



NNSA HPC Roadmap and Potential Tool Gaps

Scalable Tools Workshop
David Montoya
August 3, 2015

UNCLASSIFIED LA-UR-15-26094

What and Why..

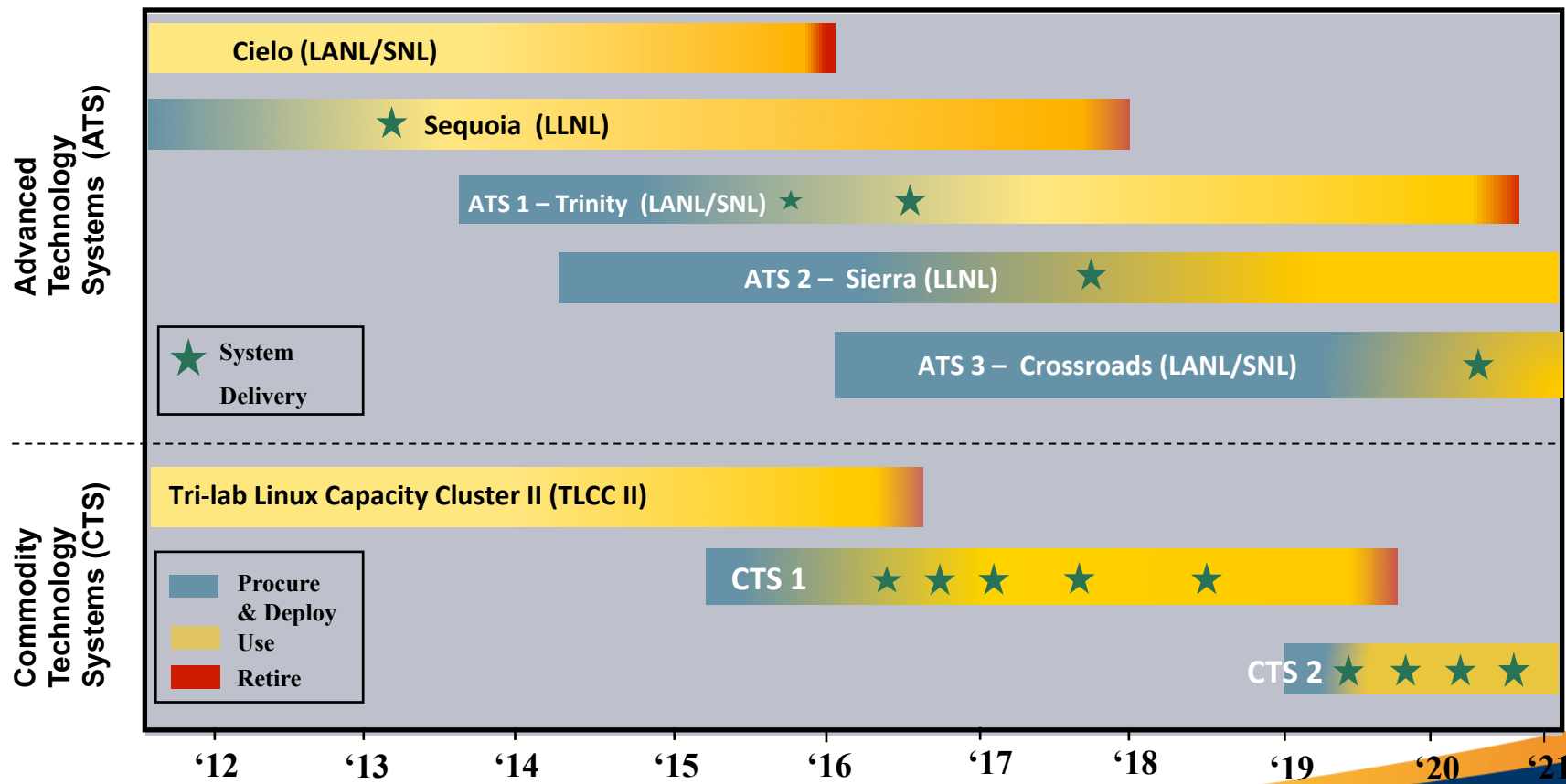
- High level picture on NNSA activities that affect tool roadmaps
- Programming Environment concerns over the next 5 years
- Feed into discussions on implementation models for tools and evolving ecosystem

