Title: Los Alamos Programming Models

Author(s): Bergen, Benjamin Karl

Intended for: Teleconference

Issued: 2016-07-07

Los Alamos
NATIONAL LABORATORY
EST.1943

Delivering science and technology
to protect our nation
and promote world stability

UNCLASSIFIED

NNSA
National Nuclear Security Administration

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

Advanced Architectures

Los Alamos Programming Models

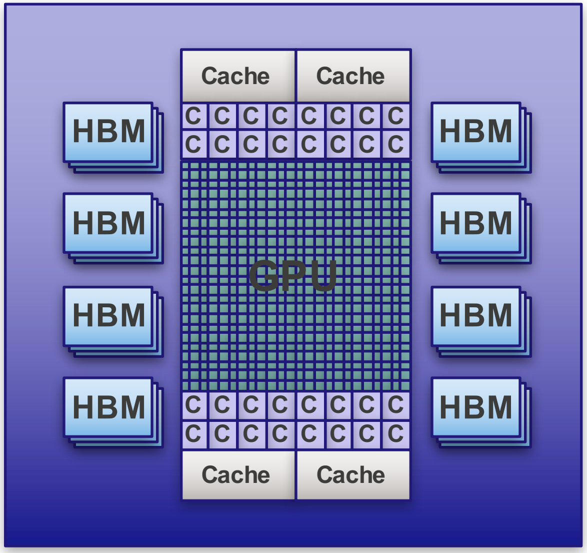
Ben Bergen

July 7th, 2016

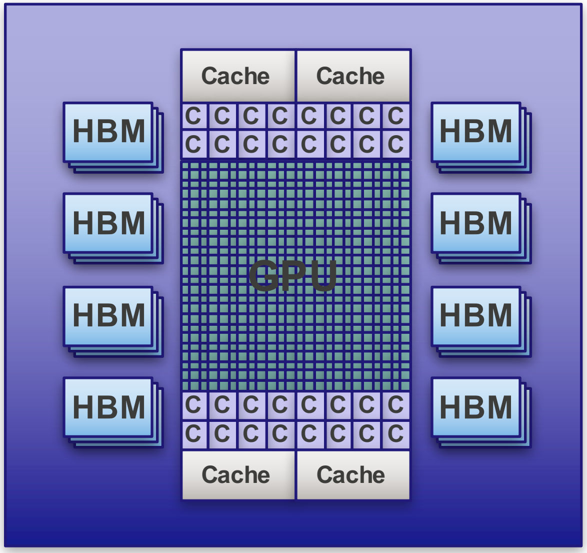
Operated by Los Alamos National Security, LLC for the U.S. Department of Energy's NNSA

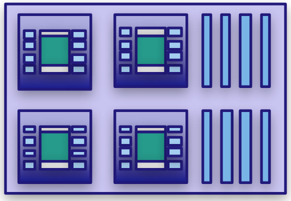
What are our assumptions?

