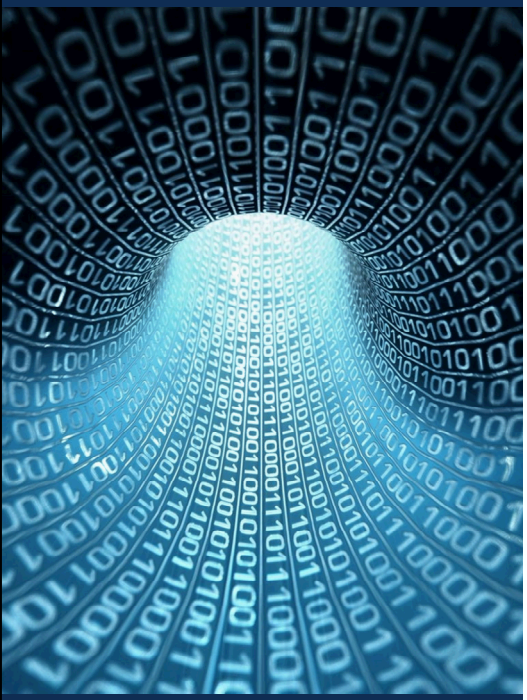
SAND2016-0930C



Sandia National Laboratories

*Exceptional service in the national interest*

# The DHARMA Approach to Asynchronous Many-Task Programming

David S. Hollman, Jeremiah J. Wilke, Nicole Slattengren, Hemanth Kolla, Francesco Rizzi, Keita Teranishi, Janine C. Bennett (PI), Robert L. Clay (PM)

PSAAP-West

January 28, 2016

U.S. DEPARTMENT OF ENERGY | NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2011-XXXXP

# What are core requirements for next-generation applications?

Sandia National Laboratories

- An abstraction layer between applications and the runtime system/physical architecture
- An implementation of a software stack that supports this abstraction layer
- Community best practices and eventual standards for this abstraction layer

# What are core requirements for next-generation applications?

Sandia National Laboratories

- An abstraction layer between applications and the runtime system/physical architecture
- An implementation of a software stack that supports this abstraction layer
- Community best practices and eventual standards for this abstraction layer

The DHARMA project at Sandia performs AMT runtime system R&D to address these core application requirements

# Asynchronous Many-Task (AMT) runtimes address key performance challenges posed by future architectures

Sandia National Laboratories

- Performance challenges:
  - Utilizing whole machine requires more parallelism
  - Managing deep memory hierarchies requires flexible staging of data/assigning work
  - Handling dynamic workloads requires flexible task scheduling