What's in play

- What's the machine roadmap?
- Research and Fast/Design Forward efforts
- ATDM projects / Co-design Centers
- NNSA Five year production plan

UNCLASSIFIED LA-UR-15-26094

ASC Platform Timeline



Fiscal Year

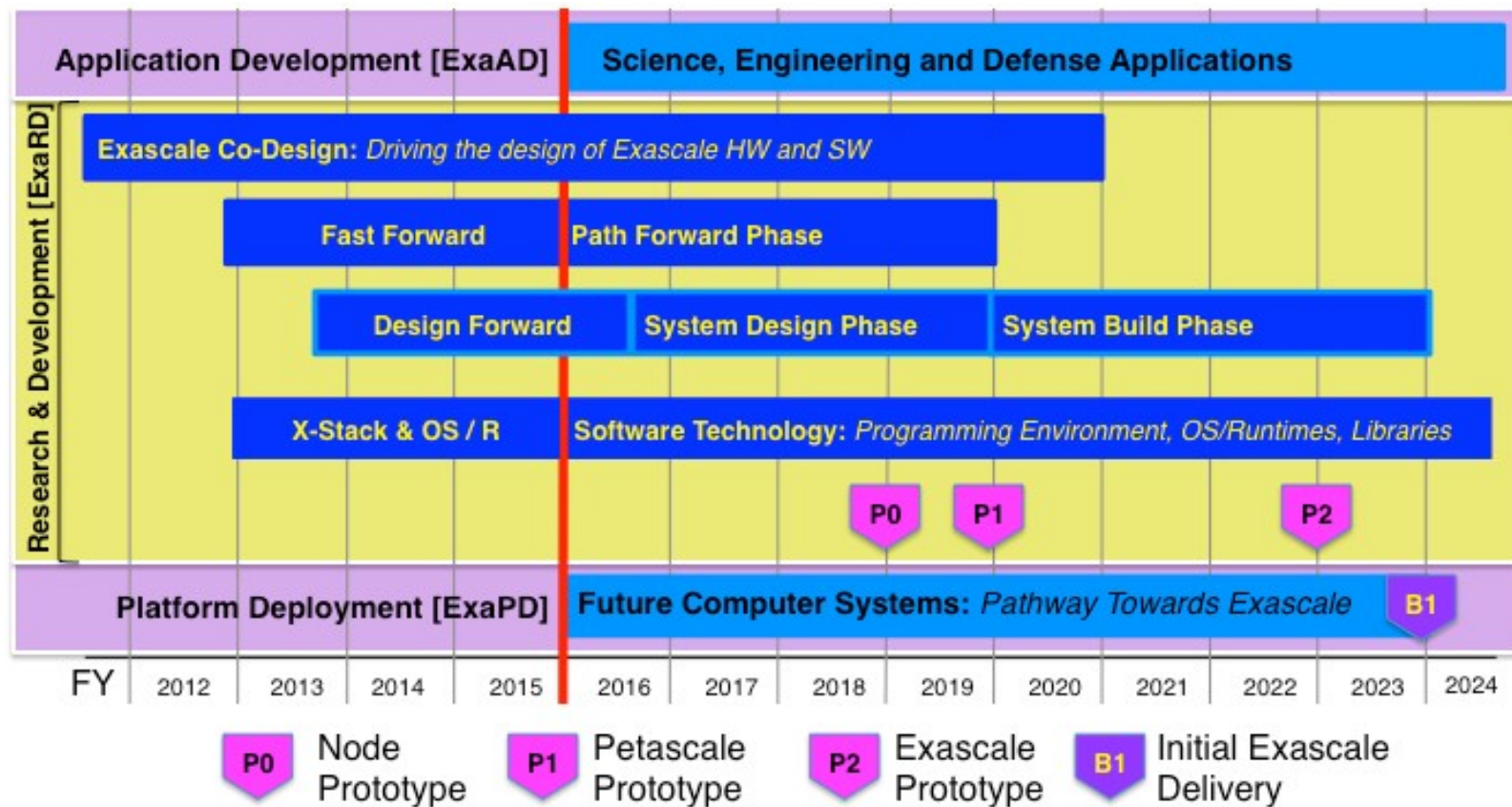
UNCLASSIFIED LA-UR-15-26094

Operated by Los Alamos National Security, LLC for the U.S. Department of Energy's NNSA