We assume that processors will be heterogeneous…

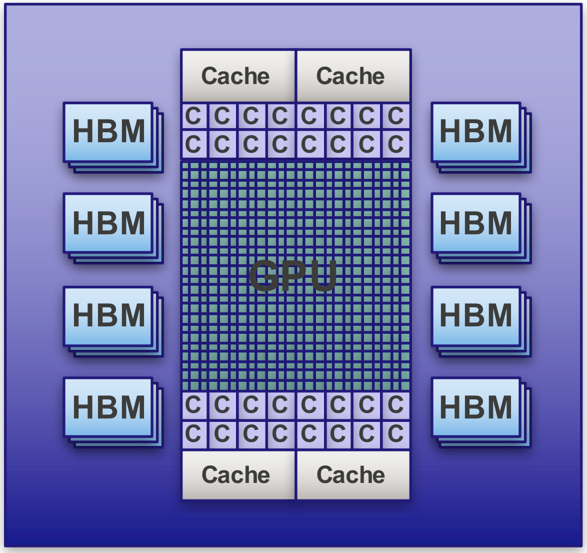


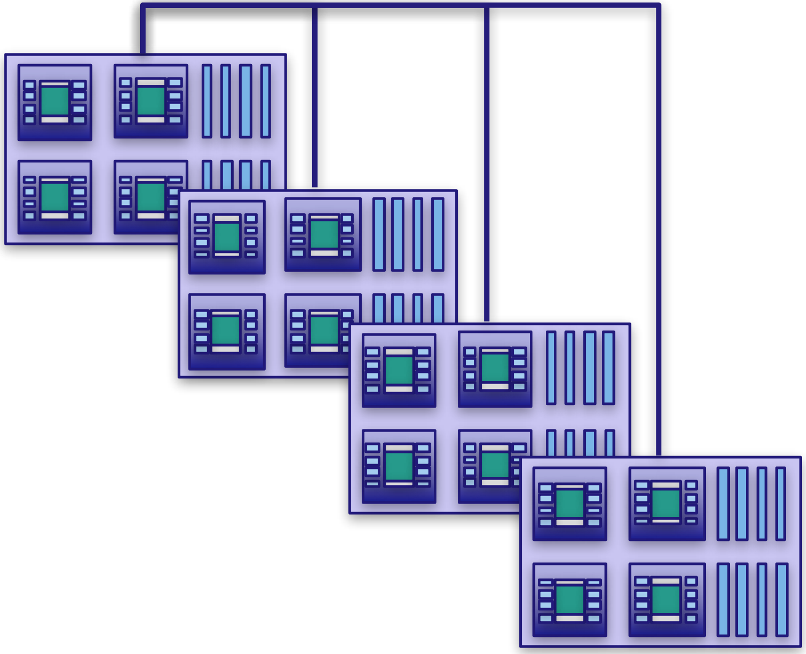
We assume that there will be non-uniform memory access to capacity memory…



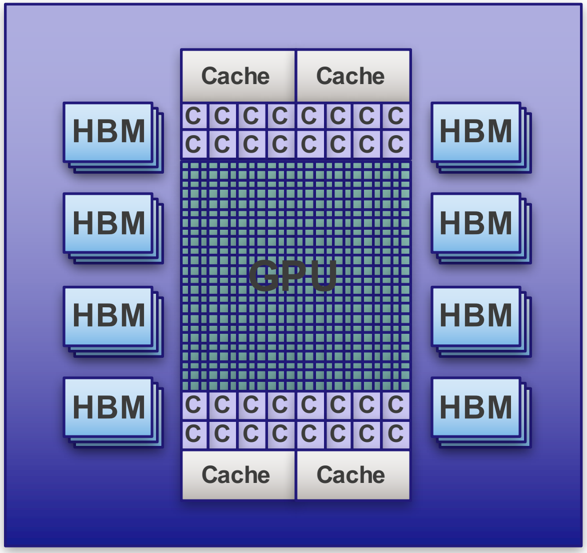


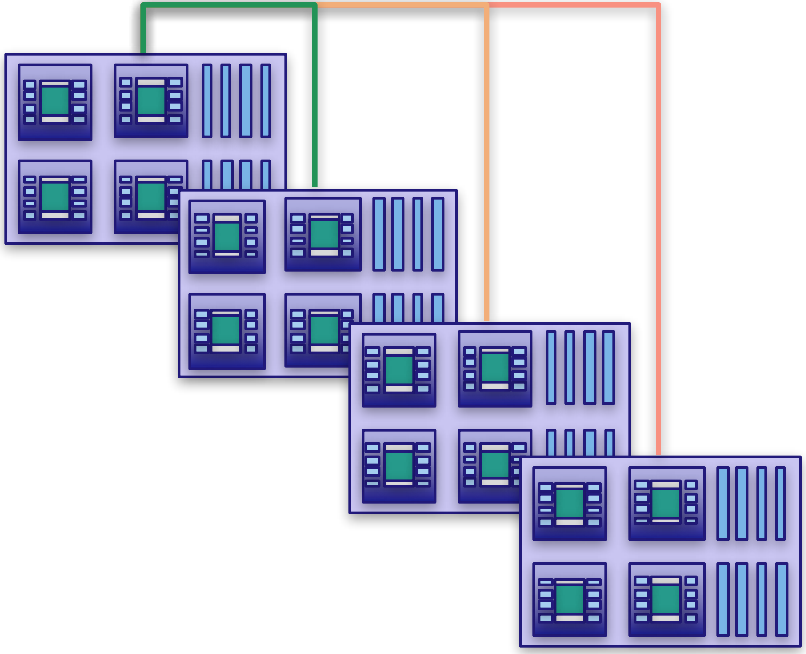
We assume that systems will be composed of inter-connected nodes…