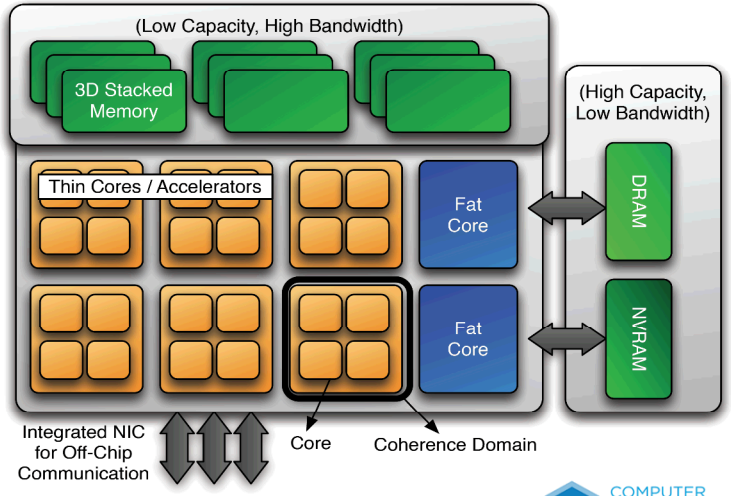



Image courtesy of www.cal-design.org

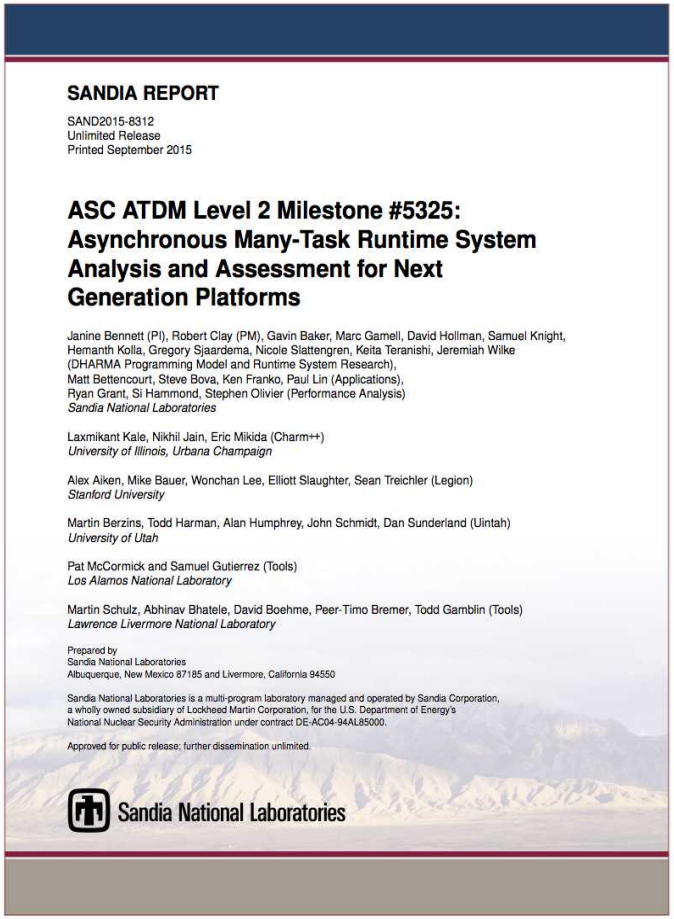
COMPUTER ARCHITECTURE LABORATORY
EXASCALE DESIGN SPACE EXPLORATION

- **Asynchronous**: express all possible parallelism and minimize/hide communication/scheduling latency
- **Many-Task**: Chunks of work of ``correct'' granularity that can be flexibly assigned to different memory/execution spaces

# Sandia led a comparative analysis study of leading AMT runtimes to inform our technical roadmap

Sandia National Laboratories

- Broad survey of many AMT runtime systems
- Deep dive on Charm++, Legion, Uintah
- Programmability
  - Does this runtime enable efficient expression of our workloads?
- Performance
  - How performant is this runtime for our workloads on current platforms?
  - How well suited is this runtime to address exascale challenges?
- Mutability
  - What is the ease of adopting this runtime and modifying it to suit our needs?



**SANDIA REPORT**

SAND2015-8312
Unlimited Release
Printed September 2015

**ASC ATDM Level 2 Milestone #5325: Asynchronous Many-Task Runtime System Analysis and Assessment for Next Generation Platforms**

Janine Bennett (PI), Robert Clay (PM), Gavin Baker, Marc Gamell, David Hollman, Samuel Knight, Hemanth Kolla, Gregory Sjaardema, Nicole Slattengren, Keita Teranishi, Jeremiah Wilke (DHARMA Programming Model and Runtime System Research), Matt Bettencourt, Steve Bova, Ken Franko, Paul Lin (Applications), Ryan Grant, Si Hammond, Stephen Olivier (Performance Analysis)
*Sandia National Laboratories*

Laxmikant Kale, Nikhil Jain, Eric Mikida (Charm++)
*University of Illinois, Urbana Champaign*

Alex Aiken, Mike Bauer, Wonchan Lee, Elliott Slaughter, Sean Treichler (Legion)
*Stanford University*

Martin Berzins, Todd Harman, Alan Humphrey, John Schmidt, Dan Sunderland (Uintah)
*University of Utah*

Pat McCormick and Samuel Gutierrez (Tools)
*Los Alamos National Laboratory*

Martin Schulz, Abhinav Bhatele, David Boehme, Peer-Timo Bremer, Todd Gamblin (Tools)
*Lawrence Livermore National Laboratory*

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.

