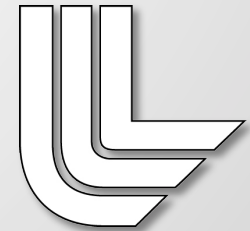


Exascale Co-Design Center for Materials in Extreme Environments (ExMatEx)*



James Belak (belak@llnl.gov)
www.exmatex.org

Center for Hierarchical Materials Design
Northwestern University

27 May, 02014

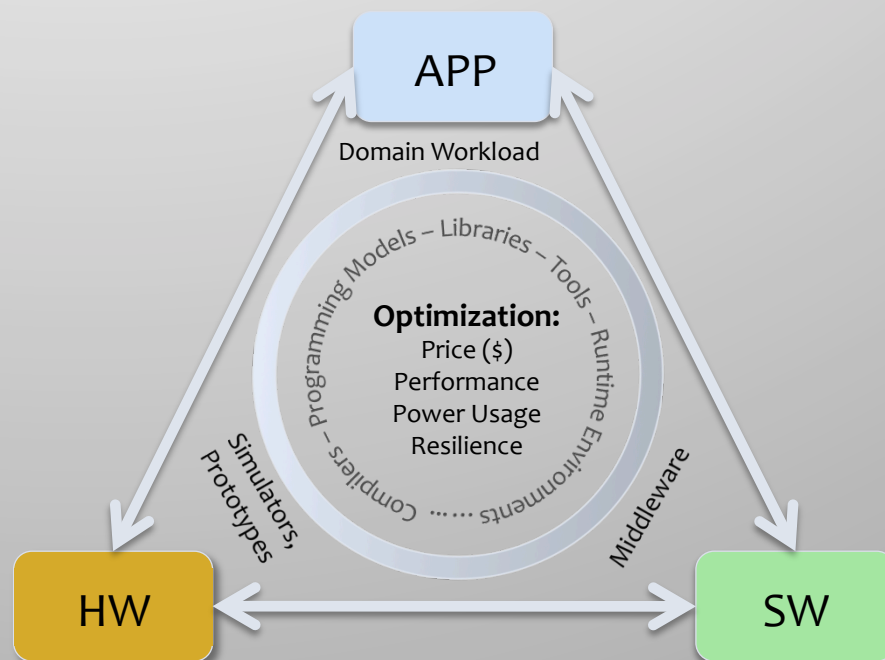
* co-PI: Tim Germann (tcg@lanl.gov)

‘Algorithm research has been driven by hard to use machines’
-Rob Schreiber (HP Labs)

‘People who are serious about software should make their own hardware.’
- Alan Kay (Xerox PARC)

LLNL-PRES-655190

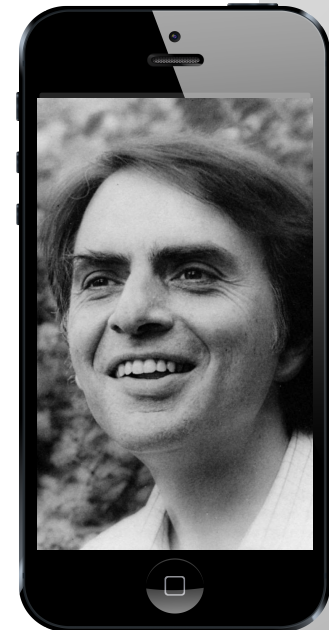
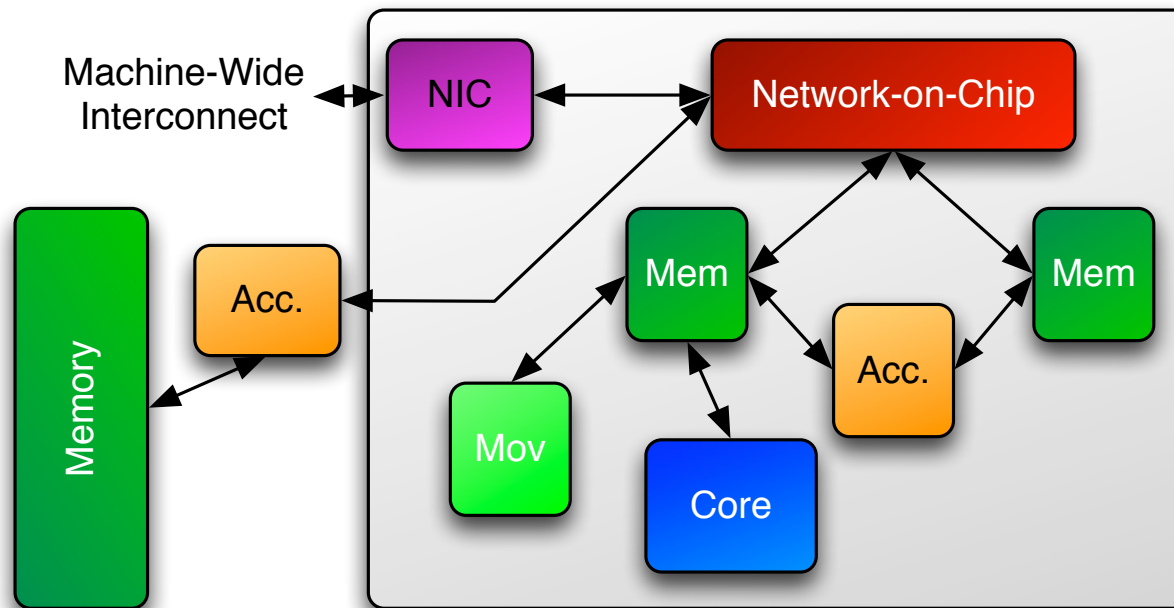
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 and supported by Office of Science Advanced Scientific Computing Research Program.



Common asked questions in exascale: What is it?

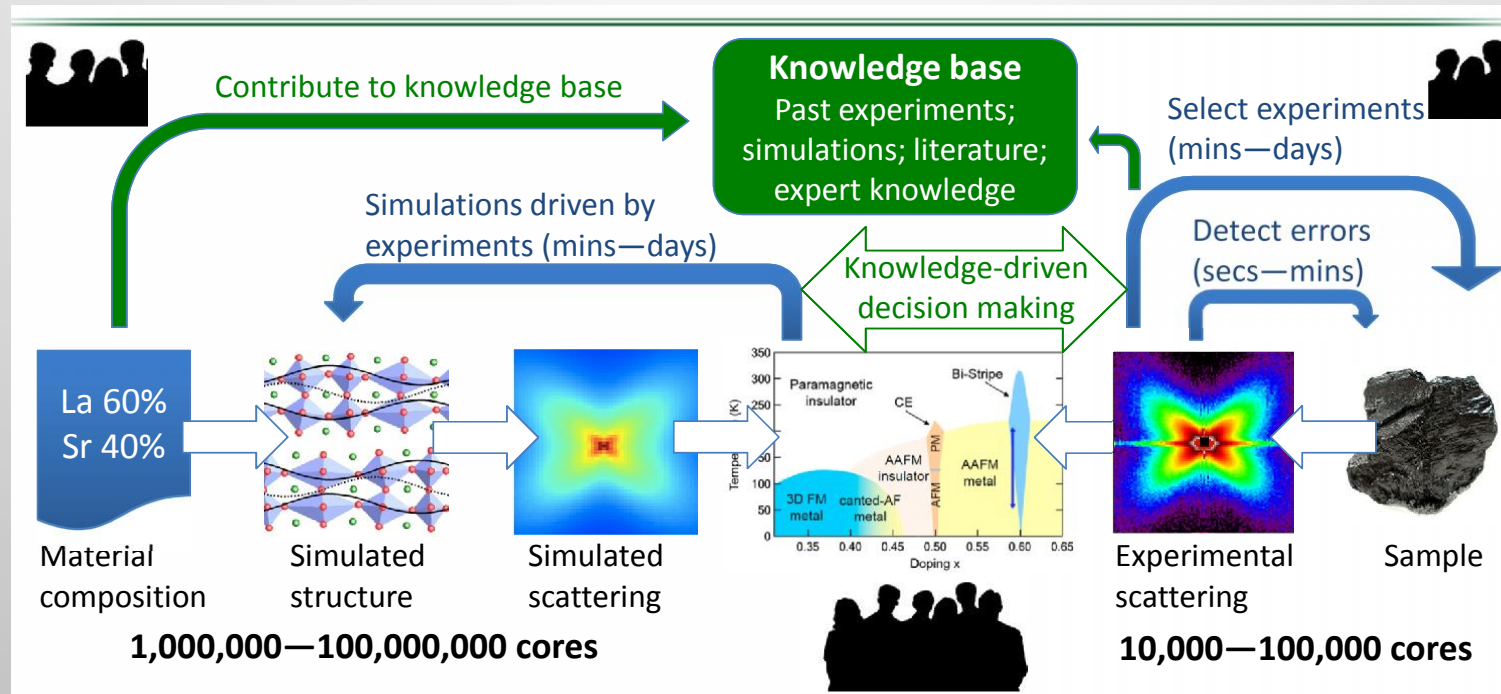
Billions of tasks all performing some calculation every nanosecond.

Conceptual Design: The Revolution is about the Node



Design by Dave Resnick,
Scalable Computer Architectures, Sandia National Labs, NM

Common asked questions in exascale: Why do I care? Accelerating the Scientific and Design Processes



Many similar opportunities across a broad spectrum of DOE science:

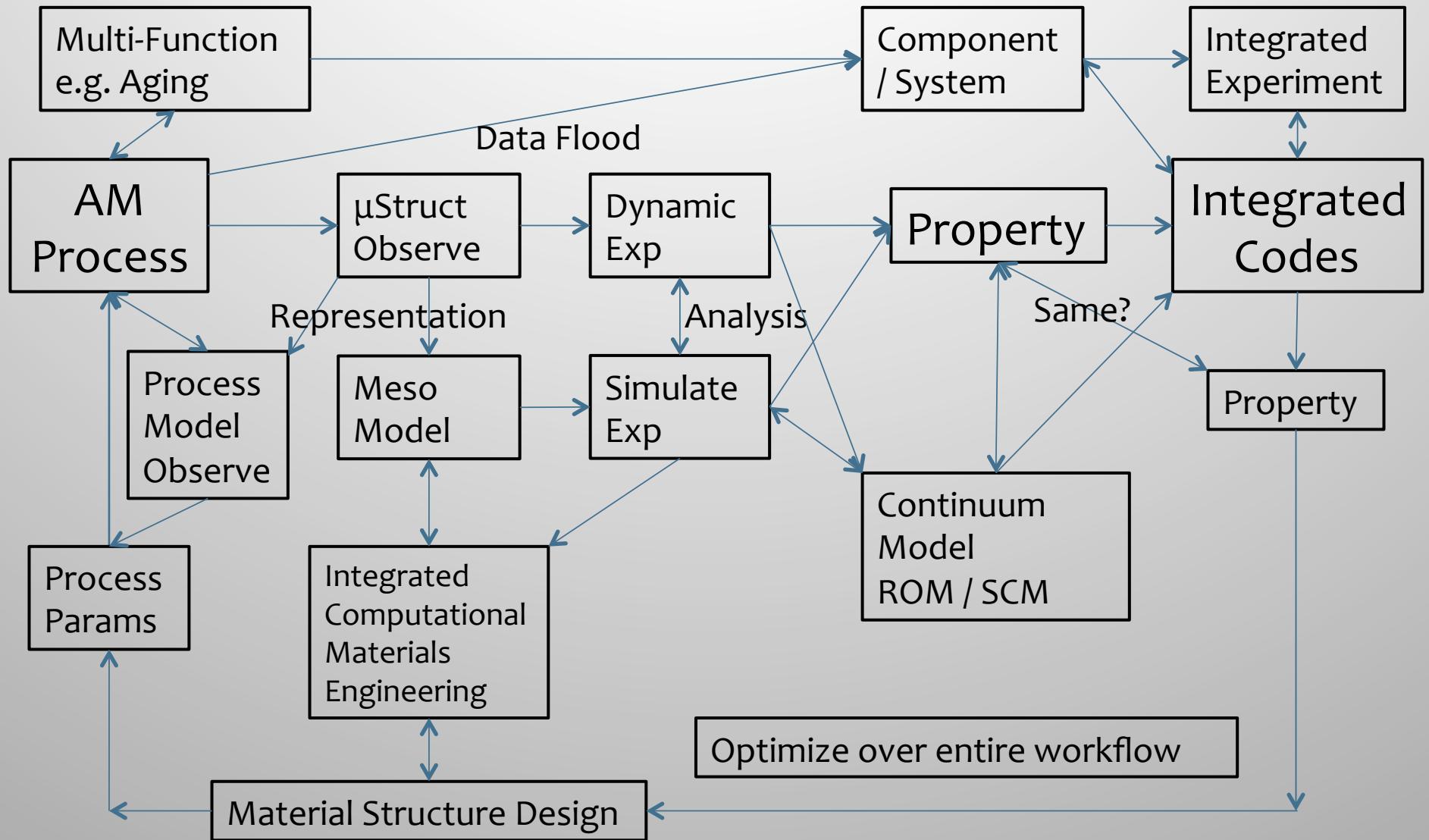


Next-generation instruments will increase data rates by x1000 or more

Diffuse scattering images from Ray Osborn et al. (MSD, APS, MCS)

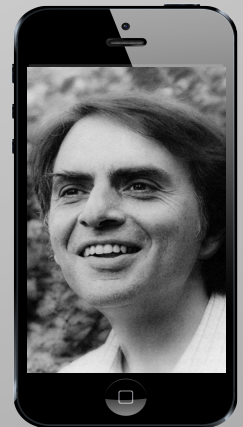
Ian Foster, ANL

Workflow for Additive Manufacturing (POCs: Melissa Marggrath, Wayne King, Chris Spadaccini, Morris Wang)



Commonly asked questions in exascale: What is the problem?

- **Power, energy, and heat dissipation** are the central issues
- Imagine a computer with **billions and billions** of cell phone processors (14MW) or **millions and millions** of throughput optimized cores, GPGPUs (20MW)
 - How do you program it to work on one science problem?
 - The architecture will be **heterogeneous** and **hierarchical**, with very high **flop/byte ratios**.
 - Single program multiple data bulk synchronous parallelism will no longer be viable.
- **Data Movement** will be expensive and computation will be cheap
 - Need to present the physics so the computation occurs where the data is!
 - Traditional global checkpoint/restart will be impractical: need local / micro checkpoint (flash memory?)
- Simulation codes will need to become **fault tolerant and resilient**
 - Recover from soft and hard errors, and anticipating faults
 - Ability to drop or replace nodes and keep on running
 - The curse of silent errors

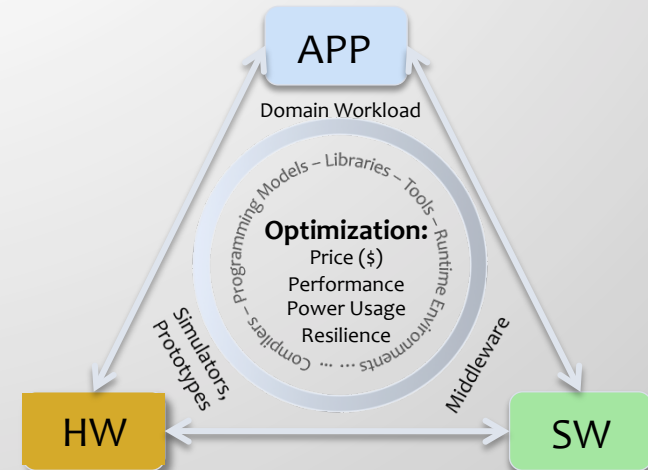


Commonly asked questions in exascale: Don't we already know how from petascale? (not really)

- **Problem: Fault tolerance is a problem at 10^5 and will be a much bigger problem at 10^9**
 - Solution: Application assisted error recovery, parity error triggers exception handler
 - Application knows what memory is “important” can catch exception and repair data
 - Exascale runtime will need to support task migration across nodes
- **Problem: Scaling (crucial for exascale) requires very very good load balancing**
 - Solution: Decomposition based on Computational Work
 - Particle-based domain decomposition - processors own particles, not regions - allows decomposition to persist through atom movement
 - Maintain minimum communication list for given decomposition - allows extended range of “interaction”
 - Arbitrary domain shape - allows minimal surface to volume ratio for communication
 - Exascale: decomposition has to become dynamic and adaptive
- **Problem: HW specific algorithms are crucial for performance but limit portability**
 - E.g. Linked cells map better to current petascale systems than neighbor lists
 - Ordering neighbors within a cell exposes SIMD parallelism
- **Problem: I/O does not work with too many files or one large file**
 - Solution: Divide and concur, what is the optimal number of files?
 - Exascale: Dedicated checkpoint filesystem (flash?)

Productive Exascale Simulation requires the coordinated efforts of Domain Scientists, Computer Scientists and Hardware Developers

- Many, many-task coordination issues
 - Greater than one hundred million, more is different
 - Synchronization (essential for time evolution)
 - Stalls (keeping everyone working)
- Better exposure into hardware details for the exascale application developer
 - Compiler Interface
 - Simulators+Emulators+Tools measure code/ecosystem metrics
 - Are we defining the right metrics?
- Application developers need a better way to express (code) the computational work of the application into the exascale computational ecosystem
 - Better programming models (e.g. domain specific languages)
 - Runtime support for heterogeneous multi-program, multi-data (MPMD) applications
- The petascale science apps are NOT general apps. They have been painfully optimized for the petascale architecture by the app developer. How do we get exascale lessons learned into quotidian science applications (VASP, LAMMPS, ...)?
- The petascale codes already account for data movement, it is only going to get worse
 - Bandwidth to memory is scaling slower than compute
 - Memory access is dominating power
- The exascale codes will need to learn to adaptively respond to the system
 - Fault tolerance, process difference, power management, ...



Note Bene: *Productive Exascale Computing will mean Ubiquitous Petascale Computing*

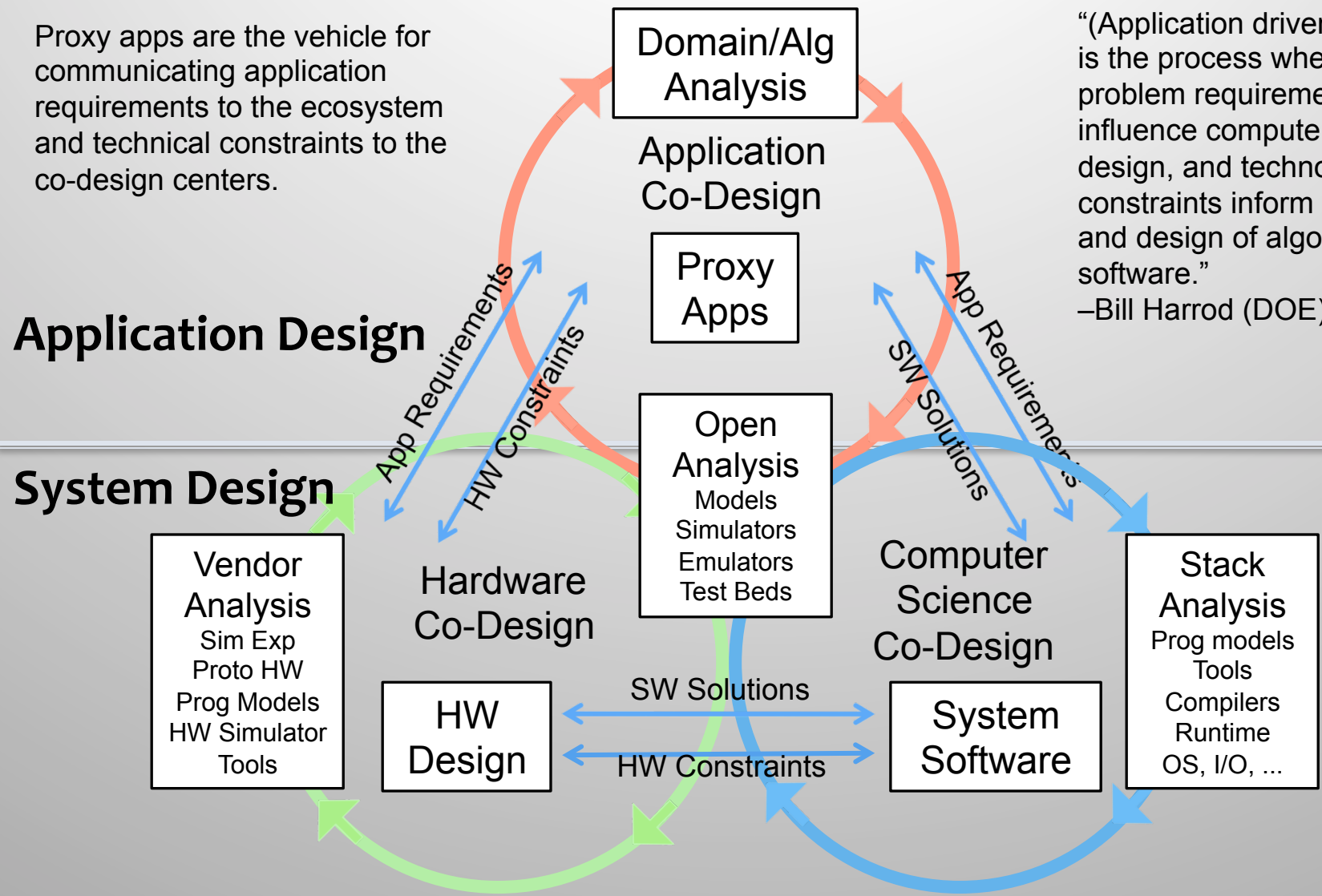
Use Case: LAMMPS (CRADA: Cray, Bristol-Myers, Dupont, LLNL, Sandia)

- 20 Years Ago
 - Hardware: Linux Beowolf Cluster
 - Software Programming Model: MPI/SPMD
 - Application Code: LAMMPS
- 5 Years in the Future
 - Hardware: Exascale (Petascale Cluster)
 - Software Programming Model: MPI+X (threads) or task-based asynchronous???
 - Application Codes???