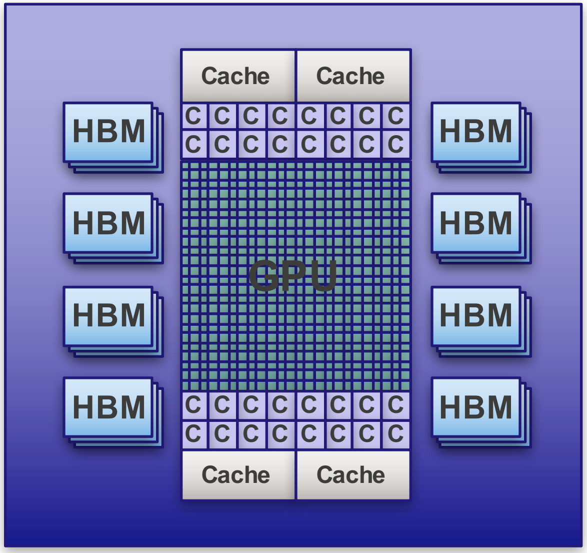


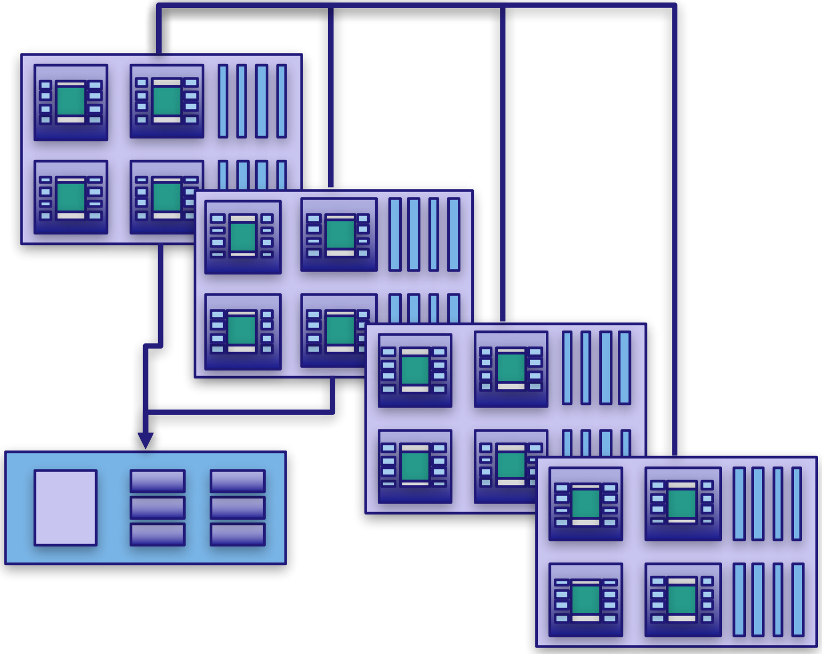
…and that they will have heterogeneous communication behavior...



