DUNE Overview

Michael Kirby/Andrew Norman

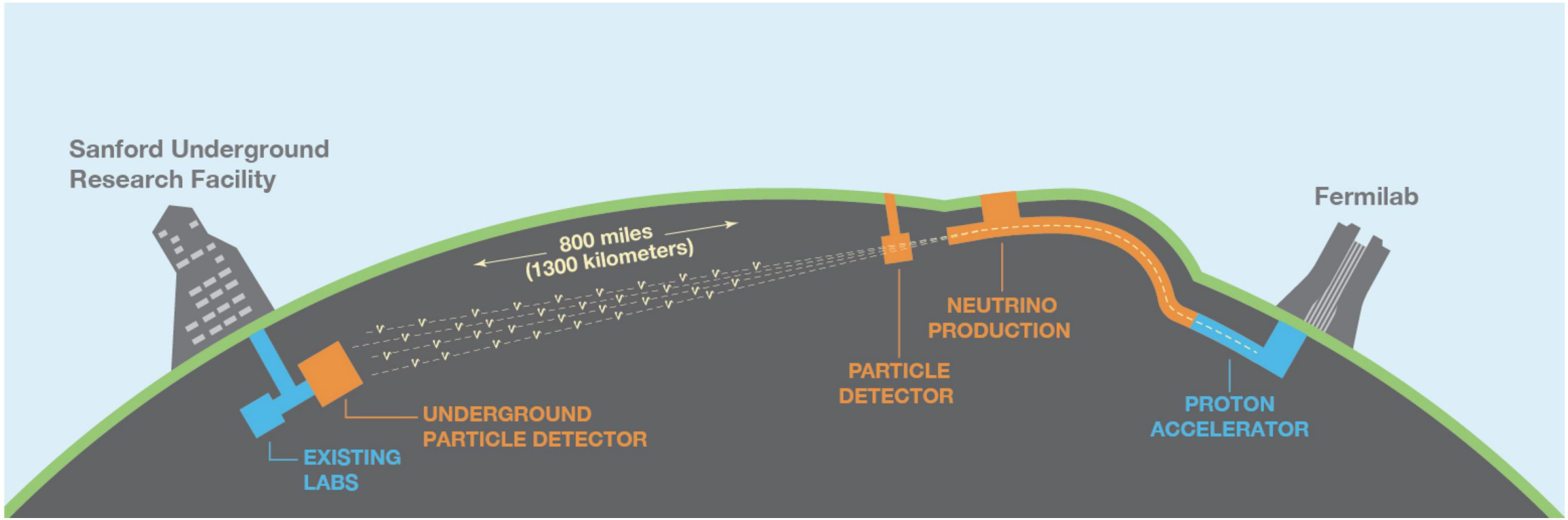
HSF/DUNE framework requirements mini-workshop

2 June 2021

DUNE Overview

- DUNE Experimental design
- DUNE Timeline
- Computing Consortium
- Framework Overview
- Framework Requirements