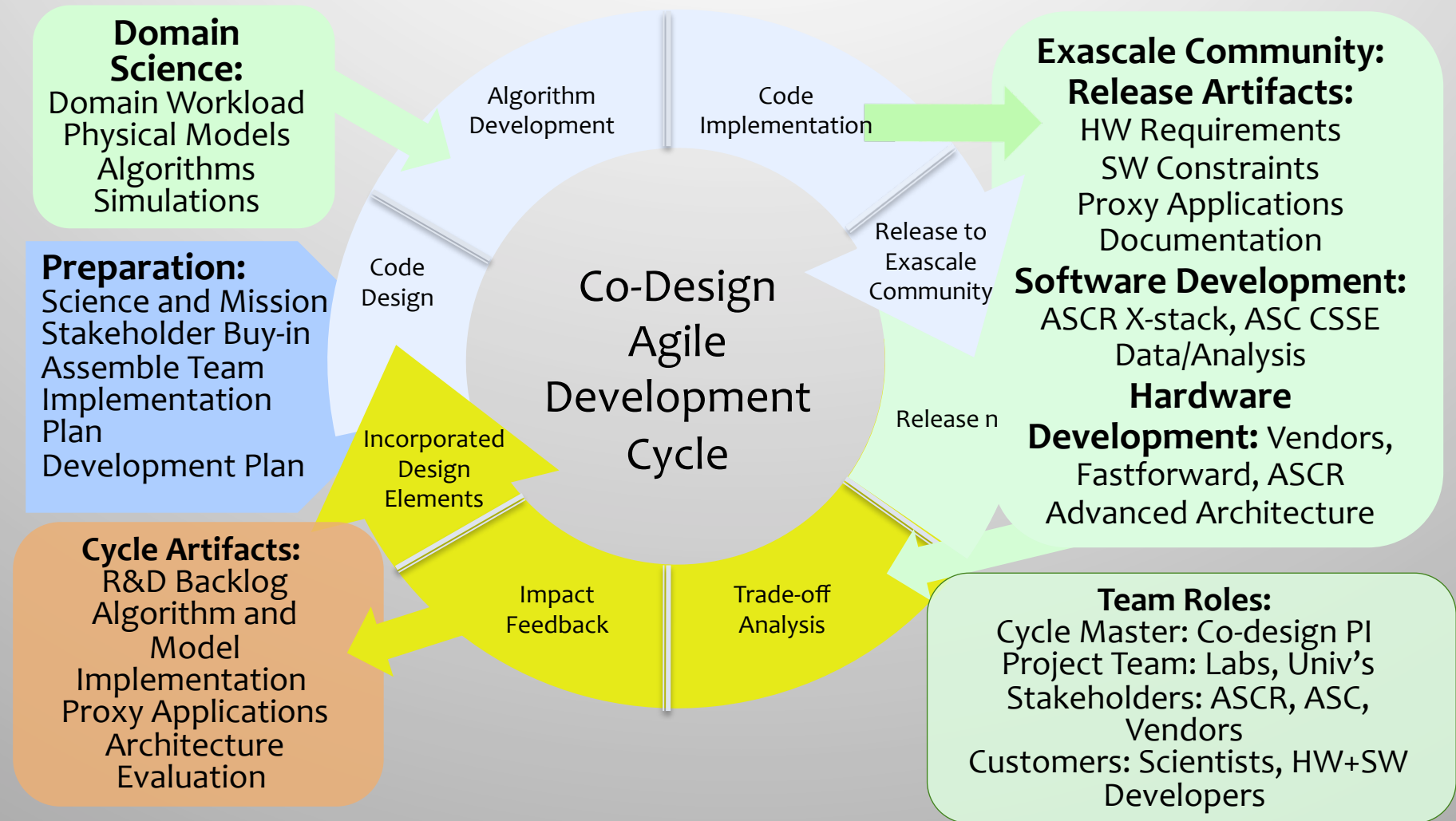
The Workflow of Co-design between Application Co-design Centers, Vendors, and the broader Research Community

Proxy apps are the vehicle for communicating application requirements to the ecosystem and technical constraints to the co-design centers.

“(Application driven) co-design is the process where scientific problem requirements influence computer architecture design, and technology constraints inform formulation and design of algorithms and software.”
 –Bill Harrod (DOE)



Each co-design project is using *proxy apps* to capture the requirements of their application and reformulating these *proxy apps* from the lessons learned from co-design trade-off analysis.



ExMatEx Communicates with the Exascale Ecosystem through Proxy-Apps

- Materials Science applications present their requirements through proxies
 - Real applications have multiple proxies
 - Proxies have docs, specs, and a reference implementation
- Ecosystem members evaluate proxies and respond with capabilities
 - This informs trade-off analysis
- Co-Design is more than just applications and hardware architecture
 - There is a whole ecosystem to influence



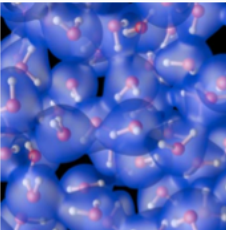
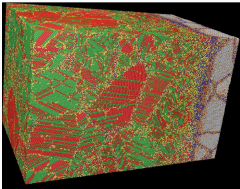
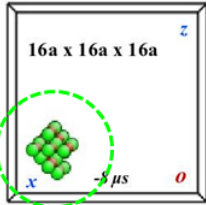
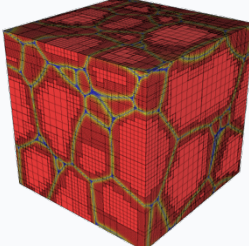
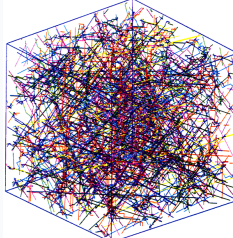
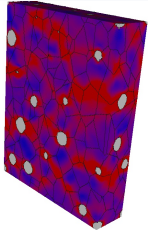
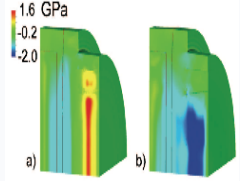
Proxy Applications: represent the application workload and requirements to the ESCE ecosystem (HW,RT,...)

A small application code that proxifies (stands for) some aspect of the computational workflow of a full application is a proxy app.

- **Kernels:** standalone pieces of code that are small and performance- and tradeoff-impacting, even though decoupled from other application components.
- **Skeleton apps:** apps that reproduce the memory or communication patterns of a physics application or package, and make little or no attempt to investigate numerical performance.
- **Mini apps:** apps that combine some or all of the dominant numerical kernels contained in an actual stand-alone application and produce simplifications of physical phenomena (the app formerly known as compact).

Proxy apps are used by the ESCE ecosystem to understand the effects of hardware and software trade-offs, and also by co-design code team members to explore new technologies, languages, algorithms and programming models. Proxy apps are not static, but evolve significantly during the co-design process. Domain application code-developers and hardware/software developers will spend significant time together executing the co-design process. (Hack-a-thons!!!)

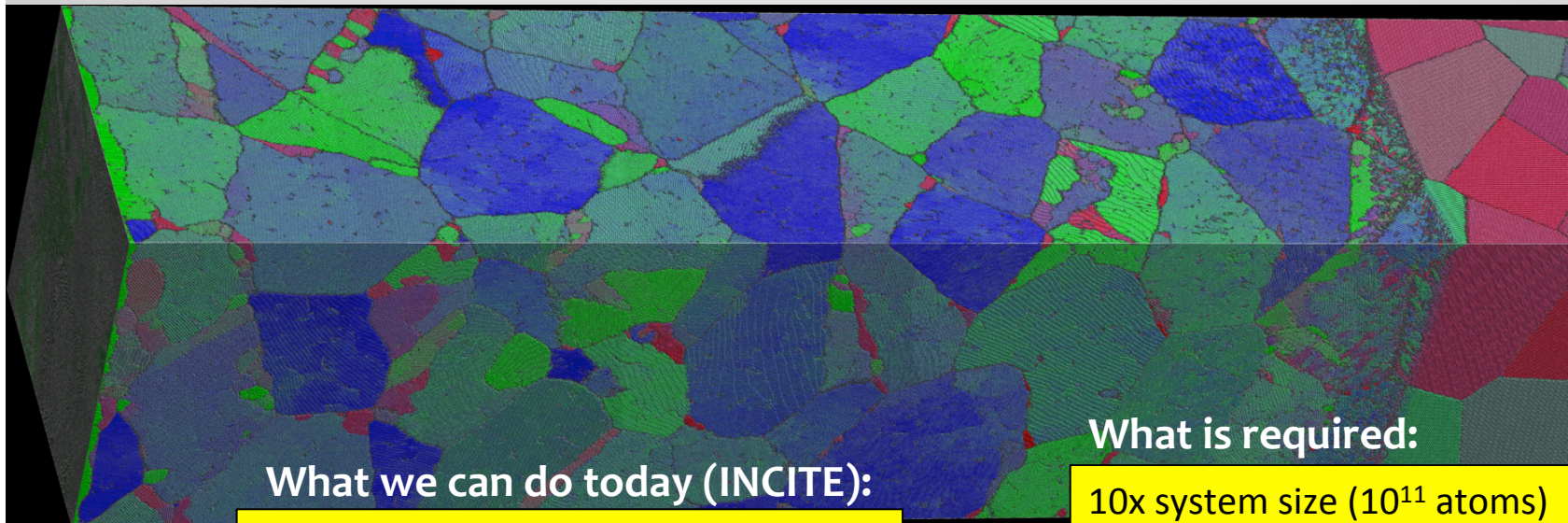
The Seven Pillars of Computational Materials Science

Ab-initio	Atoms	Long-time	Microstructure	Dislocation	Crystal	Continuum
Inter-atomic forces, EOS, excited states	Defects and interfaces, nucleation	Defects and defect structures	Meso-scale multi-phase, multi-grain evolution	Meso-scale strength	Meso-scale material response	Macro-scale material response
						
Code: Qbox/ LATTE Motif: Particles and wavefunctions, plane wave DFT, ScaLAPACK, BLACS, and custom parallel 3D FFTs Prog. Model: MPI + CUBLAS/CUDA	Code:SPaSM/ ddcMD/CoMD Motif: Particles, explicit time integration, neighbor and linked lists, dynamic load balancing, parity error recovery, and <i>in situ</i> visualization Prog. Model: MPI + Threads	Code: SEAKMC Motif: Particles and defects, explicit time integration, neighbor and linked lists, and <i>in situ</i> visualization Prog. Model: MPI + Threads	Code: AMPE/GL Motif: Regular and adaptive grids, implicit time integration, real-space and spectral methods, complex order parameter Prog. Model: MPI	Code: ParaDiS Motif: “segments” Regular mesh, implicit time integration, fast multipole method Prog. Model: MPI	Code: VP-FFT Motif: Regular grids, tensor arithmetic, meshless image processing, implicit time integration, 3D FFTs. Prog. Model: MPI + Threads	Code: ALE3D/ LULESH Motif: Regular and irregular grids, explicit and implicit time integration. Prog. Model: MPI + Threads

How do we get exascale lessons learned into quotidian science applications (VASP, LAMMPS, ...)?