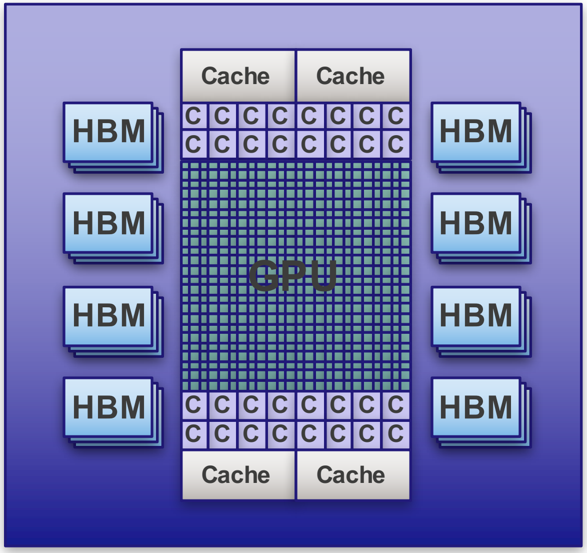


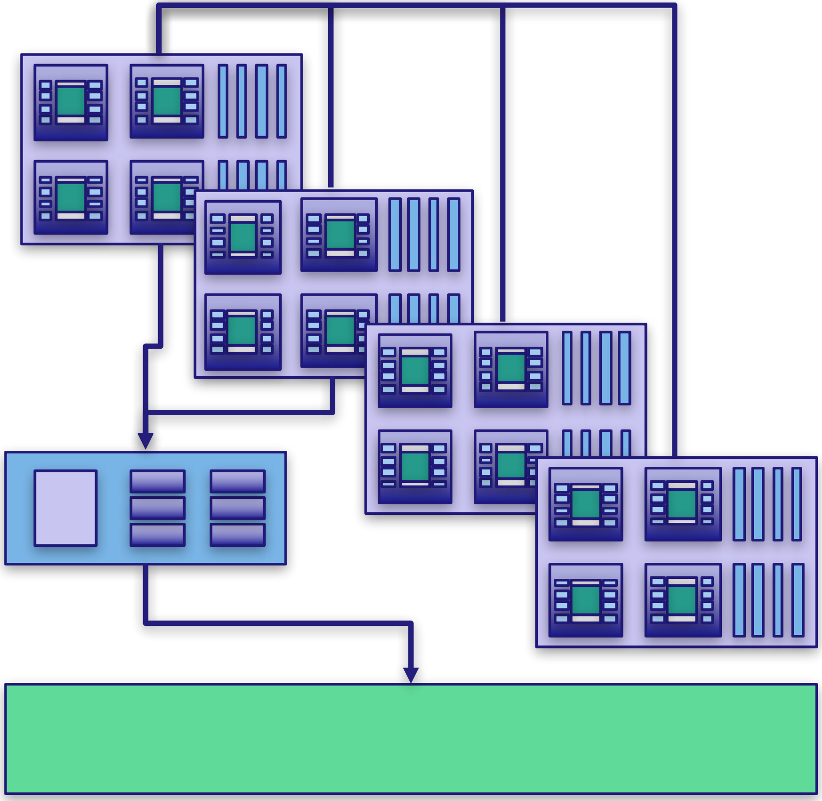
We assume that there will be specialied management and I/O nodes…