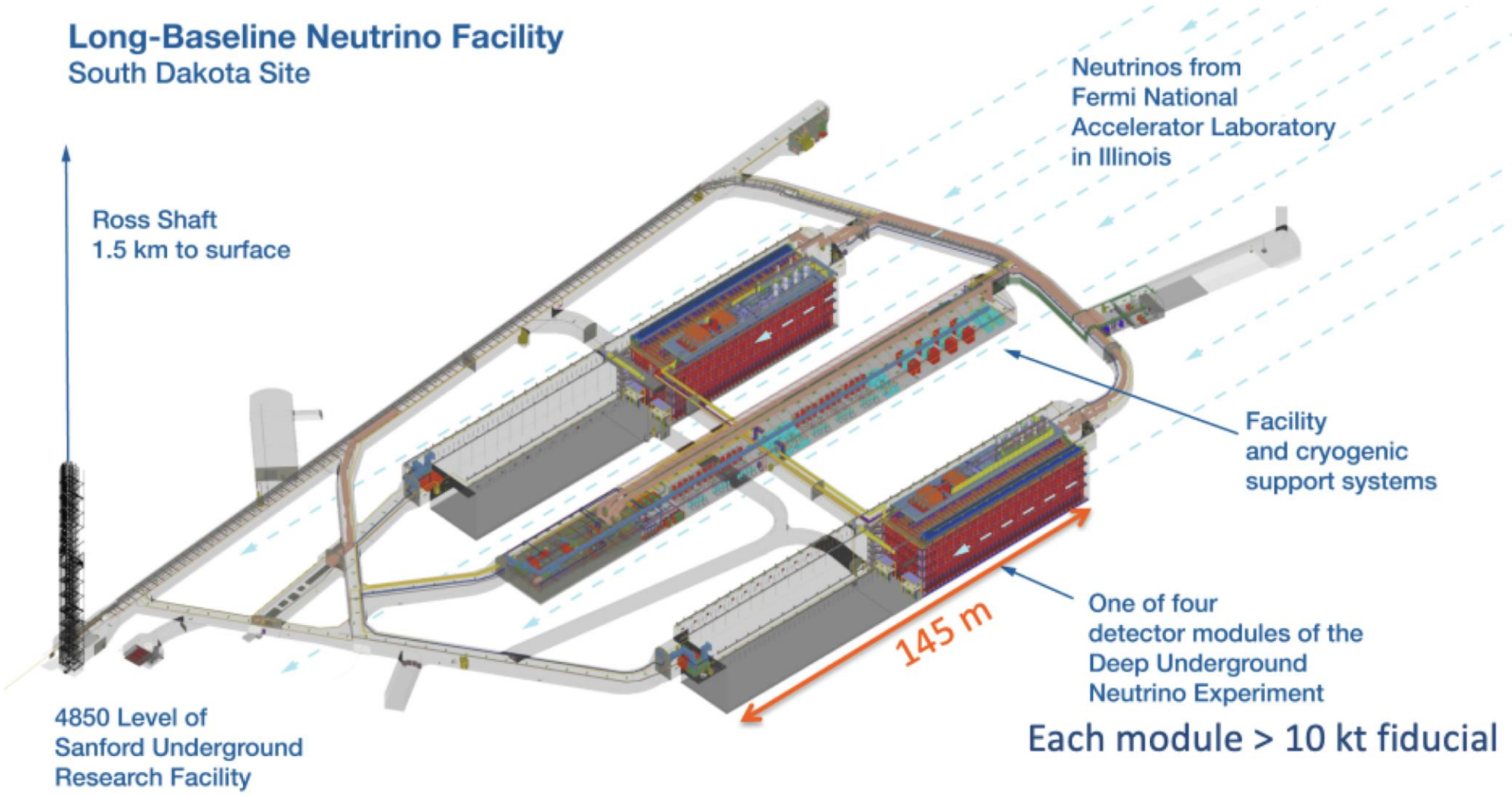
DUNE Experiment Overview



- experiment measuring neutrino oscillation parameters (mass ordering, matter vs antimatter asymmetry, unitarity)
- searching for proton decay, hoping to detect a local supernova neutrinos, and more
- 4 - 17 kT LAr TPC detectors at 4850 ft underground in Lead, SD (Homestake Mine)
- Near Detector on site at Fermilab near the neutrino production
- Two prototypes at CERN - (ProtoDUNE Horizontal Drift - ProtoDUNE Vertical Drift)