Use case: competing dislocation, twinning, and/or phase transitions under shock loading

- Direct non-equilibrium molecular dynamics simulation matching time and length scales of planned LCLS experiments
 - ~1-2 μm thick nanocrystalline samples (Cu, Ti, Fe, Ta), ~400 nm grain size
 - Laser drive: 10-20 ps rise time, 150 ps duration
 - 50 fs duration X-ray “snapshot” interrogation pulses at 10 ps intervals



NEMD simulation of shocked nc-Ta on Cielito (R. Ravelo, LANL/UTEP)

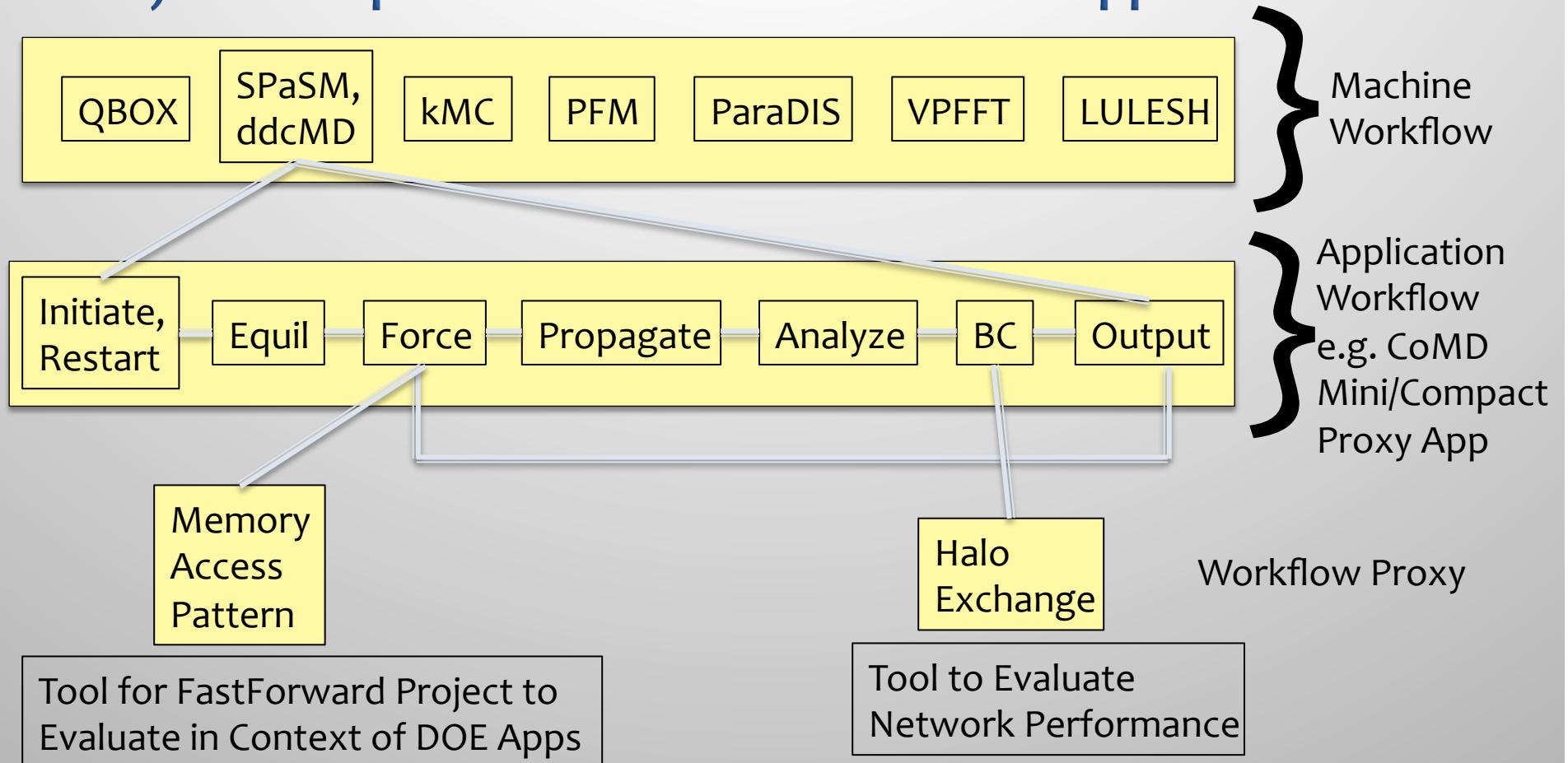
What we can do today (INCITE):

EAM potential, 200 nm grain size
 10^{10} atoms ($0.5 \mu\text{m} \times 0.5 \mu\text{m} \times 1.5 \mu\text{m}$)
Simulation time: 4 nsec (10^6 steps)
Wall clock: 2 days on Mira ($\frac{1}{2}$ Sequoia)

What is required:

10x system size (10^{11} atoms)
 $1 \mu\text{m} \times 1 \mu\text{m} \times 2 \mu\text{m}$, 400 nm grain size
More accurate MGPT potential: 100x
3 weeks on exascale system

Proxy App: a small application code that proxifies (stands for) some aspect of the workflow of a full application



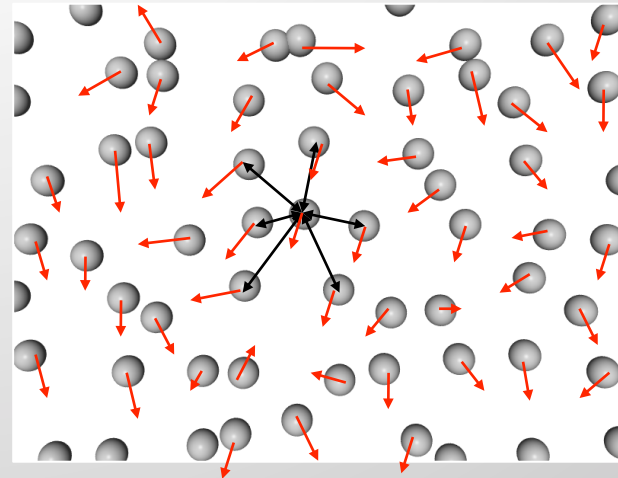
Proxy Apps are fundamentally different from Benchmark Apps. They enable the lessons learned from Co-design to be incorporated back into the full app.

Molecular Dynamics (MD) –CoMD

Molecular dynamics: particles interact via explicit interatomic potentials and evolve in time according to Newton's equations of motion:

$$\dot{\mathbf{r}}_i = \mathbf{p}_i / m_i \quad \dot{\mathbf{p}}_i = \mathbf{f}_i$$

$$\mathbf{f}_i = m_i \ddot{\mathbf{r}}_i = - \sum_j \nabla V_{ij}$$

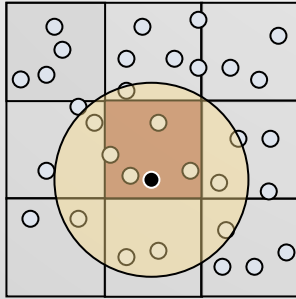


Interaction potentials determine both the physics and computer science

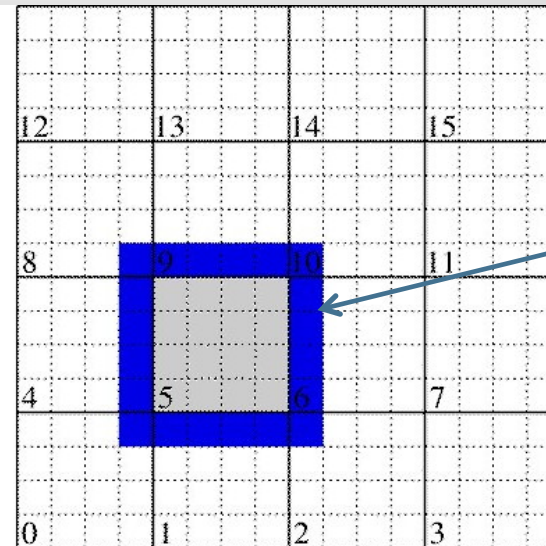
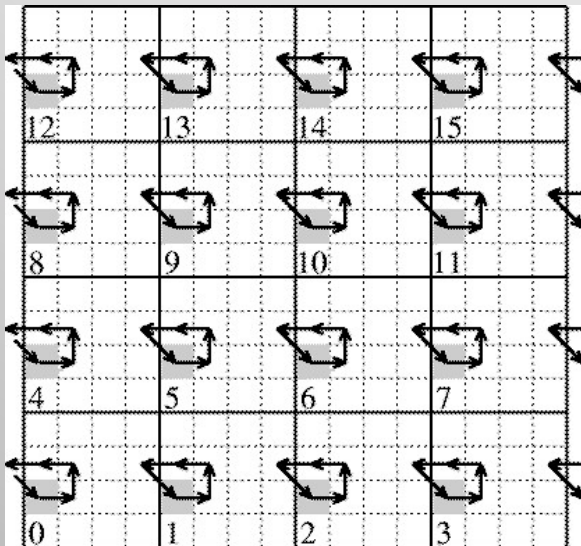
- Complex potentials are more accurate, but can require many more floating point operations.
- Locality of potential informs parallelization strategy, e.g. short-ranged potentials require only point to point communication.

Challenge Problem: Can you use an exascale computer with billion-way parallel parallelism to simulate longer in time? (not just more atoms)

How are forces calculated in a parallel MD code?



- ~ 20 atoms in each box
- ⇒ each atom interacts with 540 other atoms
- ⇒ However, only ~70 atoms lie within cutoff
- ⇒ Lots of wasted work
- ⇒ We need a means of rejecting atoms efficiently even within this reduced set



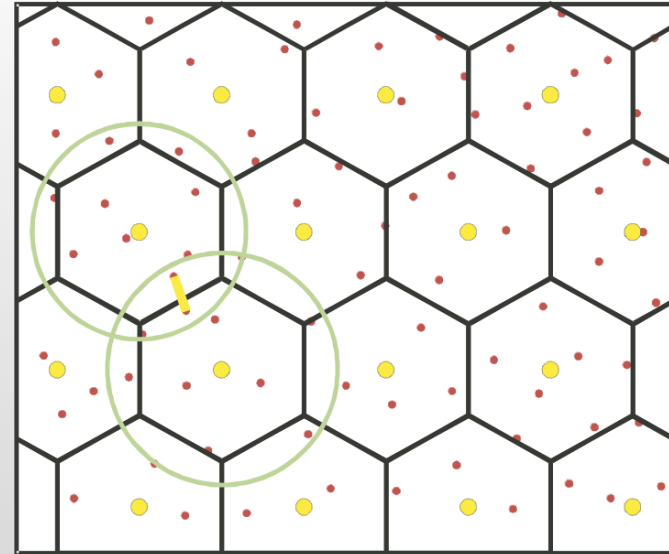
Halo Region

Fixed geometric domain decomposition limits scalability for any heterogeneous problem. Furthermore, statistical fluctuations in the force calculation between processors leads to an effective scalar term that also limits scaling (Amdahl's law).