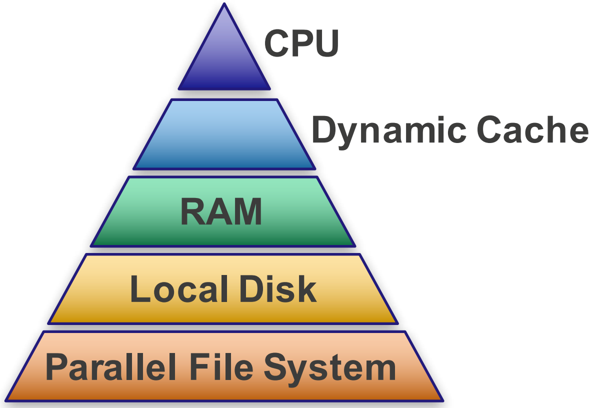


We assume that there will be a parallel file system…

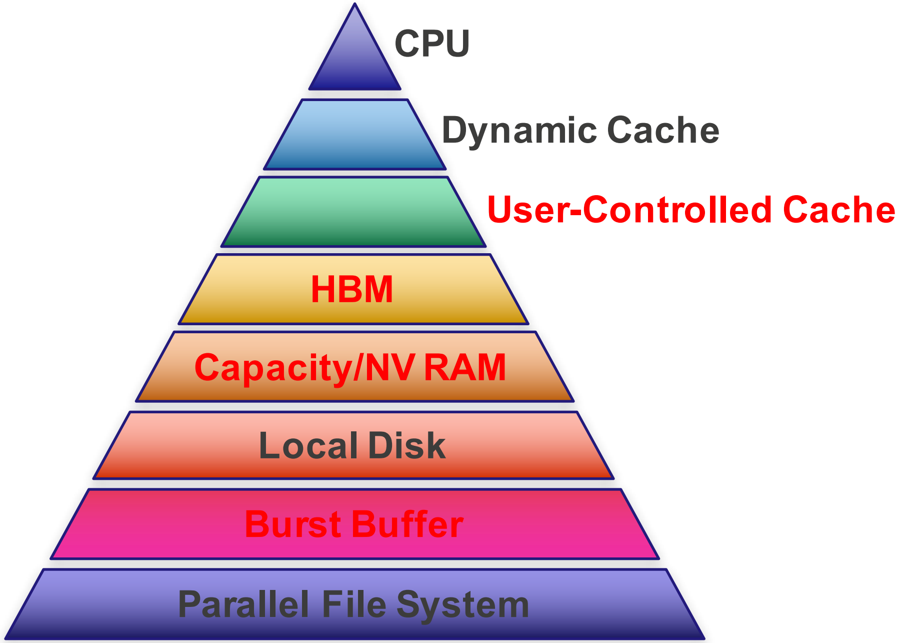




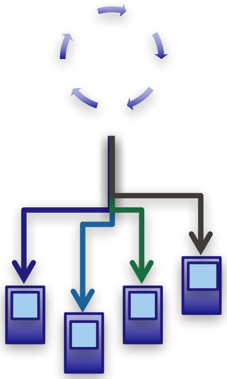
We assume that there will be a more comlex memory and I/O hierarchies…



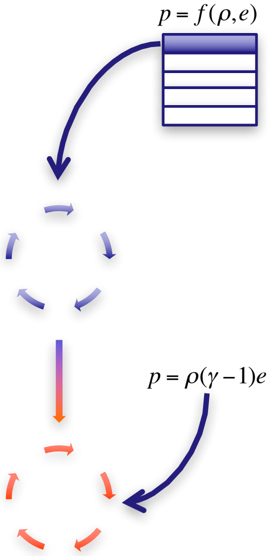
We assume that there will be a more comlex memory and I/O hierarchies…



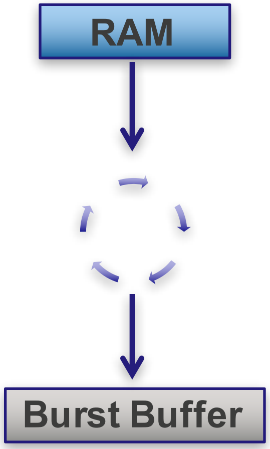
We assume that we will do more complicated methods and workflows…