Fermilab

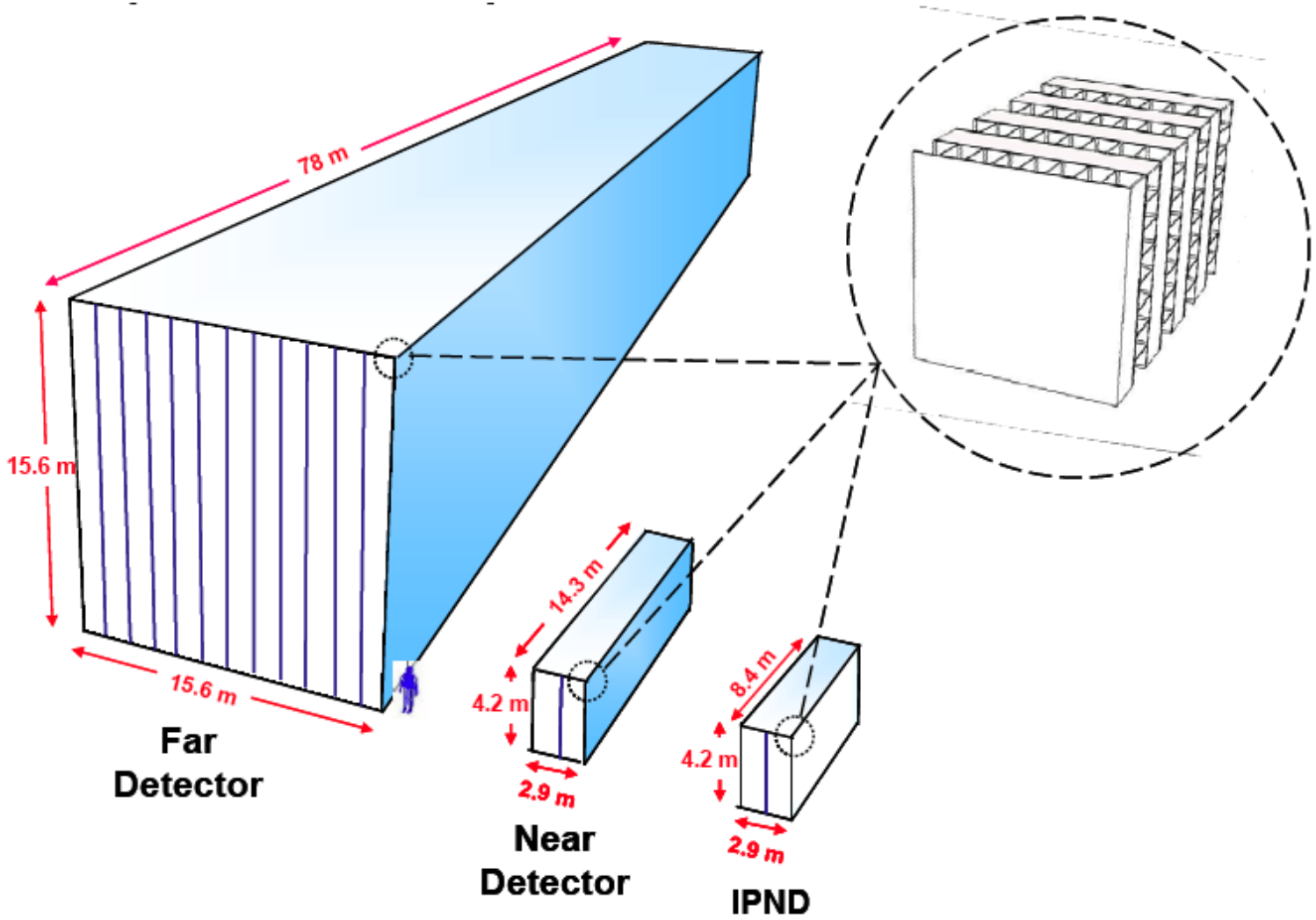
DUNE Far Detector



Fermilab

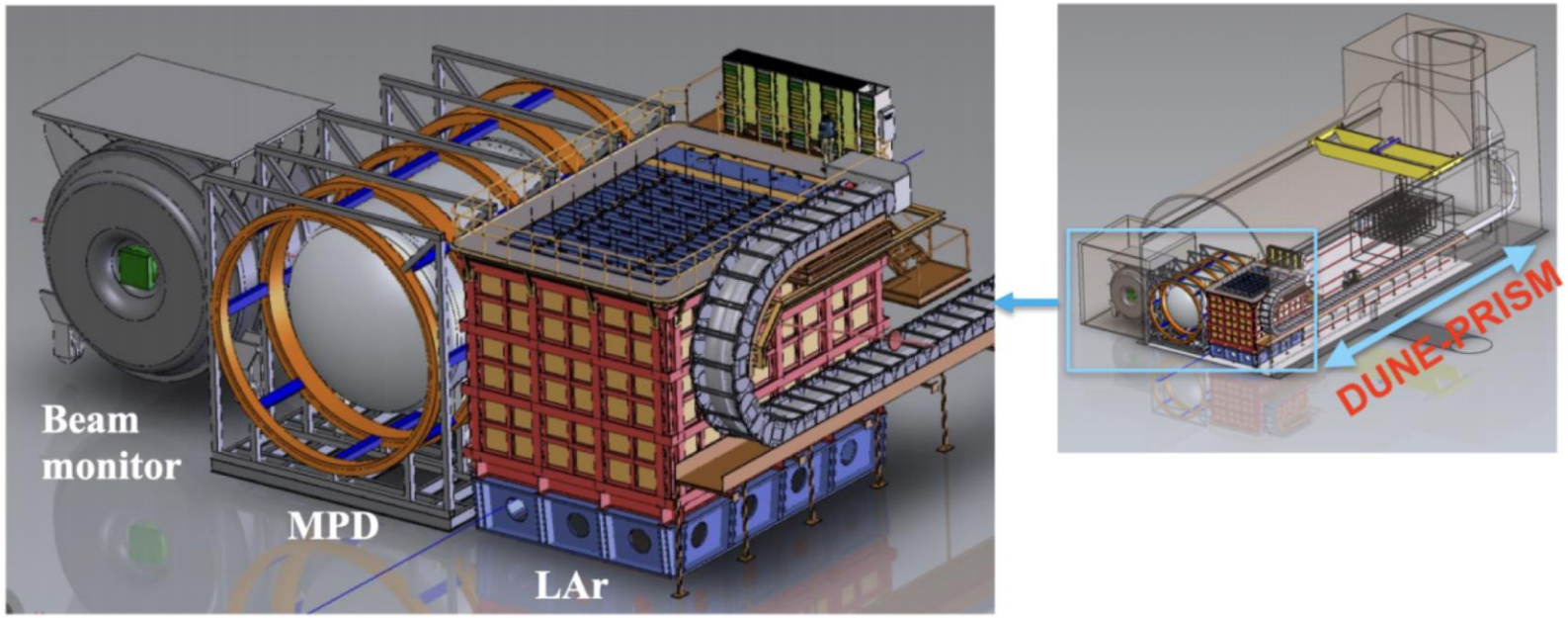
Previous long-baseline neutrino experiments at FNAL

- MINOS and NOvA both chose to minimize detector systematics with similar near and far detector designs
- They also made the reasonable decision to have their detectors stationary
- DUNE was more ambitious