Sandia National Laboratories

# Lessons learned from study led to application-driven programming model specification co-design effort

Sandia National Laboratories

- Data, task, and pipeline parallelism can be expressed in different ways
  - Explicit parallelism vs apparently sequential semantics
  - Arbitrary data structures vs strong data model
  - Runtime vs user-level control
  - New language vs embedded in C/C++
- Model should enhance performance, productivity, resilience
  - Applications should not be (much) more difficult to write than MPI
  - Make difficult things more tractable, e.g. load balancing fault-tolerance
- Design space tradeoffs need further assessment prior to committing to a single runtime
  - Across variety of applications and architectures
  - Further research required in some aspects of runtime (e.g., resource management)

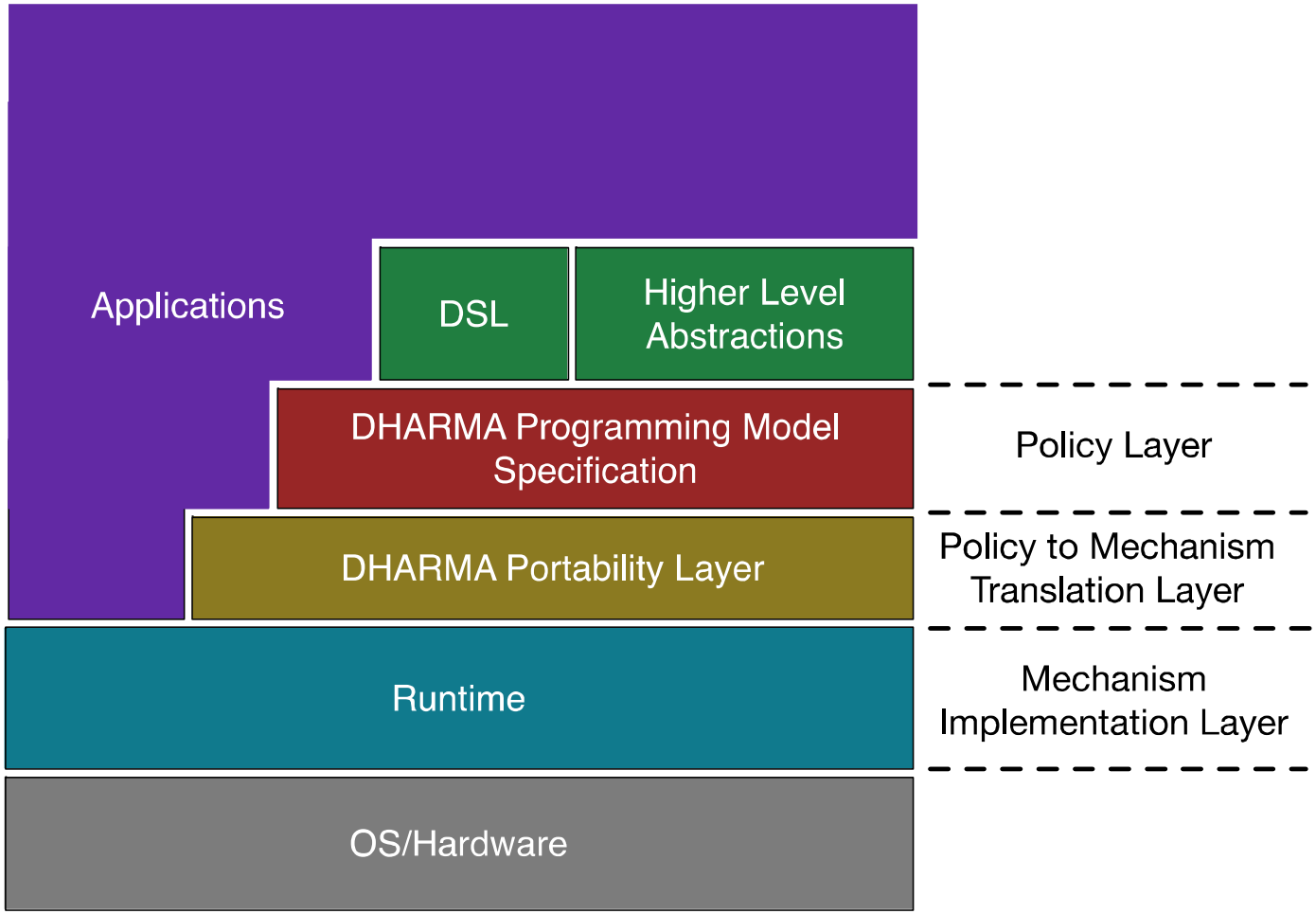
# What do we mean by "application-driven co-design of a programming model specification"?