Domain decomposition strategy for ddcMD (Dave Richards and Jim Glosli)

Design requirements:

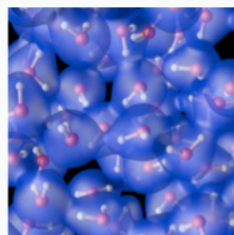
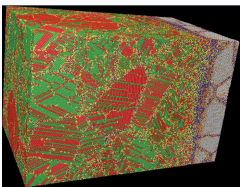
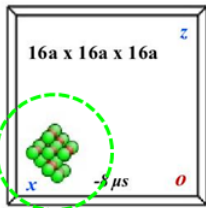
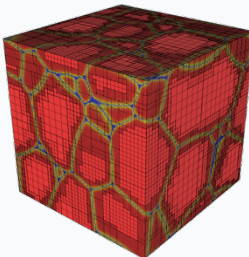
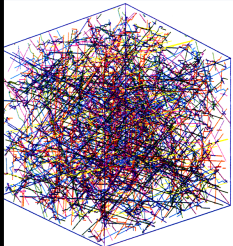
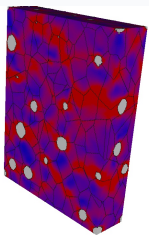
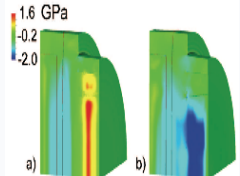
- Run efficiently on arbitrary number of processors
- Excellent weak scaling to extend size of simulation
- Excellent strong scaling to extend MD time scale



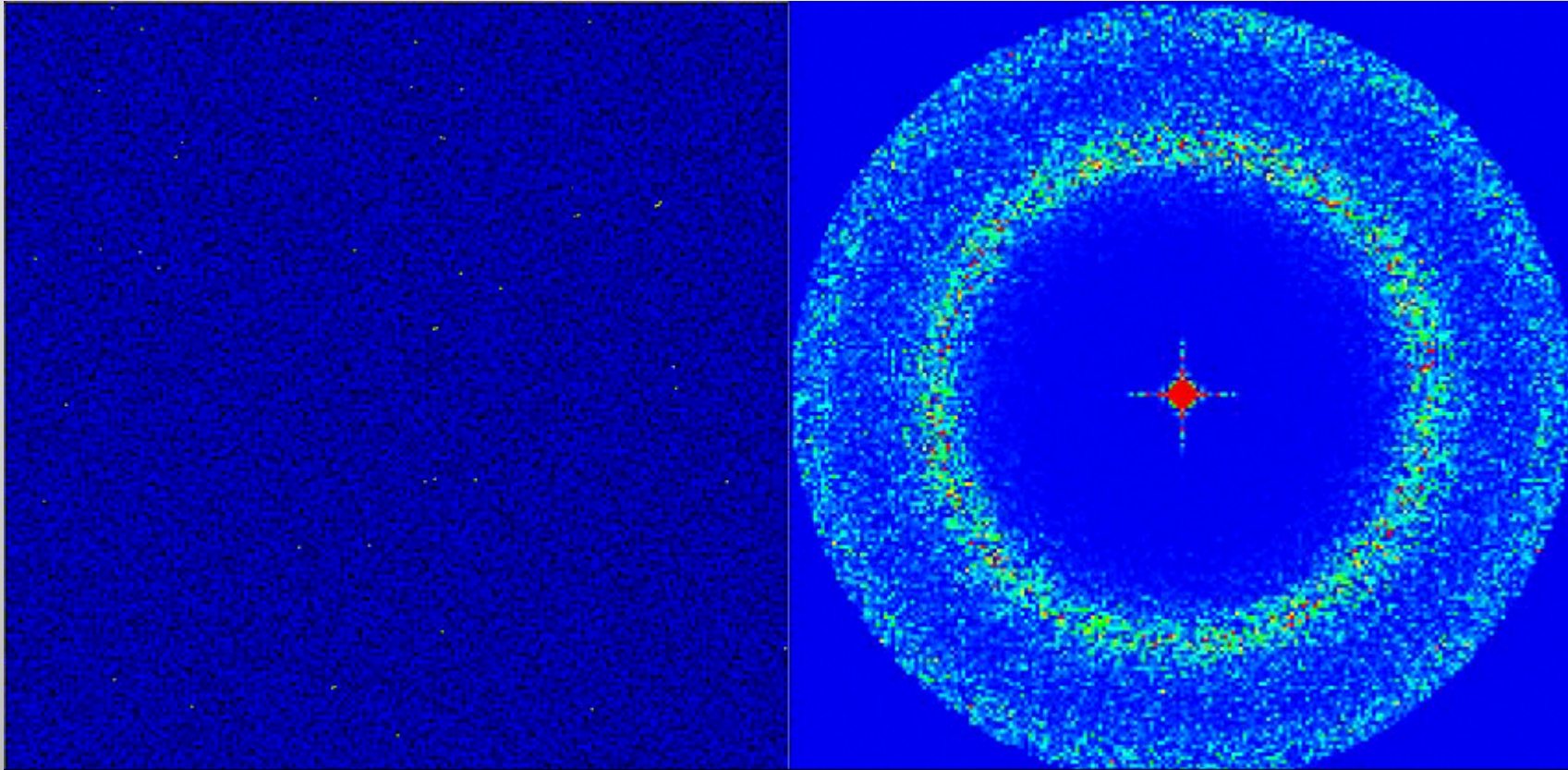
Solution:

- Particle-based domain decomposition - processors own particles, not regions - allows decomposition to persist through atom movement
- Maintain minimum communication list for given decomposition - allows extended range of "interaction"
- Arbitrary domain shape - allows minimal surface to volume ratio for communication

ExMatEx: Exascale is about better Physics Fidelity: Coupling Atomic with Microstructural Scales

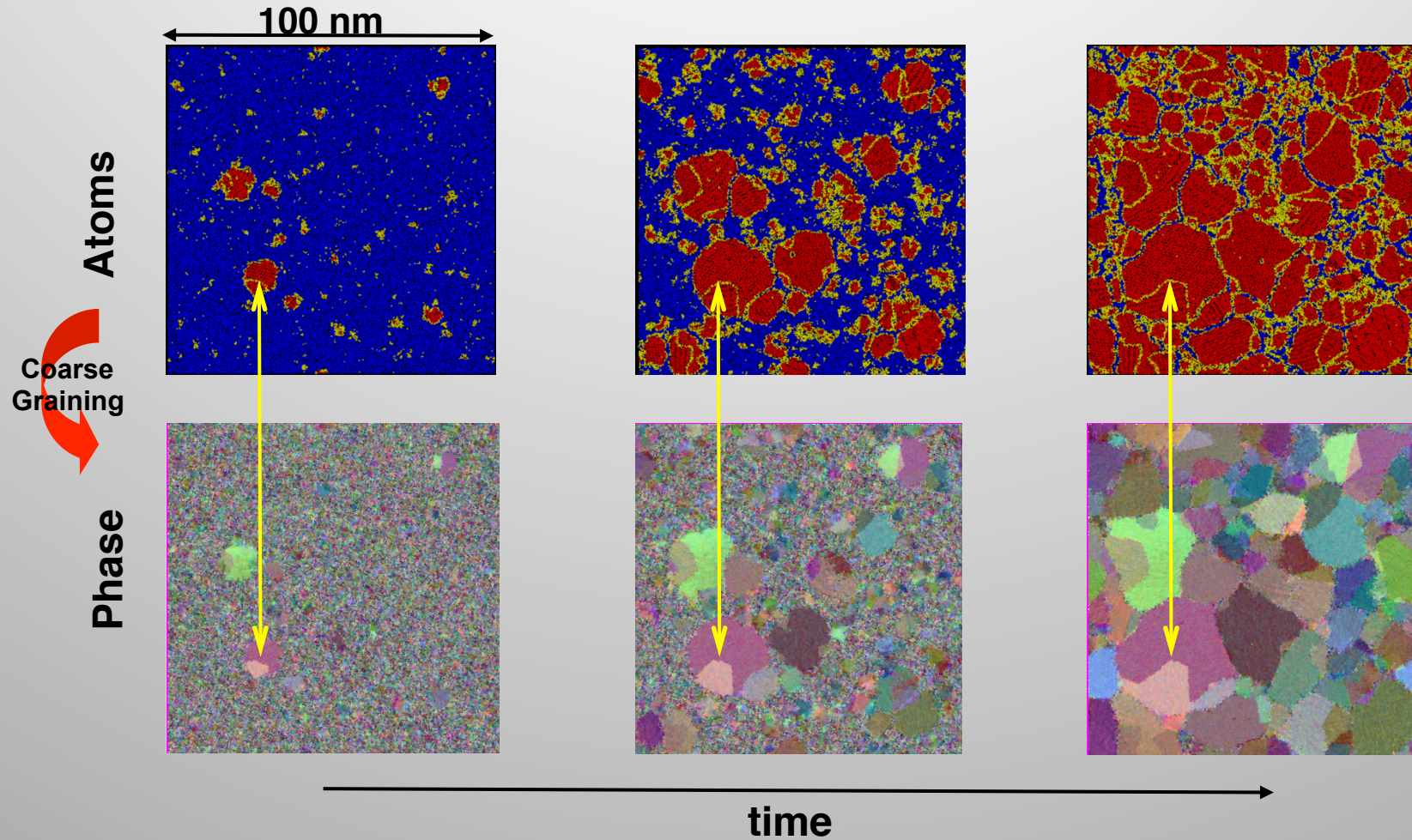
Ab-initio	Atoms	Long-time	Microstructure	Dislocation	Crystal	Continuum
Inter-atomic forces, EOS, excited states	Defects and interfaces, nucleation	Defects and defect structures	Meso-scale multi-phase, multi-grain evolution	Meso-scale strength	Meso-scale material response	Macro-scale material response
						
Code: Qbox/ LATTE	Code:SPaSM/ ddcMD/CoMD	Code: SEAKMC	Code: AMPE/GL	Code: ParaDiS	Code: VP-FFT	Code: ALE3D/ LULESH
Motif: Particles and wavefunctions, plane wave DFT, ScaLAPACK, BLACS, and custom parallel 3D FFTs	Motif: Particles, explicit time integration, neighbor and linked lists, dynamic load balancing, parity error recovery, and <i>in situ</i> visualization	Motif: Particles and defects, explicit time integration, neighbor and linked lists, and <i>in situ</i> visualization	Motif: Regular and adaptive grids, implicit time integration, real-space and spectral methods, complex order parameter	Motif: “segments” Regular mesh, implicit time integration, fast multipole method	Motif: Regular grids, tensor arithmetic, meshless image processing, implicit time integration, 3D FFTs.	Motif: Regular and irregular grids, explicit and implicit time integration.
Prog. Model: MPI + CUBLAS/CUDA	Prog. Model: MPI + Threads	Prog. Model: MPI + Threads	Prog. Model: MPI	Prog. Model: MPI	Prog. Model: MPI + Threads	Prog. Model: MPI + Threads

Molecular Dynamics (MD) are now large enough to model the initiation of realistic microstructure

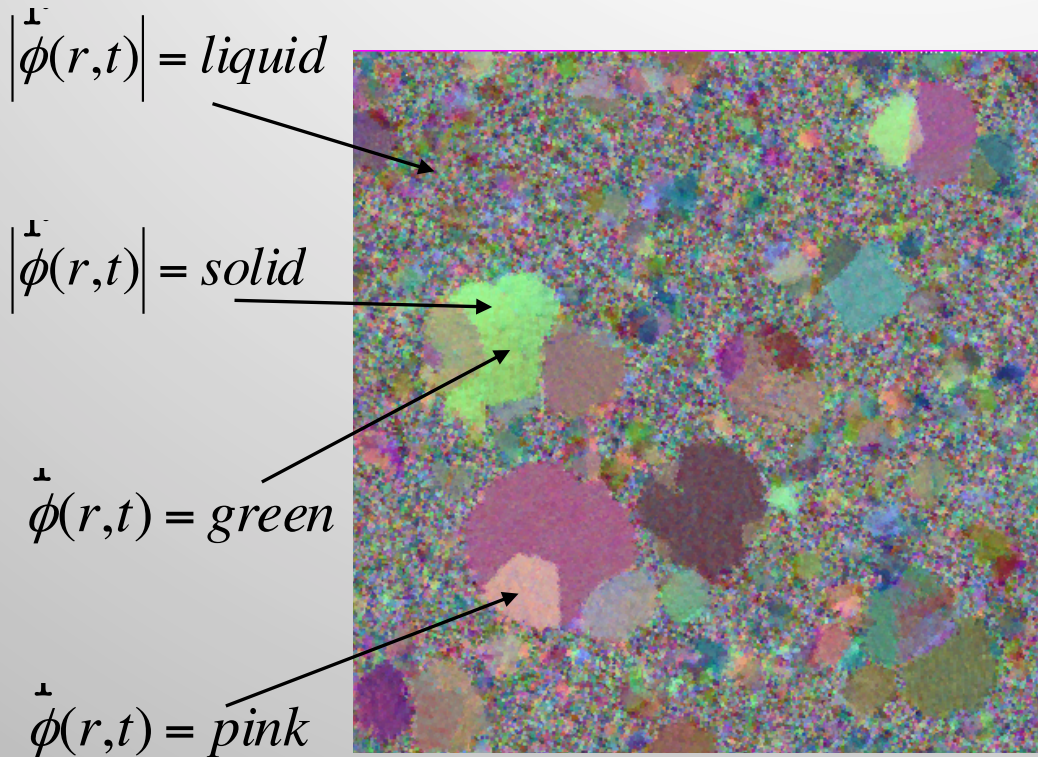


Simulations suggest novel in situ x-ray scattering experiments using emerging sources such as LCLS

Multi-scale paradigm: Phase-field model and MD simulations that overlap in space and time



What is Phase Field modeling? - PFM



Thermodynamic representation of phase (or “color”) everywhere

- Each color represents a different value of the phase field ϕ (solid orientation)
- Free energy describes how colors interact and evolve
- Accuracy depends on fidelity of physics in the equations

Evolution Equations

$$F(P,T) = \int dx \left[\Gamma |\nabla \phi|^2 + f(\phi, P, T) + \mathbb{K} \right]$$

$$\frac{\partial \phi}{\partial t} = -\Gamma \frac{\delta F}{\delta \phi} + \text{noise}$$

What does a crystallographic-aware phase-field model of polycrystal solidification look like?