Fermilab

DUNE Near Detector

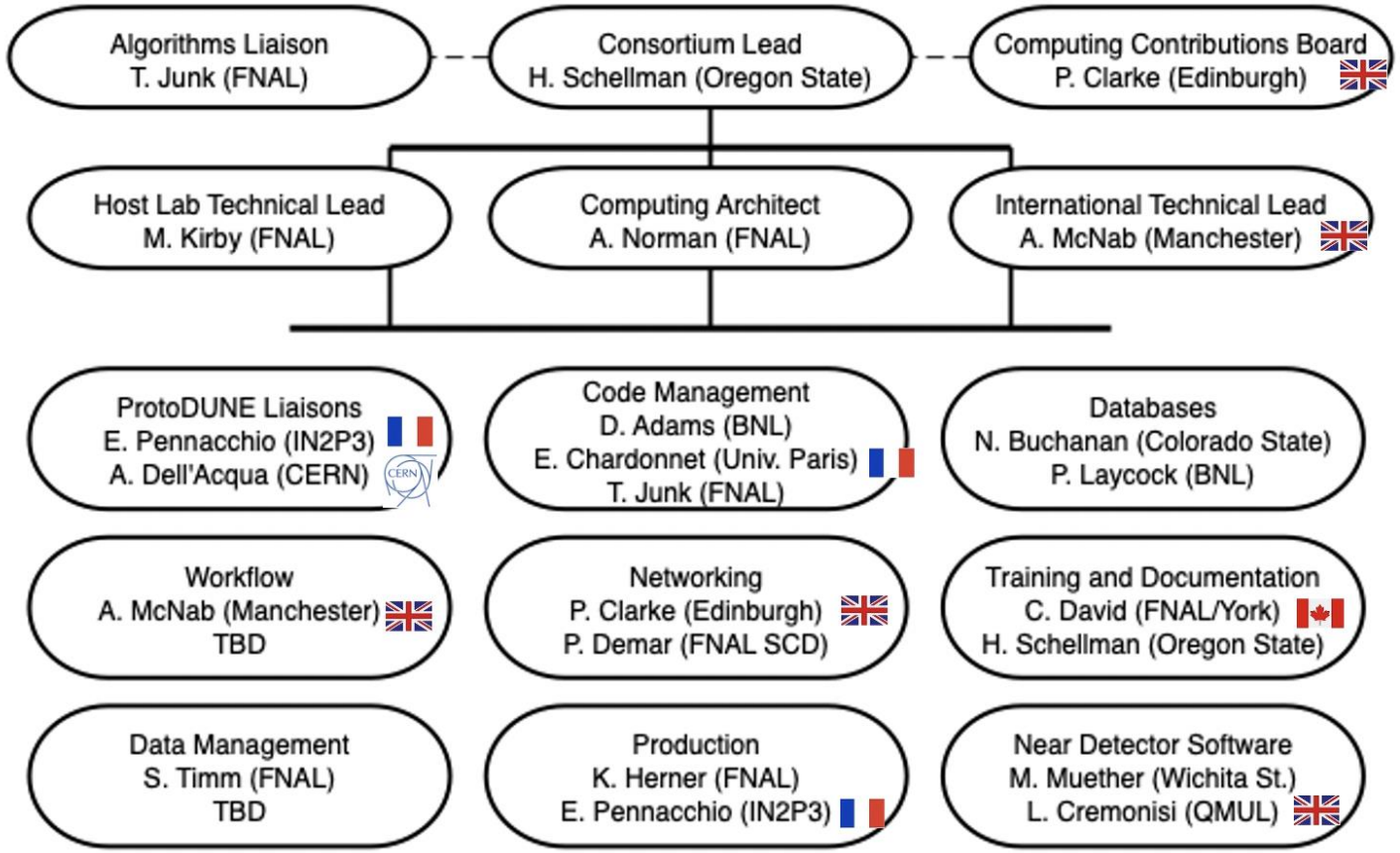


- LAr: Highly segmented LAr TPC (ArgonCube)
- MPD (Multi-purpose detector): High Pressure Gas Argon TPC, Calorimeter, and muon system magnetized by superconducting coils
- Beam monitor: High density plastic scintillator detector with tracking chambers and calorimetry in KLOE magnet
- DUNE-PRISM: Movement of LAr+MPD transverse to the beam, sampling different E_ν