Sandia National Laboratories

- Runtimes can be decomposed into a specification and implementation
  - The specification is formal documentation that
    - Provides abstractions for expressing what an application does (i.e., the programming model)
    - Can express correctness requirements
    - Can express performance requirements
  - The implementation maps specification requirements onto specific operations/events
- We are co-designing an AMT programming model specification with application and runtime developers
  - Meet our application requirements
  - Generate the runtime requirements

# DHARMA software stack separates policy and mechanism

Sandia National Laboratories

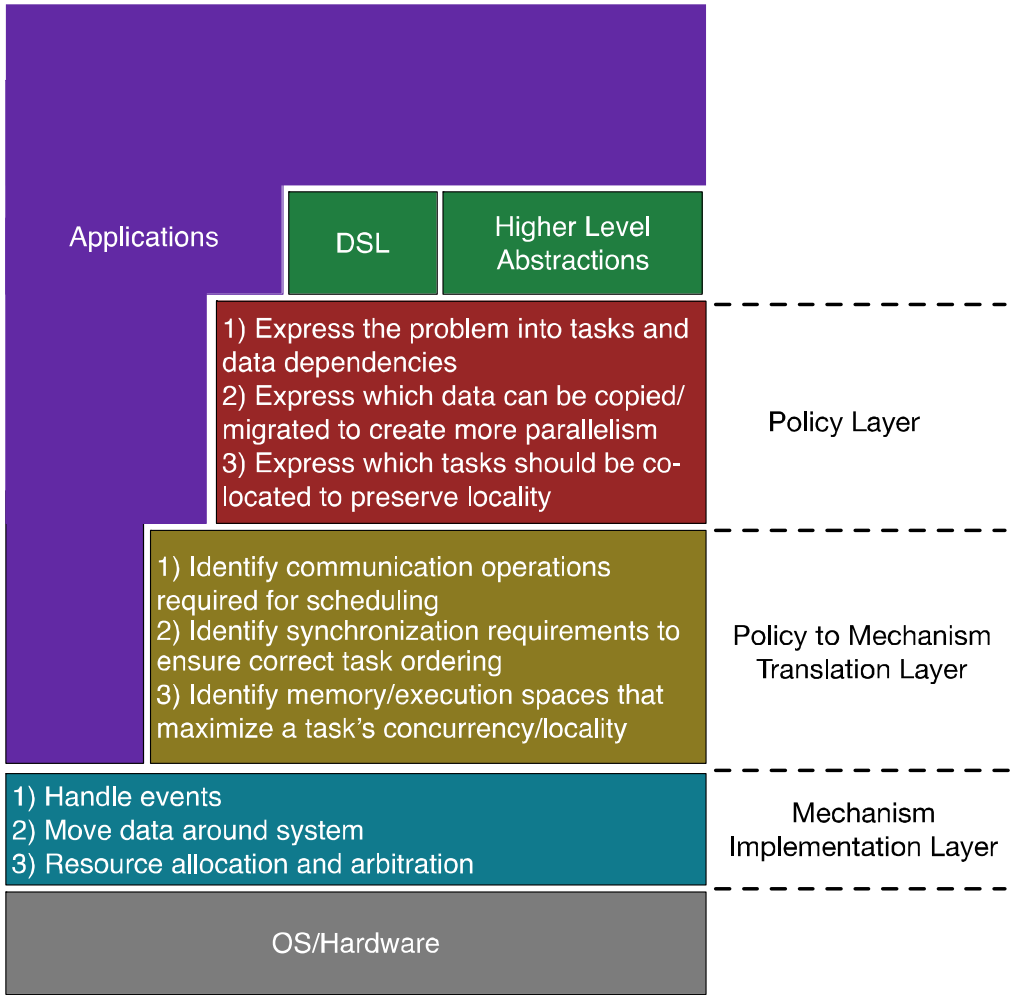


# Expression of policy enables runtime freedom to make complex performance-oriented decisions

Sandia National Laboratories

## Design Intent:

- Applications specify policy
  - Enable rapid development of correct implementation
- Applications can specify mechanism
  - Enable improvement towards performant implementation