Architecture – near term

- The upcoming Trinity platform (ATS-1, 2015) which is made up of Intel Haswell and Knights Landing (Xeon Phi) processors. Burst Buffers at the IO node layer
- Contrast to the subsequent DOE Coral procurement (ATS-2 machine, 2017, LLNL – Sierra, ORNL - Summit) which is made up of IBM power9 and NVIDIA Volta GPUs. Burst Buffer at the node layer.
- Vast differences as we march through platform evolution. Trinity, with ~20,000 nodes compared to ORNLs ATS-2 Summit with 3,400 nodes.
- Memory architectures are also evolving with tighter integration, deeper hierarchies and additional capabilities.
- How do applications evolve to take advantage of the architecture advances and integrated software components?

Where are we going? The march toward Exascale



UNCLASSIFIED LA-UR-15-26094

ATDM (Advanced Technology Development and Mitigation)

Includes laboratory code and computer engineering and science projects that pursue long-term simulation and computing goals relevant to both exascale computing and the broad national security missions of the NNSA

- All three ASC labs are funded to undertake **new code development** under ATDM (aka “next gen”)
 - Funding levels at levels commensurate with staff sizes for a “typical” code team (~dozen people), plus a small amount of additional CSSE support (e.g. tools and programming models)
 - Targeting ATS platforms in 5+ years (CORAL, APEX, exascale, ...)
 - Higher risk / high reward strategy taking advantage of new technologies
 - Each ASC lab is pursuing a slightly different approach, with increased emphasis on sharing lessons-learned and solutions
 - Provides much needed “free energy” to maintain current production capabilities while addressing long-term goals
- Current production codes are likewise undergoing aggressive transformations to prepare for ATS deployments

ATDM represents the first time ASC has undertaken “from scratch” multi-physics code startups since the beginning of ASCI (mid-late 1990’s)

UNCLASSIFIED LA-UR-15-26094

ASC applications and the (inevitable?) movement toward Asynchronous Task Models

- MPI will continue to be relevant at Exascale – with evolution
- Different applications/algorithms have varying needs
 - Some algorithms should see a large benefit from dynamic scheduling and load balancing
 - Some algorithms can be statically scheduled very efficiently
 - Multi-physics will require flexibility and composability
- Many algorithms are only semi-asynchronous
 - Timestep reductions
 - Inter-package dependencies
- The “right” level of granularity is a research question
 - Coarse grained = replace MPI with tasks, and manage threading, SIMT, and SIMD within the task
 - Fine grained = attractive, but requires very fast/smart runtime
 - Programming model abstraction shouldn't dictate this

UNCLASSIFIED LA-UR-15-26094

Programming model adoption – an incremental approach to multi-physics

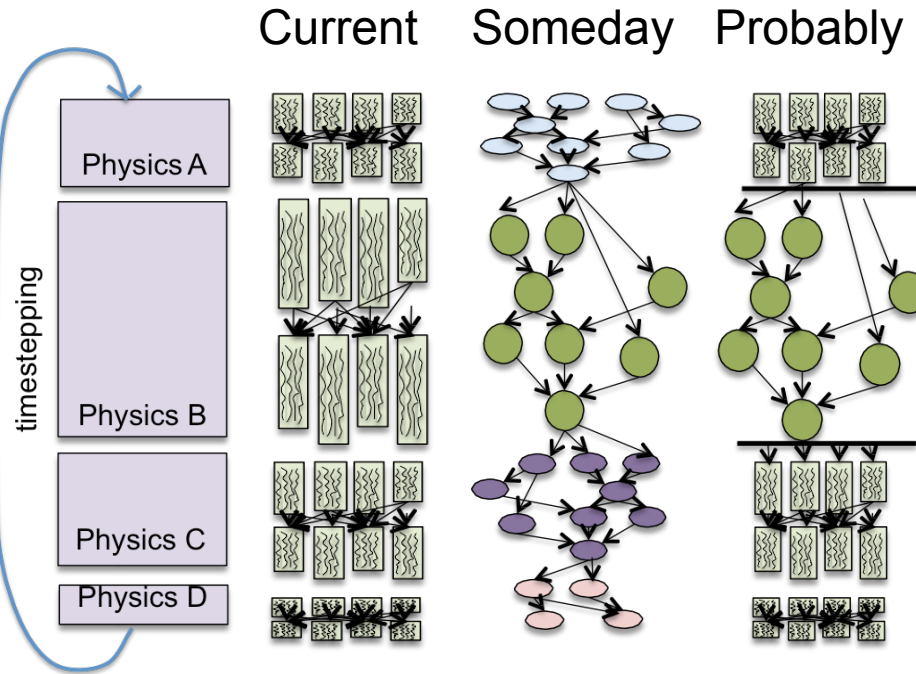
Distinct physics packages are implemented differently