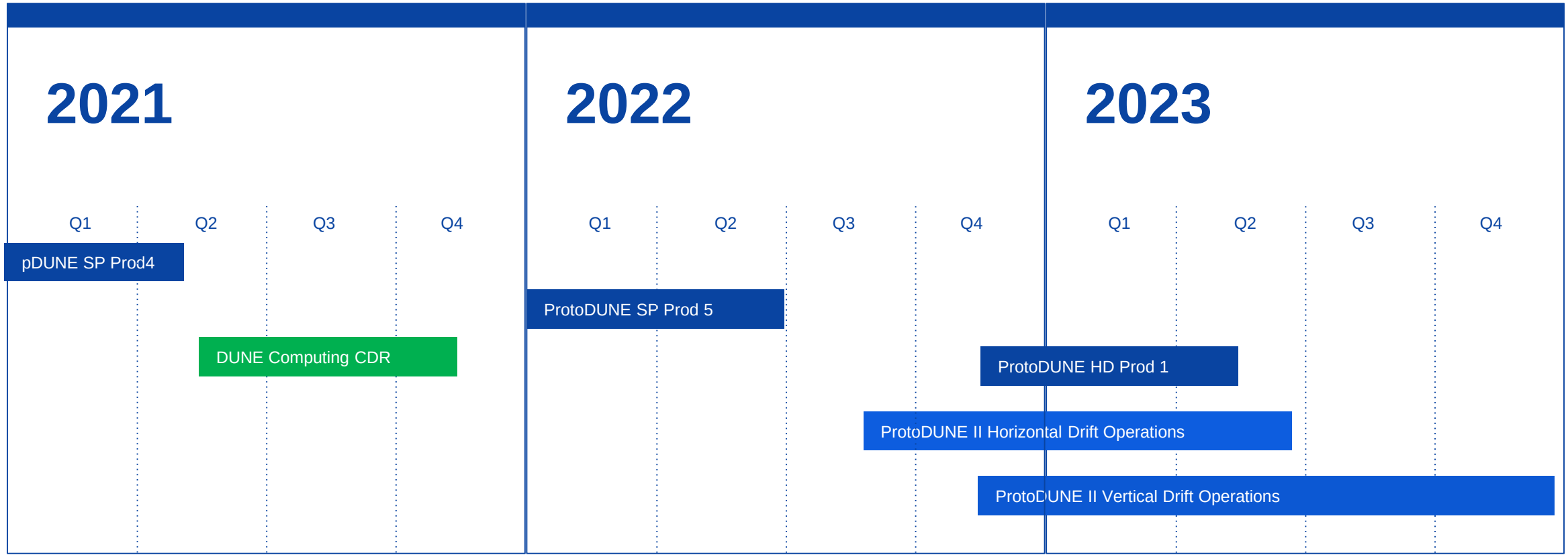
Fermilab

DUNE Organization Chart for Offline Computing

- Computing Consortium organization has been operating for close to two years now
- Significant progress in both organization and incorporating new partners into the consortium
- Limited amount of direct funding for projects, but this is expanding and exploring opportunities
- Well defined work packages for those funding applications are important going forward
- Expectation is the DUNE will contribute to research and development of solutions and services

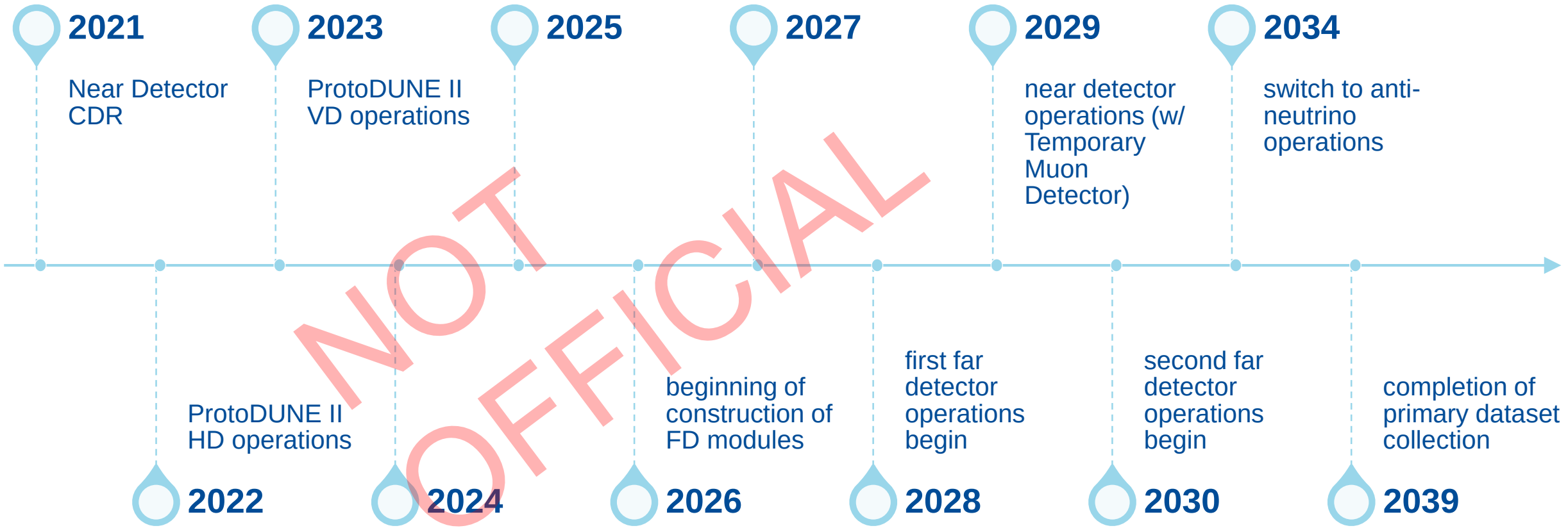


Short-term timeline for DUNE experiment



Fermilab

Long-term Timeline for DUNE Experiment



Only intended to give you an impression of timeline

Fermilab

The Need for a Framework

- The large data volume, complexity of analysis, and the number of contributing collaborators combine to make us worry about:
  - Consistency across analysis chains
  - Reproducibility of results
  - Bookkeeping and systematics accounting

- The problem is that the landscape that we need to handle these across has also gotten more complex:
  - We must use HPC centers in the U.S. (and these have different architectures)
  - We need to use accelerators to process our data efficiently
  - Our analysis methods are more complex and rely on external black boxes more