Pusztai et al., have proposed a 3D quaternion-based phase-field model

- Represents crystal orientation with quaternion order parameter
- Quaternions are widely used to analyze crystallography of polycrystal interfaces
- Quaternion algebra is fast, efficient, avoids singularities, ...

Free Energy

$$F = \int \left[\frac{\epsilon_\phi^2}{2} |\nabla \phi|^2 + f(\phi, c, T) + HT[1 - p(\phi)] \left(\sum_i (\nabla q_i)^2 \right)^{1/2} \right] d^3 r$$

Evolution

$$\frac{\partial q_i}{\partial t} = -M_q \frac{\delta F}{\delta q_i} + \xi_i = M_q \left[\nabla \cdot \left(D \frac{\nabla q_i}{|\nabla q_i|} \right) - 2\lambda q_i \right] + \xi_i$$

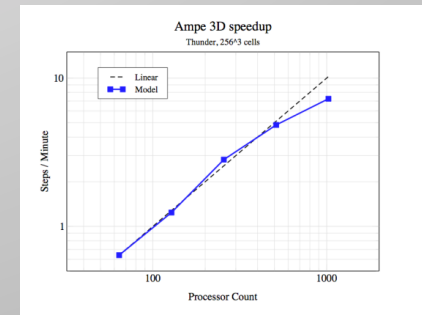
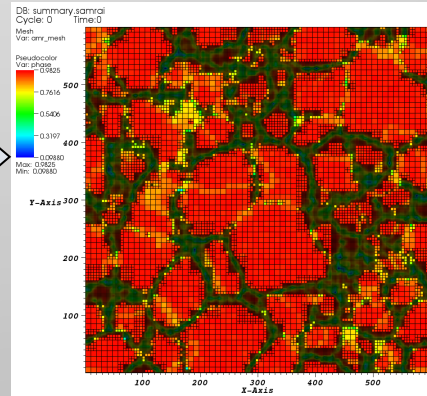
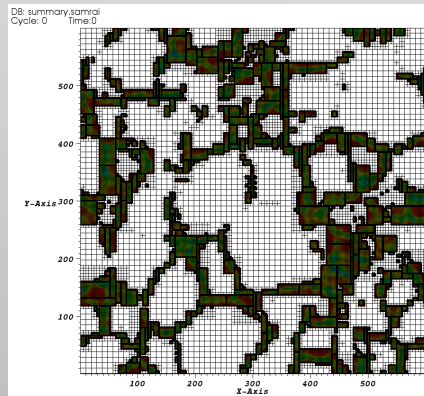
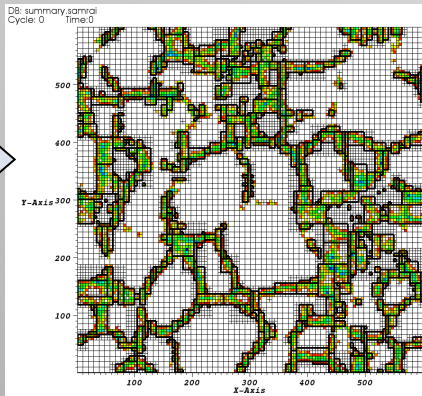
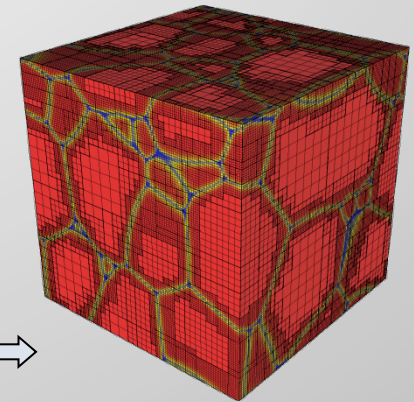
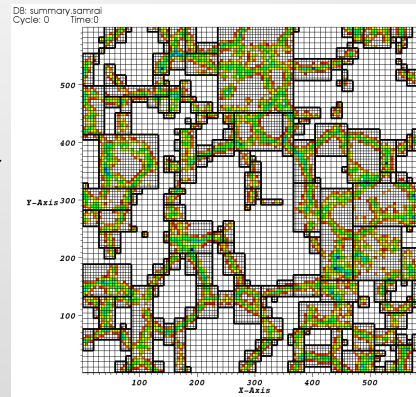
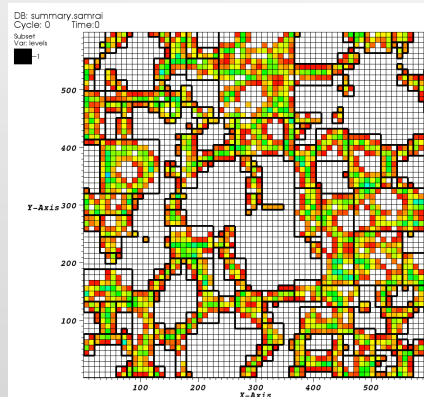
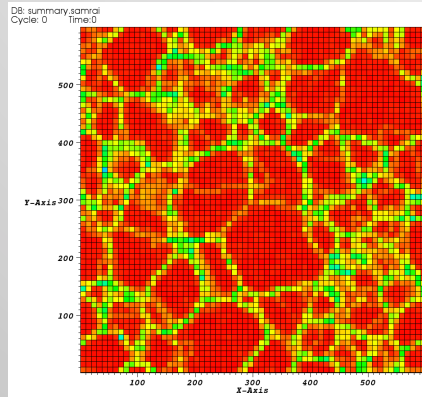
Where q_i is the quaternion order parameter, M_q is the associated mobility and ξ is the fluctuation in q .

We have implemented the Pusztai model in our 3D AMR code

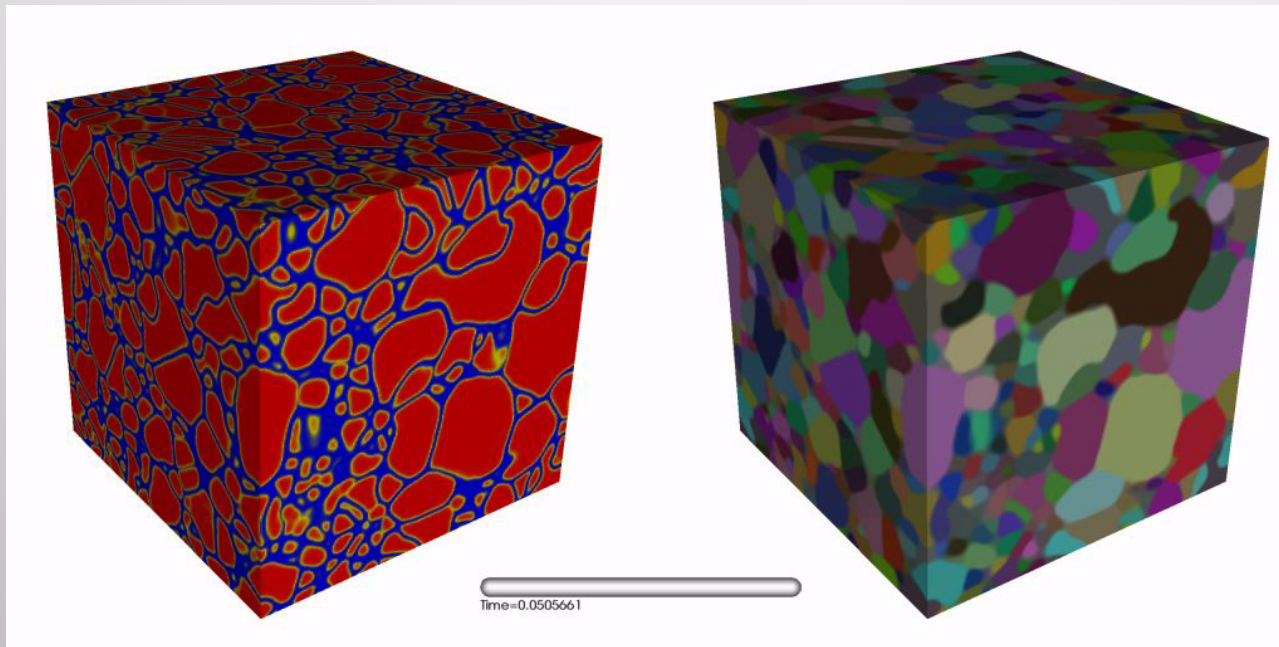
- Enhance energy functional to represent energetics of grain boundaries
- Crystal symmetry aware quaternion mathematics
- Extend energy functional to include elasticity and alloy concentration

Refs: T. Pusztai, G. Bortel, and L. Granasy, "Phase field theory of polycrystalline solidification in three dimensions," Europhys. Lett, 71 (2005) 131-137; R. Kobayoshi and J.A Warren, "Modeling the formation and dynamics of polycrystals in 3D," Physica A 356 (2005) 127-132.