- Languages
- MPI patterns/communication abstractions
- Load balancing strategy
- Use of threading

Little or no overlapping of physics packages in time or space

Issues looming today mixing MPI + X packages

- E.g. OpenMP calling lib written in pthreads
- MPI communicator size dictated by the least-threaded (or longest running) package



Introduce tasks in the packages that will benefit the most

- Dynamic, load imbalanced
- Long-running

Interoperability is key to this strategy

- Initially, we do not need tasks and MPI+X to overlap in time or space
- Just share the same executable, hand-off control between package boundaries


What breakthrough in programming environments is required for exascale?

- For OpenMP/threading – static and dynamic analysis to detect non-deterministic race conditions
- Performance
 - Focus on *actionable* outcomes, not just collection – workflows, feedback loops
- Debugging
 - At scale with MPI+X
 - Task-models
- Construction and maintenance of asynchronous task models
 - Helping domain scientists reason about execution flow – visual tools
 - Ability to unit test task behavior with full coverage of inputs/outputs
- Memory / Data movement analysis tools
 - How/where is data motion a bottleneck (esp. between levels of the memory hierarchy)
 - Access patterns / locality / movement
- Compilers
 - Optimizing through abstractions (e.g. templates, threads, lambdas)

UNCLASSIFIED LA-UR-15-26094

Five Year *Plan for ASC Pre-exascale Platform Environments*

Plan for ASC Pre-exascale Platform Environments identifies, assesses, and specifies development and deployment approaches for critical infrastructure components in five key Computational Systems and Software Environment/Facility Operations and User Support (CSSE/FOUS) technical areas:

- **Programming environments and tools** 
- Petascale/Exascale data analysis
- Input/output (I/O), I/O, file systems and archives
- Networks and interconnects
- System Management and Monitoring

UNCLASSIFIED LA-UR-15-26094

Programming Environments and Tools

Areas of Concern

Balancing Evolutionary and Revolutionary Programming Models

- **CONCERN: Need for Targeted Evolution of Existing Models**
- **CONCERN: Lack of Consensus in New Programming Models**
- **CONCERN: Transition to New Programming Models**
- **CONCERN: Missing Support for Multilevel Memory Models**

Support for Co-Existence of Evolutionary and Revolutionary Runtimes

- **CONCERN: Missing Runtime Interoperability**
- **CONCERN: Lack of Mechanisms to Control Data, Thread, and Task Placement**

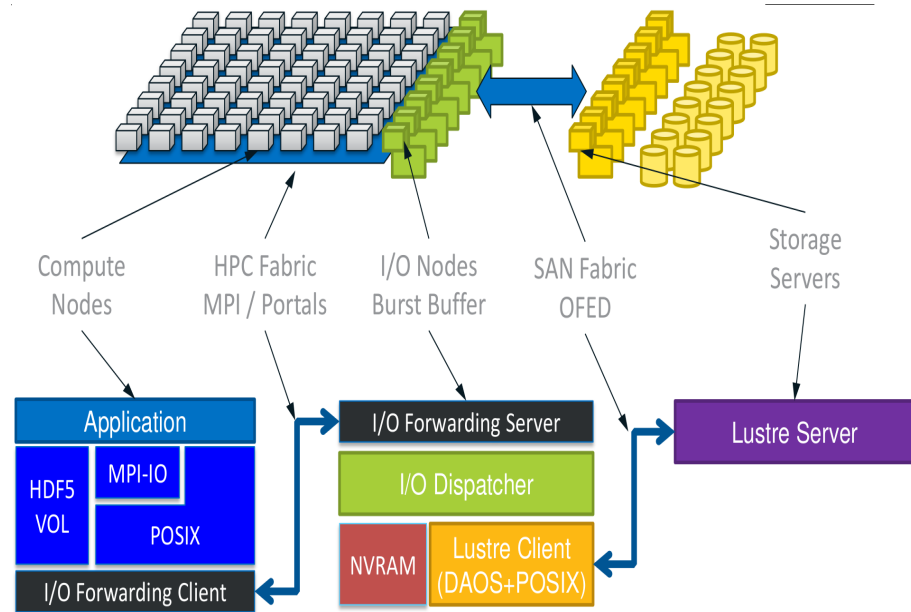
Data Storage System efforts through Fast Forward program