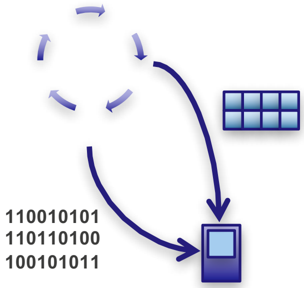
Dynamic Load Balancing



State-Dependent Models



In-situ/In-transit Analysis



Kernel Specialization

We would like to use a hierarchical programming model…

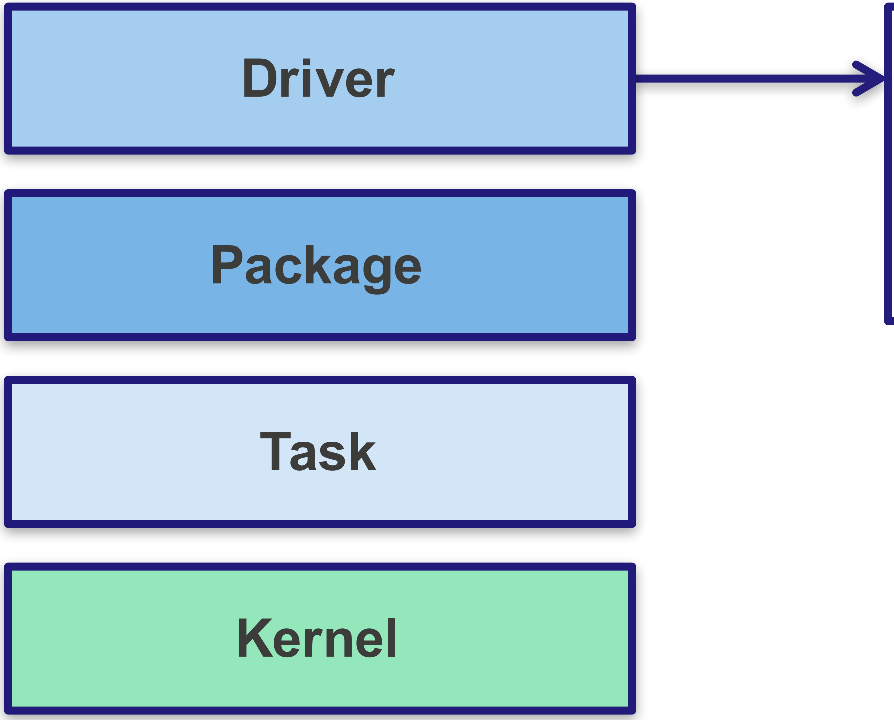
System Level (inter-node)

- Task-Based
- Distributed-Memory
- Data-Centric (runtime understands application data)

Node Level

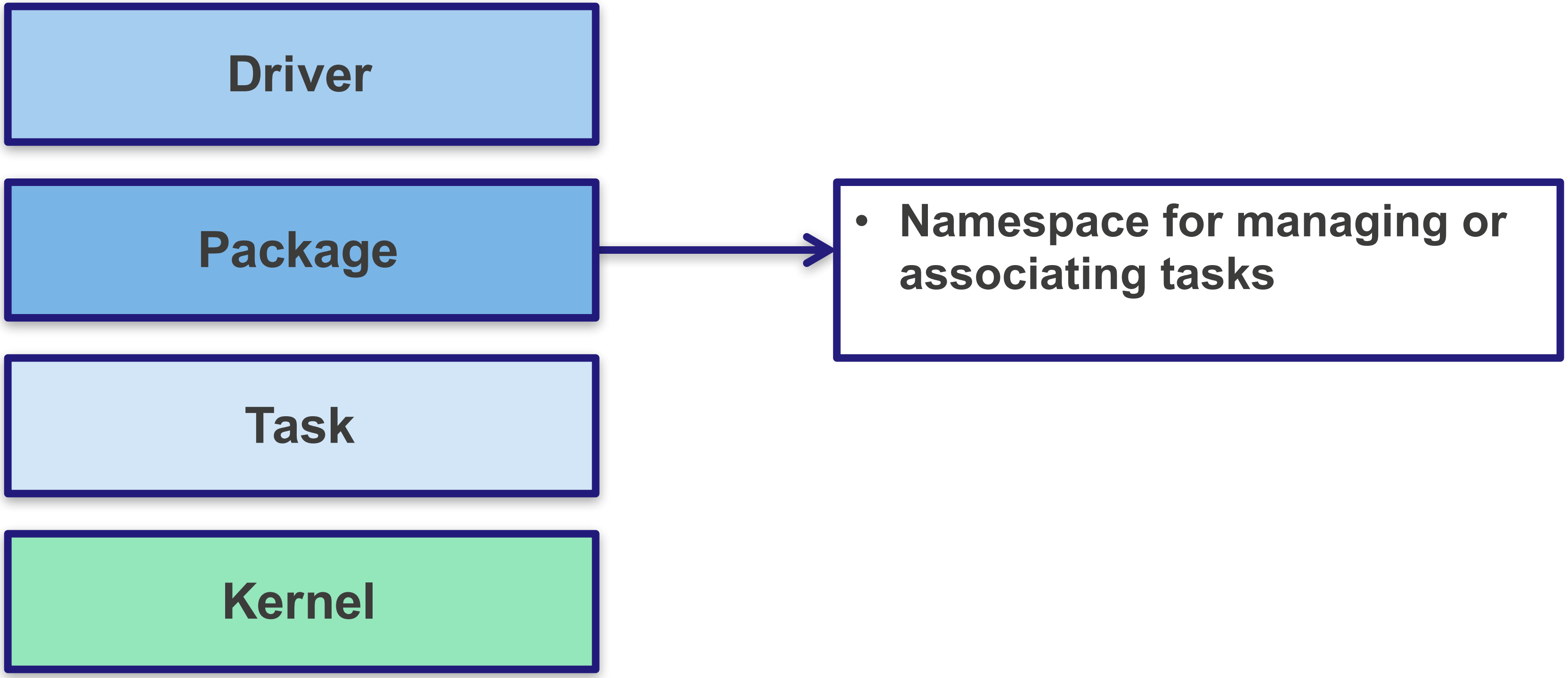
- Task-Based & Kernel-Based
- Relaxed Memory Consistency (between kernels)
- Sequential Memory Consistency (fine-grained)