Representation of MD Data onto the AMR Grid Hierarchy using the SAMRAI AMR Library



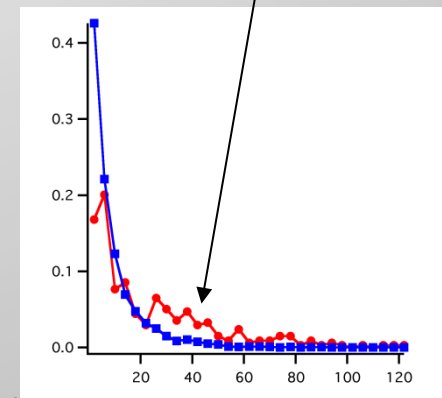
MD nucleated microstructure onto the micro-second hydro time-scale with the crystallographic quaternion model



Phase Order Parameter

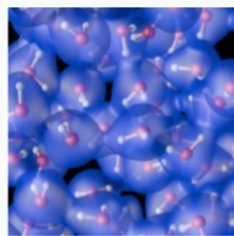
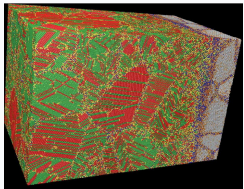
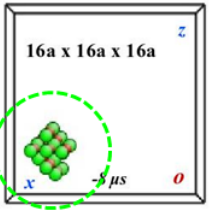
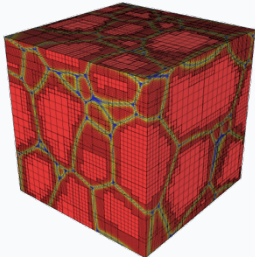
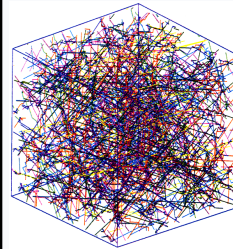
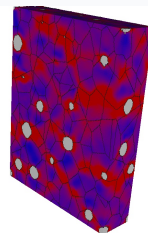
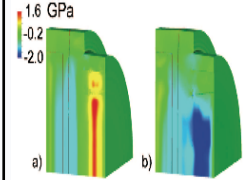
Quaternion Order Parameter

Growth of large grains
Blue: MD nucleation
Red: Phase-field evolution



While significant grain coarsening has occurred on the microsecond scale, the microstructure is far from log-normal

ExMatEx: Exascale is about better Physics Fidelity: Adaptive Physics Refinement

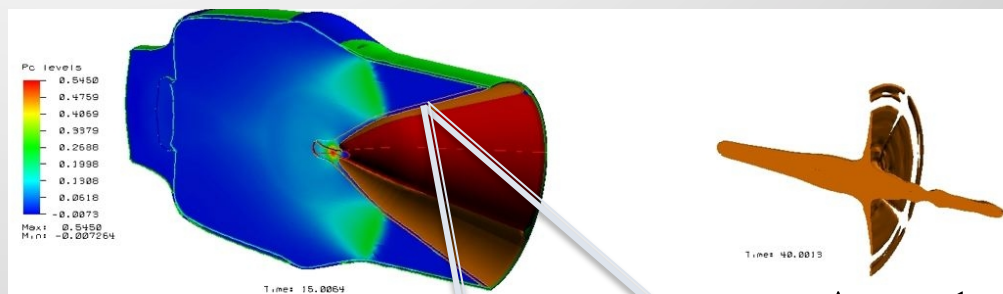
Ab-initio	Atoms	Long-time	Microstructure	Dislocation	Crystal	Continuum
Inter-atomic forces, EOS, excited states	Defects and interfaces, nucleation	Defects and defect structures	Meso-scale multi-phase, multi-grain evolution	Meso-scale strength	Meso-scale material response	Macro-scale material response
						
Code: Qbox/ LATTE Motif: Particles and wavefunctions, plane wave DFT, ScaLAPACK, BLACS, and custom parallel 3D FFTs Prog. Model: MPI + CUBLAS/CUDA	Code:SPaSM/ ddcMD/CoMD Motif: Particles, explicit time integration, neighbor and linked lists, dynamic load balancing, parity error recovery, and <i>in situ</i> visualization Prog. Model: MPI + Threads	Code: SEAKMC Motif: Particles and defects, explicit time integration, neighbor and linked lists, and <i>in situ</i> visualization Prog. Model: MPI + Threads	Code: AMPE/GL Motif: Regular and adaptive grids, implicit time integration, real-space and spectral methods, complex order parameter Prog. Model: MPI	Code: ParaDiS Motif: “segments” Regular mesh, implicit time integration, fast multipole method Prog. Model: MPI	Code: VP-FFT Motif: Regular grids, tensor arithmetic, meshless image processing, implicit time integration, 3D FFTs. Prog. Model: MPI + Threads	Code: ALE3D/ LULESH Motif: Regular and irregular grids, explicit and implicit time integration. Prog. Model: MPI + Threads

Use Case: Shaped-charge jets, breakup and 3D effects (e.g. spinning) require crystal plasticity and anisotropy

What is required:

Resolution: 10^{12} zones (10 cm cube)
Simulation time: 100 μ sec (10^5 steps)
Strain rate: 10^6 /sec
Strain: 1-3
Using Small Strain Crystal Plasticity Model:
 $\sim 10^4$ sec (~ 3 h) wall clock on 10^9 cores
Large Strain Crystal Plasticity Model: 10x
Twinning / Scale Bridging Model: 100x

ALE3D simulation of shaped-charge jet
(Rose McCallen, LLNL)



$$\Delta \varepsilon \geq 1$$

What we can do today:

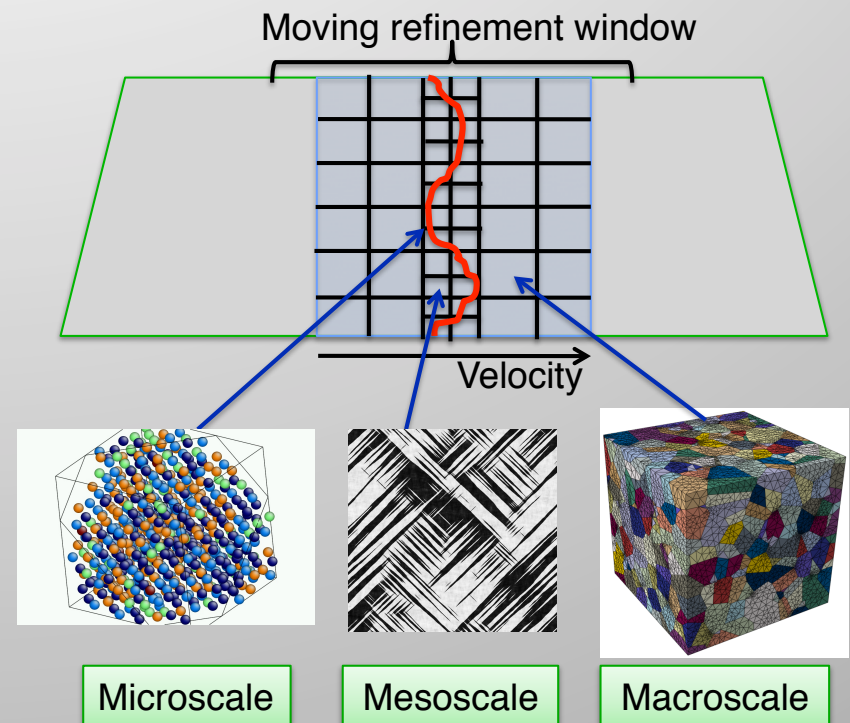
Crystal plasticity simulation of high rate deformation (Nathan Barton, LLNL)
Model: Small Strain Crystal Plasticity
Number Zones: 10^7 (100 micron cube)
Simulation time: 10 μ sec (10^4 steps)
Strain rate: 10^6 /sec
Strain: 0.15
Wall Clock: 1 day on 1/10 Cielo



$$\Delta \varepsilon = 0.15$$

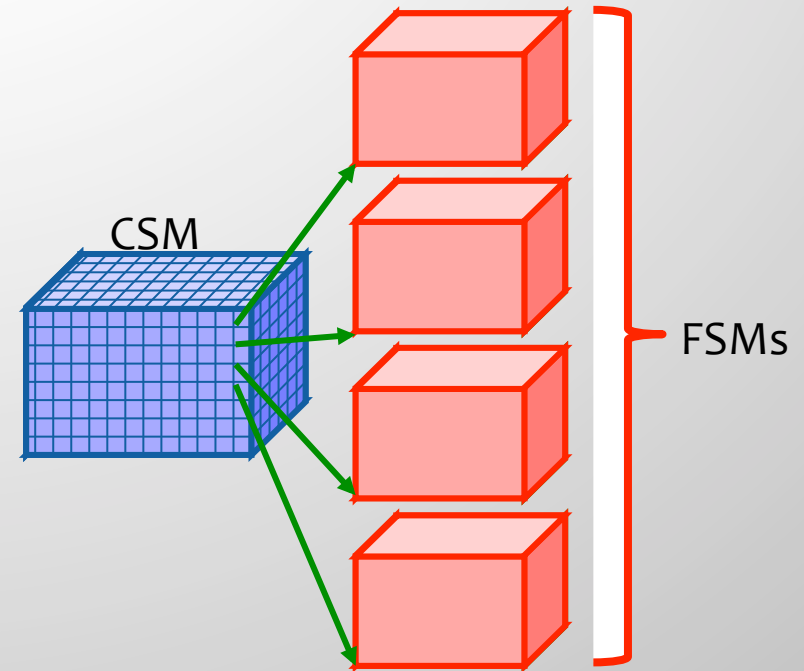
Embedded Scale-Bridging Algorithms

- Our goal is to introduce more detailed physics into computational materials science applications in a way which escapes the traditional synchronous SPMD paradigm and exploits the heterogeneity expected in exascale hardware.
- To achieve this, we are developing a UQ-driven *adaptive physics refinement* approach.
- Coarse-scale simulations dynamically spawn tightly coupled and self-consistent fine-scale simulations as needed.
- This *task-based* approach naturally maps to exascale heterogeneity, concurrency, and resiliency issues.



Direct multi-scale embedding requires full utilization of exascale concurrency and locality

- *Brute force multi-scale coupling*: Full fine scale model (FSM, e.g. a crystal plasticity model) run for every zone & time step of coarse scale mode (CSM, e.g. an ALE code)
- *Adaptive Sampling*:
 - Save FSM results in database
 - Before running another FSM, check database for FSM results similar enough to those needed that interpolation or extrapolation suffices
 - Only run full FSM when results in database not close enough

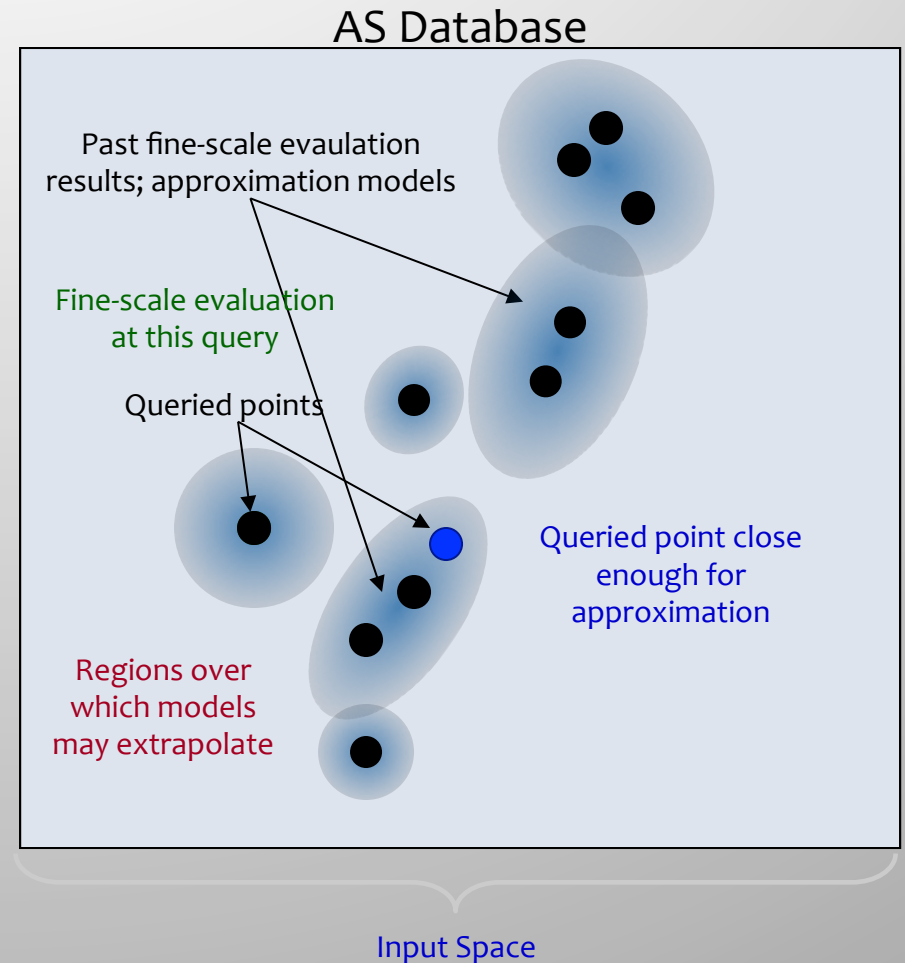


- Heterogeneous, hierarchical MPMD algorithms map naturally to anticipated heterogeneous, hierarchical architectures
- Escape the traditional bulk synchronous SPMD paradigm, improve scalability and reduce scheduling
- Task-based MPMD approach leverages concurrency and heterogeneity at exascale while enabling novel data models, power management, and fault tolerance strategies