How can tools help here?

- Complete memory to storage pipeline
- New runtime??
- Good architectural concept of what is coming

Fast Forward Prototype

- HDF5 used to prototype application interaction with IO environment. The application stack expected to evolve.
- Burst Buffers shown here as near extensions to node memory systems
- I/O Dispatcher serves as the data runtime across the environment
- DAOS the interface to the further removed storage elements
- <https://users.soe.ucsc.edu/~ivo//blog/2013/04/07/the-ff-stack/>



Programming Environments and Tools

Areas of Concern

Introspection across the Hardware/Software Stack

- **CONCERN: Missing Interfaces in the Hardware/Software Stack**
- **CONCERN: Missing Integration of External Sensors**

Support For Efficient and Scalable Resilience

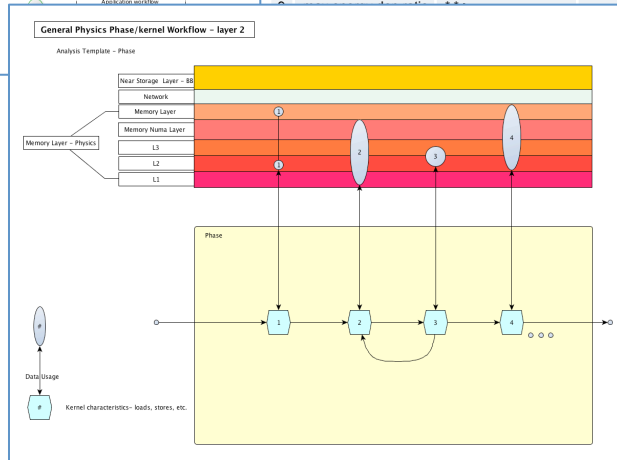
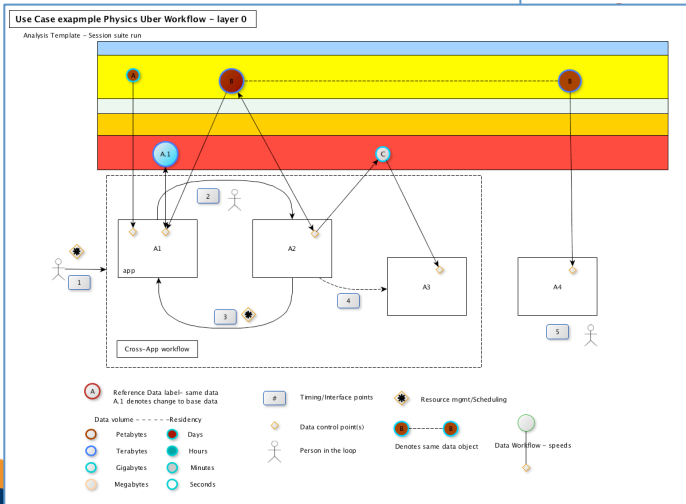
- **CONCERN: Interfaces across the HPC Ecosystem Are Not Being Focused On and Currently Provide Limited Support**
- **CONCERN: Existing Resilience Approaches Will Not Scale to New Machines**

Power-Aware and Power-Limited High Performance Computing

- **CONCERN: Lack of Tools that Help Developers Understand and Influence Power Impacts of Their Design Decisions**
- **CONCERN: No Integrated Approach for System-Wide Power Caps /Impact on System Design**
- **CONCERN: Power as a Constraint Leads to Performance Issues**

UNCLASSIFIED LA-UR-15-26094

- Workflow performance
- Interface points
- Memory/Data Hierarchy use
- Programming Model integration