How do we map this to our internal nomenclature?



- “main” function (serial/MPI)
- Top-level task (Legion)
- Time evolution control
- Associates packages

How do we map this to our internal nomenclature?



Driver

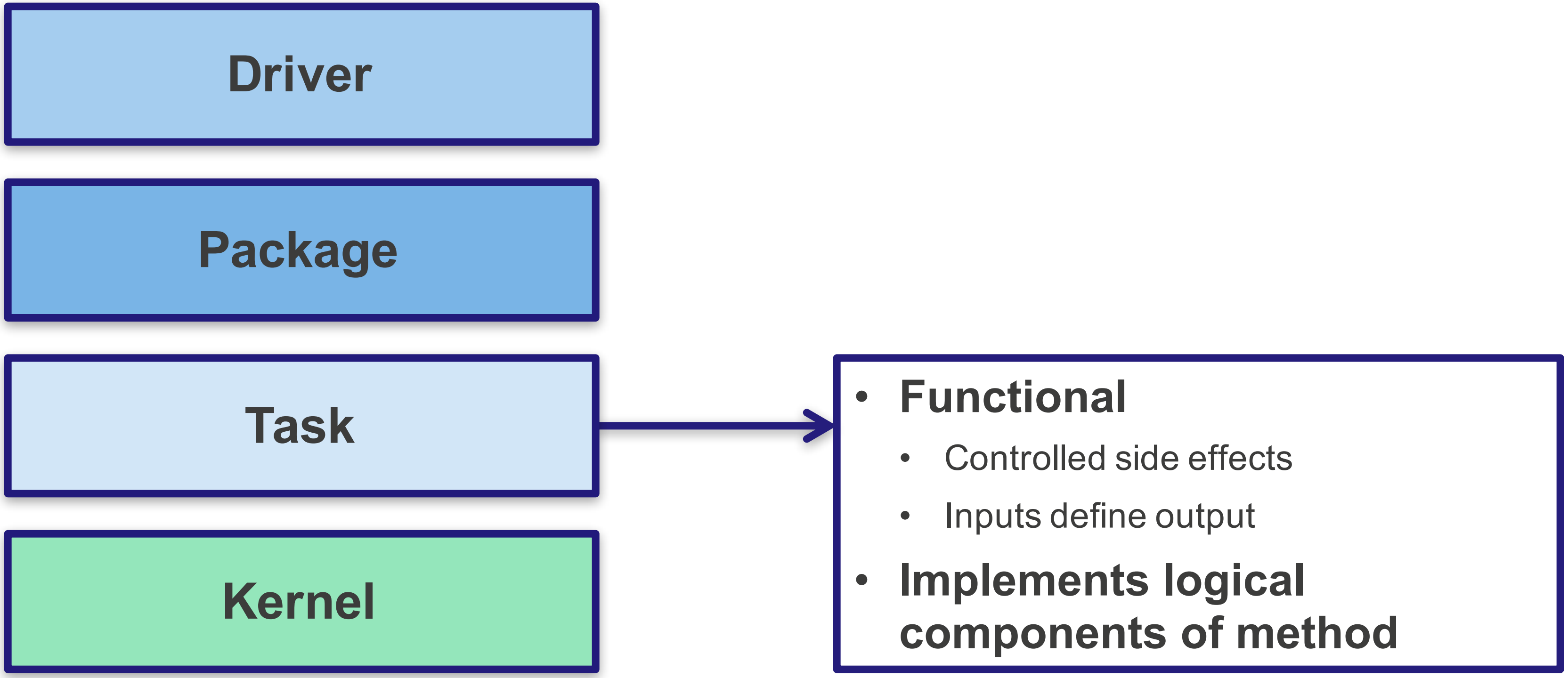
Package

Task

Kernel

- Namespace for managing or associating tasks

How do we map this to our internal nomenclature?



Driver

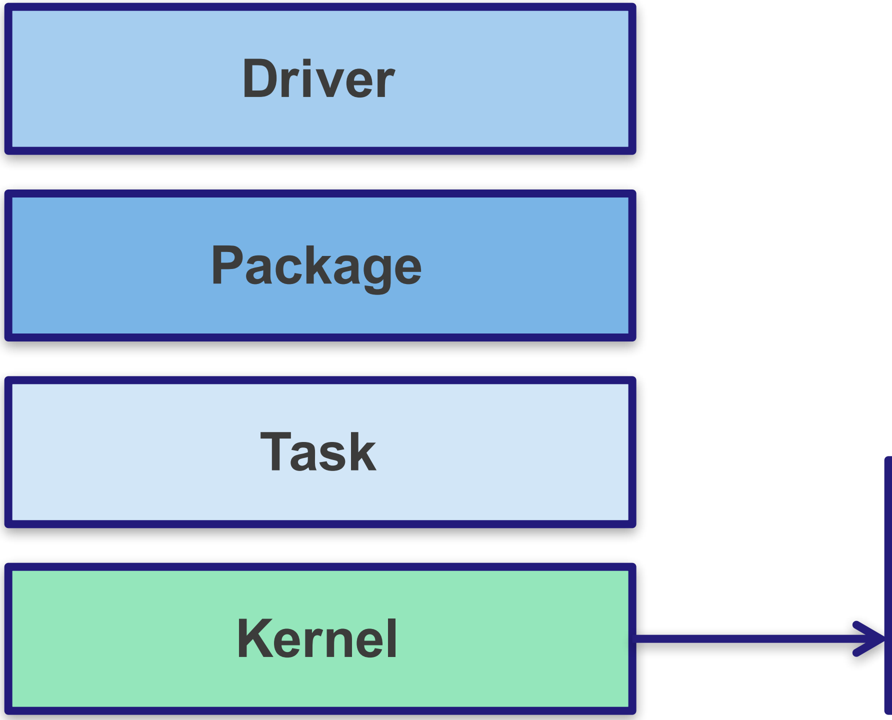
Package

Task

Kernel

- **Functional**
  - Controlled side effects
  - Inputs define output
- **Implements logical components of method**

How do we map this to our internal nomenclature?



- Fine-grained, data-parallel functions, e.g., *foreach*
- Use on-node runtimes: Kokkos, C++17

What are we doing?

- Legion or STAPL for distributed-memory (system-level & node-level)
- Kokkos or C++17 for shared-memory (node-level)
- Flexible Computational Science Infrastructure (FleCSI) abstraction layer to insulate us from uncertainty of runtimes
- We will consider new runtimes as they mature or displace current instances…