Ref: Barton et.al, 'A call to arms for task parallelism in multi-scale materials modeling,' *Int. J. Numer. Meth. Engng* 2011; 86:744–764

Adaptive Sampling builds response on the fly

- Coarse scale model queries database for fine-scale material response
- If possible, approximate response from past evaluations
- Otherwise perform fine scale evaluation
- Fine-scale evaluations grow database

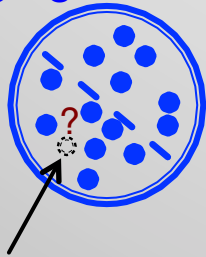


Kriging estimates are based on previously computed fine-scale responses.

Fine-scale responses accumulated in a database are interpolated (with error estimation) via a kriging algorithm.

- = fine scale evaluation
- == = linear regression model

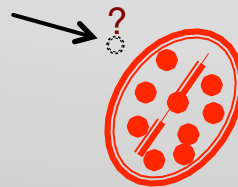
Kriging model 1



Sample point near existing model and satisfies tolerance:

- Just interpolate (saves fine-scale evaluation)

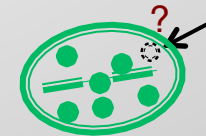
Kriging model 2



Sample point too far from existing models:

- Evaluate fine scale
- Create new model

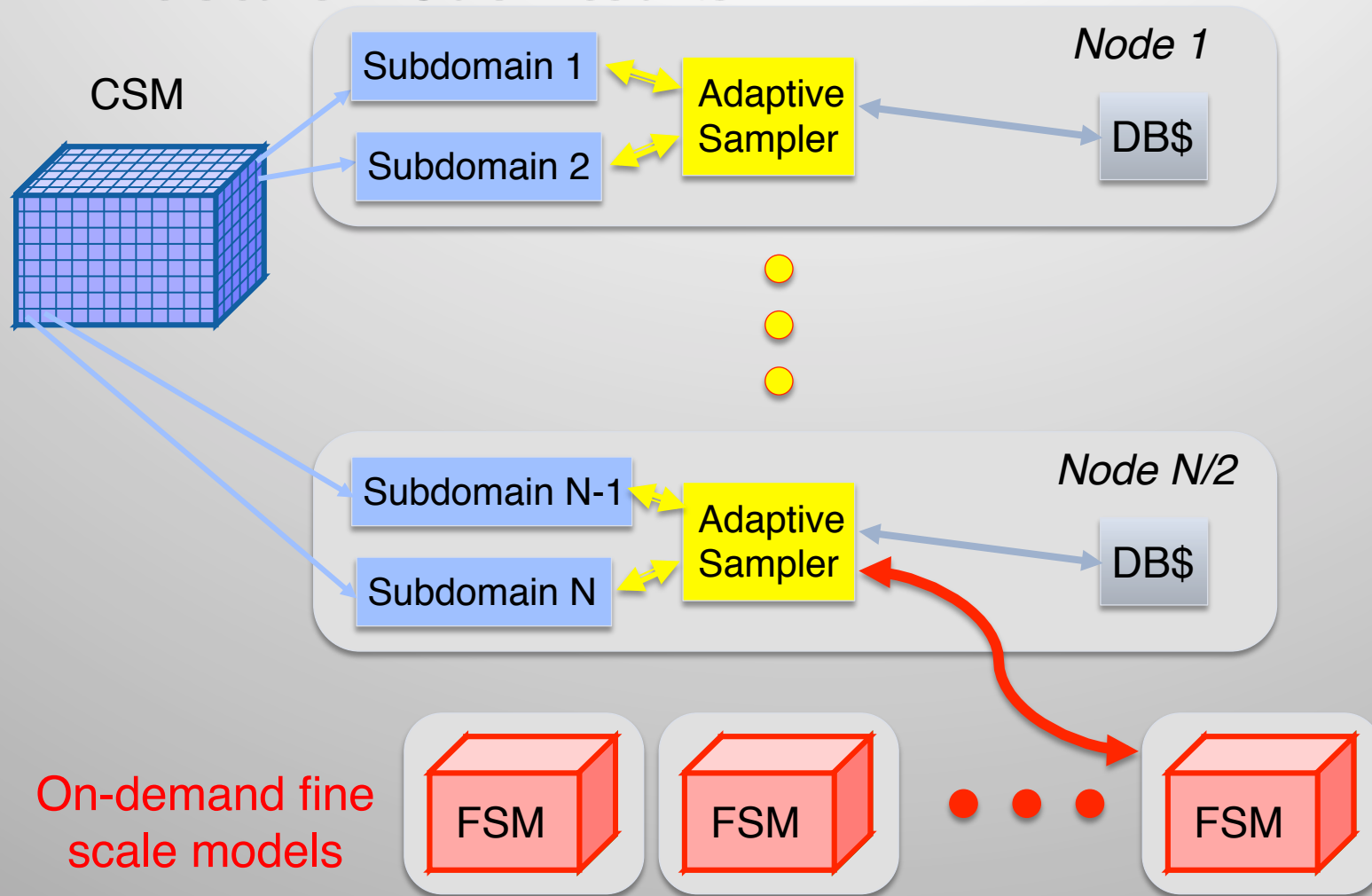
Kriging model 3



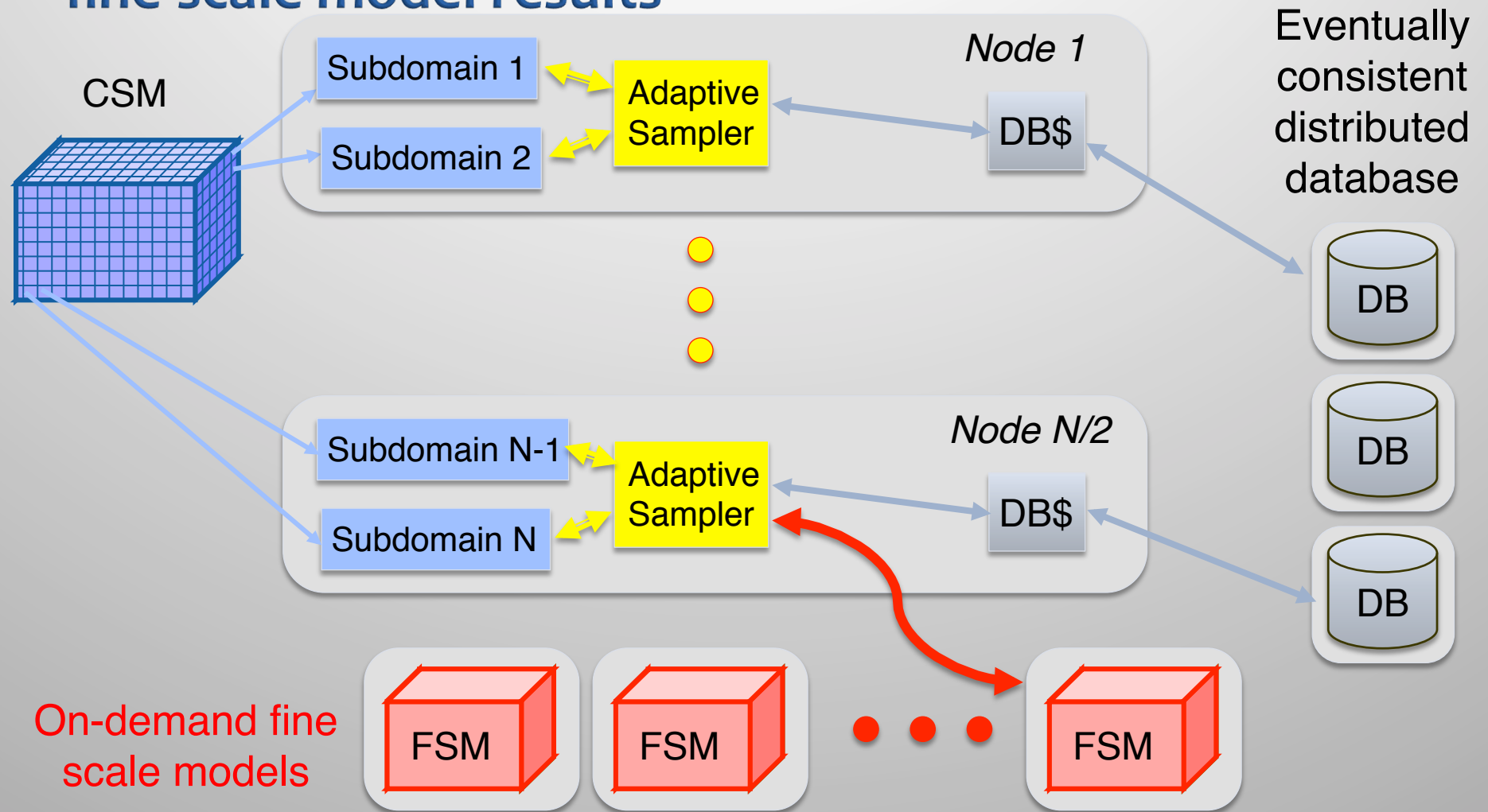
Sample point near existing model, but fails error tolerance:

- Evaluate fine scale
- Add to existing model

Tradeoff: re-use vs. re-computation of expensive fine-scale model results



Tradeoff: re-use vs. re-computation of expensive fine-scale model results



Co-Design Summary

- Our **goal** is to establish the interrelationship between hardware, middleware (software stack), programming models, and algorithms required to enable **a productive exascale environment** for multiphysics simulations of materials in extreme mechanical and radiation environments.
- We will exploit, rather than avoid, the greatly increased levels of concurrency, heterogeneity, and flop/byte ratios on the upcoming exascale platforms.
- Our **vision** is an uncertainty quantification (UQ)-driven *adaptive physics refinement* in which meso- and macro-scale materials simulations spawn micro-scale simulations as needed.
 - This *task-based* approach leverages the extensive concurrency and heterogeneity expected at exascale while enabling fault tolerance within applications.
 - The programming models and approaches developed to achieve this will be broadly applicable to a variety of multiscale, multiphysics applications, including astrophysics, climate and weather prediction, structural engineering, plasma physics, and radiation hydrodynamics.