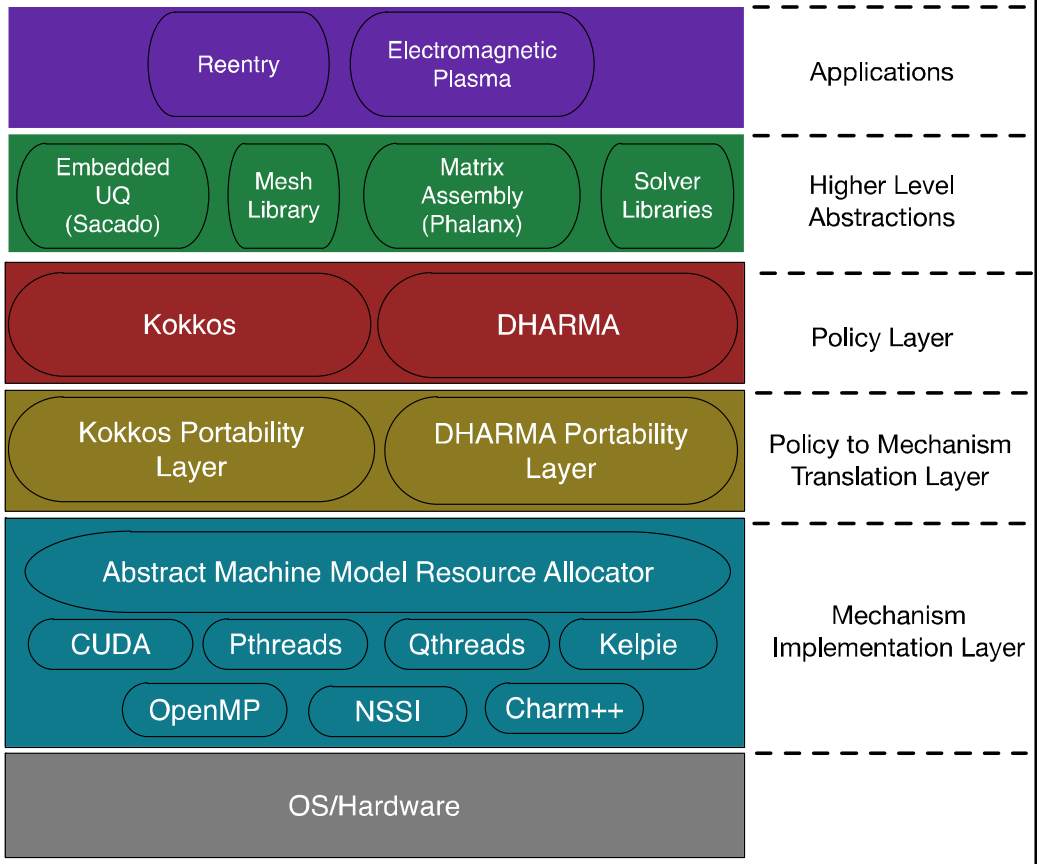


# The separation of policy and mechanism facilitates exploration of runtime design space

Sandia National Laboratories

- AMT software stack working group at Sandia
  - DHARMA
  - Kokkos
  - Data Warehouse/Kelpie
  - Resource allocation and management
  - Qthreads
- Initial implementation of stack this year leveraging Charm++
- Working with community to explore alternative stack implementations
  - OCR, REALM

Sample Sandia Software Stack



# Application requirements shape the initial programming model specification

Sandia National Laboratories

- Application controls initial problem decomposition and distribution
- Runtime should support
  - Efficient SPMD launch and coordination semantics
  - Collectives
  - Basic checkpoint/restart fault recovery supported
  - Replay of tasks for ease of debugging
  - Application-specific data structures/layouts
  - Expression of all forms of parallelism (data, pipeline, task)
- Embed in C++

# The DHARMA programming model specification is a set of parallel semantics embedded in C++ syntax

Sandia National Laboratories

- Declarative, not procedural imperative
- Coordination semantics replace explicit send/recv
- Enqueue work to be performed instead of explicitly (imperatively) doing work immediately or blocking
- As much as possible, preserve sequential semantics to simplify reasoning about code correctness
- *Use standard C++ constructs* (e.g., reference-counted pointers) to manage parallelism
- Do not need to know C++11 to code
  - Works with gcc >= 4.7, clang >= 3.5

# The DHARMA project has three closely-coupled key activities

Sandia National Laboratories

- Co-design AMT programming model specification
  - Gather application requirements for programming model/runtime
  - Assess what runtime requires programming model/application to express
- Implement specification
  - Leverage existing efforts
  - Encourage vendor involvement
- Work with community to define best practices and eventual standards for AMT
  - Collaborating with Tim Mattson's team at Intel on DHARMA programming model specification
  - Recurring engagement with Charm++, OCR, Legion teams

Let us know if you are interested in collaborating in any of these areas!