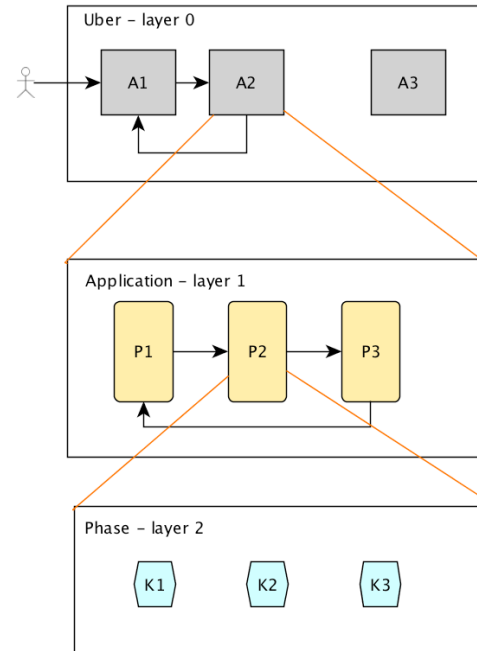
				moonlight			cielito		
				Min cycles per vector of zones			Min cycles per vector of zones		
				stores			stores		
Step result		FP operations per zone	Count	FP	s	min	FP	stores	min
1	In Ti interval index, interpolation parameter	In		30	2	30	30.5	1	30.5
2	In <gv>	- * int min max double	1 T grid	4	3	4	4	1.5	4
3	energy dep/reaction	* +	5 reactions	5	10	10	5	5	5
4	<gv>	* +	3 reactions	3	6	6	3	3	3
5	max stable time step	exp	5 reactions	145	10	145	105	5	105
6	reaction rates, sum	++ * + * + * + >		5		5	5		5
7	energy dep rate, sum	11* 5+	5 reactions	11	10	11	11	5	11
8		5* 5+	5 reactions	5	2	5	5	1	5
				2		2	2		2
				8	10	10	8	5	8
				6		6	6		6
				12	12	12	12	6	12
				4	2	4	4	1	4
				64.8		64.8	92.4		92.4
				304.8	67	314.8	292.9	33.5	292.9
				3.41E+07		3.30E+07	1.64E+07		1.64E+07

Workflow Layers

Layer 0 – UBER layer. Application to application that constitute a suite series, which may include closely coupled applications and decoupled ones that provide an end-to-end target case. This layer is at the project layer and where there is user and system interaction, constructed to find an answer to a specific science question. Layer 0 is from the perspective of an end user.

Layer 1 – Application layer. Within an application that may include one or more phases with differing computational and data requirements. Interacts across memory hierarchy to archival targets. The subcomponents {P1..Pn} are meant to model various aspects of the physics; Layer1 is that part of the workflow that incorporates the viewpoint of the scientist.

Layer 2 – Phase layer. This describes the processing of kernels within a phase and associated interaction with various levels of memory that include cache levels. This layer is the domain of the computer scientist and is where the software and hardware first interact.



Programming Environments and Tools

Areas of Concern

Performance Tuning and Optimization Tools

- **CONCERN: Crosscutting Issue—the Need for Scalable Infrastructures**
- **CONCERN: Limited Tool Support for Investigating Memory Usage and Efficiency**
- **CONCERN: *Tools Lacking* for Asynchronous Task/Data Programming Models**
- **CONCERN: *Tools Lacking* Support for New Hardware Features, including Accelerators**

Debugging and Correctness Tools

- **CONCERN: Currently Limited Support for Next-Generation System Designs**
- **CONCERN: Pre-Emptive Correctness Checking Too Limited and Does Not Support Emerging Runtimes/Programming Models**

Initial Challenges going forward -

- Multi-level memory models (the data pipe)
 - Coherent memory spaces ease transition, but consensus that application “hints” or explicit direction will be necessary.
- OpenMP scalability and portability
 - Even the best implementations have too much overhead to allow fine-grained parallelism
 - Will OpenMP 4.x be sufficient for targeting GPUs?
- Asynchronous Task Models
 - Research needed in how applications can easily express tasking
 - Growing consensus in its viability, but lots of unanswered questions
 - Will domain-scientists revolt? Yes, if we don’t have good...
- Tools
 - Need for cross-platform, better integrated, easy-to-use, scalable tools that focus on insight

Backup slides

Back up slides – strategy details on 5 year plan

Balancing Evolutionary and Revolutionary Programming Models

CONCERN: Need for Targeted Evolution of Existing Models

- **Strategy** - Maintain active involvement in standardization bodies.
- **Strategy** - Research new abstractions for the existing models.

