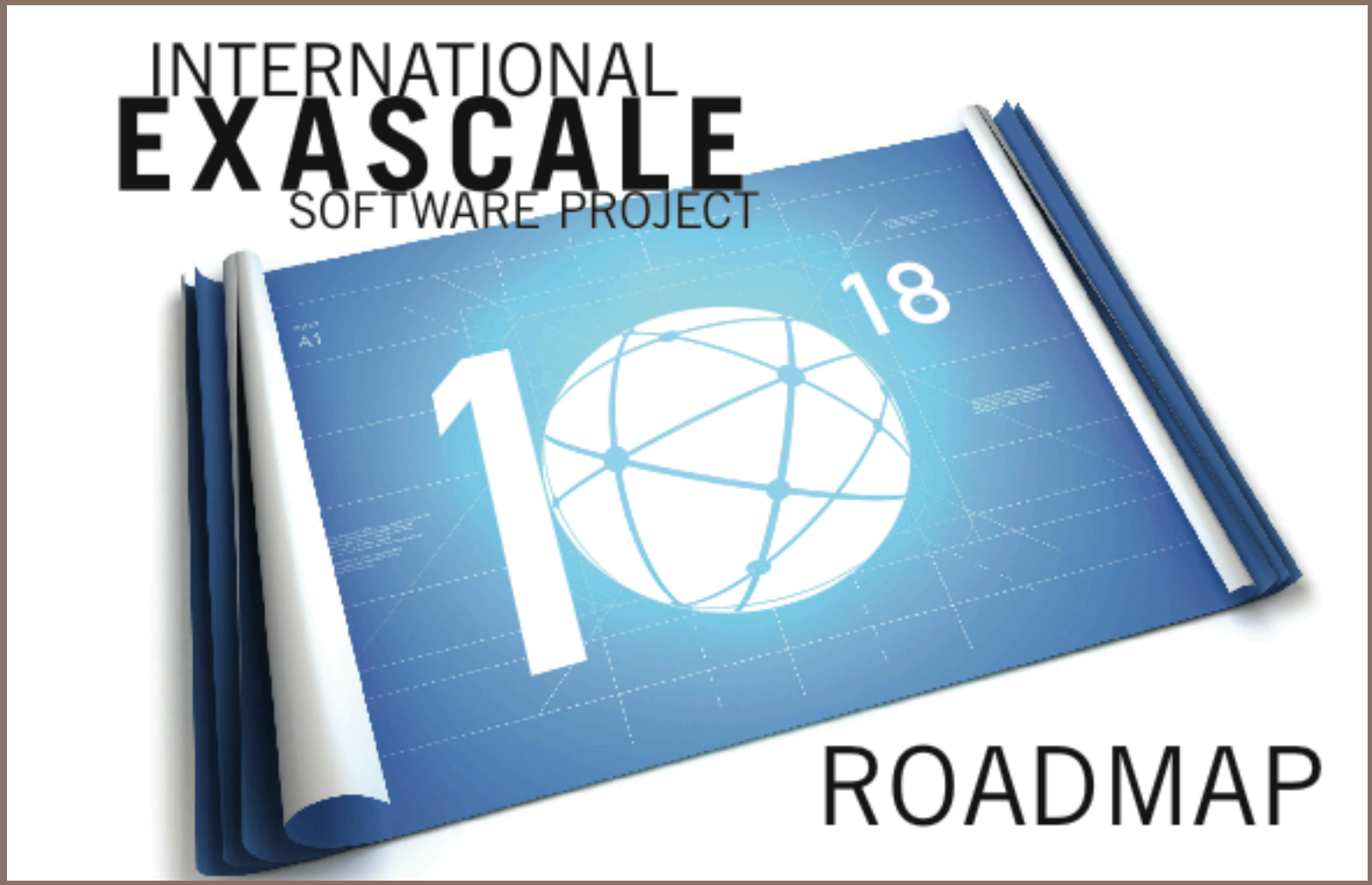
Potential impact on usability, capability, and breadth of community

- **Make systems more usable by a wider group of applications**
- **Enhance programmability**

4.2.4 Numerical Libraries

Numerical Libraries
Structured grids
Unstructured grids
FFTs
Dense LA
Sparse LA
Monte Carlo
Optimization





http://www.exascale.org

Pete Beckman & Jack Dongarra