CONCERN: Lack of Consensus in New Programming Models

- **Strategy** - Establish closer ties with programming model research efforts and increase early evaluation efforts.
- **Strategy** - Support development of best practices and ultimately standardization.

CONCERN: Transition to New Programming Models

- **Strategy** - Increase design studies, in close collaboration between system, library, and application developers.

CONCERN: Missing Support for Multilevel Memory Models

- **Strategy** - Develop and test new application programming interfaces for memory management.

Support for Co-Existence of Evolutionary and Revolutionary Runtimes

CONCERN: Missing Runtime Interoperability

- **Strategy** - Increase design studies, in close collaboration between system, library, and application developers.

CONCERN: Lack of Mechanisms to Control Data, Thread, and Task Placement

- **Strategy** - Develop new intra-node data locality and movement abstractions.
- **Strategy** - Develop a new technique for inter-node data staging that is also applicable to composite workflows.

Introspection across the Hardware/Software Stack

CONCERN: Missing Interfaces in the Hardware/Software Stack

- **Strategy** - Survey existing interfaces.
- **Strategy** - Ensure tighter integration of introspection into the software stack.
- **Strategy** - Standardize introspection interfaces.

CONCERN: Missing Integration of External Sensors

- **Strategy** - Provide introspection into job-level data through the resource managers.
- **Strategy** - Integrate and provide access to machine-room level sensors.

Support For Efficient and Scalable Resilience

CONCERN: Interfaces across the HPC Ecosystem Are Not Being Focused On and Currently Provide Limited Support

- **Strategy** - Play an active role in the definition of new memory interfaces.
- **Strategy** - Play an active role in the inclusion of fault-tolerance (FT) techniques into runtimes and programming standards.

CONCERN: Existing Resilience Approaches Will Not Scale to New Machines

- **Strategy** - Expand efforts in efficient checkpointing and integrate into overall software stack.

Performance Tuning and Optimization Tools

CONCERN: Crosscutting Issue—the Need for Scalable Infrastructures

- **Strategy** - Develop tool-, system-, and application-agnostic tool infrastructure.
- **Strategy** - Integrate tool infrastructures across components and make them available as a shared system service.

CONCERN: Limited Tool Support for Investigating Memory Usage and Efficiency

- **Strategy** - Develop memory usage tools.
- **Strategy** - Develop tools to better understand node local and job wide data transfers.

CONCERN: Tools Lacking for Asynchronous Task/Data Programming Models

- **Strategy** - Develop tools and new metrics applying to task-based systems.
- **Strategy** - Develop tools to support porting to new programming models.

CONCERN: Tools Lacking Support for New Hardware Features, including Accelerators

- **Strategy** - Enhance current existing hardware and vendor-agnostic tools (possibly by integrating vendor-specific approaches).
- **Strategy** - Develop tools to model and estimate benefits from porting to new architectures.

Debugging and Correctness Tools

CONCERN: Currently Limited Support for Next-Generation System Designs

- **Strategy** - Extend existing debugging support for next-generation architectures and runtime systems.

CONCERN: Pre-Emptive Correctness Checking Too Limited and Does Not Support Emerging Runtimes/Programming Models

- **Strategy** - Enhance current existing hardware and vendor-agnostic tools (possibly by integrating vendor-specific approaches).

Power-Aware and Power-Limited High Performance Computing

CONCERN: Lack of Tools that Help Developers Understand and Influence Power Impacts of Their Design Decisions

- **Strategy** - Increase power instrumentation and access to power (and thermal) data in future machines.
- **Strategy** - Provide new generation of tools that enable the correlation of power, power caps, and performance.
- **Strategy** - Develop high-level abstractions to hide the impact of power-aware programming.

CONCERN: No Integrated Approach for System-Wide Power Caps / Impact on System Design

- **Strategy** - Develop power management system and policy.

CONCERN: Power as a Constraint Leads to Performance Issues

- **Strategy** - Develop performance models for applications under power caps.
- **Strategy** - Expand efforts in the development of power-aware runtime scheduling and integration into other runtime efforts.