The Need for a Framework

- The only reasonable way of addressing this is to impose order on our software
- In particular we need to control:
  - Our I/O access, persistency, portability, and provenance of data
  - Our safety and reproducibility of results under differing levels of concurrency, architecture, and offloading to computation engines
  - Our algorithm consistency and efficiency across platforms and compute topologies
  - Our ability to perform both high throughput data processing and high speed analysis
  - Insulate us against user error pertaining to low level ”CS” issues
- A formal framework is our choice for doing this.
  - This worked well for NOvA and has converted a large segment of the neutrino community to the paradigm of using rigid formal frameworks.
  - We want to us DUNE as an opportunity to expand on this

What we want from HSF

- We want:
  - A candid analysis of how much overlap exists between DUNE and other HEP experiment's needs
  - An analysis of where our requirements have already been satisfied by existing frameworks/technologies (i.e. existence proofs)
  - An analysis of where our requirements deviate significantly from existing technologies
  - An estimate of the technical feasibility of satisfying specific requirements and generalized estimates of the associated effort
  - Recommendations on additional requirements that we may have overlooked

- Our intent is to derive a set of "work packages" that can be presented to funding agencies and collaborating institutions.

Why LArTPC Neutrino Analysis is Different from other HEP Domains

- LArTPC detectors have:
  - Extended time window readouts (order of a few ms)
  - There is a spatial/temporal degeneracy created by the drift
    - This means that signals from adjacent times and spatial coordinates need to be accounted for together (i.e. these are not prompt signals so we can't treat things as independent)
  - The readout data volume is large because we are effectively reading out waveforms along the wires to "voxelize" the volume.
    - This means that data volume scales with the physical extent of the detector and the fidelity of the voxels.
    - Zero suppression is also NOT able to be applied early in the analysis chain due to how signals are combined in 2-D deconvolution transforms to form the "hits" or regions of activity.
    - This means we have HUGE readouts which looks like a combination of waveform and image data

Why LArTPC Neutrino Analysis is Different from other HEP Domains

- Our analysis techniques look more like digital signal processing at some stages, image transforms at other stages, and generalized matrix transforms at others.
  - This means we often need to change the layout and organization of the data to apply the appropriate transforms (think of this as matrix transpose operations so we can apply FFT style manipulations and then back transforms and transpositions)
    - This means we tend to want to carry around intermediate data representations
      - These are similarly large and create problems with memory footprints
    - Our data is also highly compressible in some representations, but needs to be represented in full form during transforms (think of the sparse matrix problem)
- Our data and transform chains are HIGHLY amenable to GPU style acceleration
- We rely more and more on machine learning techniques over traditional pattern recognition and reconstruction techniques (more data representations)
- We are never able to fully drop our dependence on raw data.

Why LArTPC Neutrino Analysis is Different from other HEP Domains

- Neutrino analysis is tricky because we always have a unobserved initial state and have to rely heavily on simulation to understand the beam fluxes, nuclear interaction dynamics, neutrino cross sections, etc…
  - All of these carry with them significant systematic uncertainties
- The main neutrino oscillation analyses are driven by needing to control systematics under different modeling assumptions
  - This is a pain. In current 3-flavor analyses it is typical to have systematic variations of the data across $O(10^2)$ leading variables and a number of different major models.
  - Flux accounting is critical to making the measurements
    - We account for beam flux on a spill by spill basis and have to match our Monte Carlo to this.
- Final analysis techniques are complicated shape fits with brute-force techniques for getting the stats right.

Scope of what we want from a Framework

- DUNE wants a framework that can:

1. Handle primary data processing
2. Natively handle accounting (flux)
3. Natively handle systematic variation of data
4. Use targeted offloads to accelerators (GPUs)
5. Split event processing across high levels of concurrent compute
6. Use external “as-a-service” resources (i.e. ML inference as a service)
7. Aggregate data across events w/ proper accounting
8. Handle aggregated selection/analysis/fitting

Our requirements

- Our taskforce document lays out our needs as driven by our physics. This includes:
  - 3-Flavor oscillations and other oscillation analyses
  - Neutrino cross section measurements (near detector)
  - Supernova neutrino bursts and multi-messenger astro
  - Other mainstream DUNE analyses (see Physics TDR)

- We did not constrain ourselves to “what frameworks do today”
  - DUNE is NOT a legacy experiment and is not tied to a software stack
    - LArSoft however is a toolkit that we want to leverage as is Pandora, Wirecell, etc…
  - We have time before the first data arrives from the far detectors
    - We can use our early prototypes as testbeds for new software

Leading Requirements

The following are the requirements that we want to call out as highest priority to DUNE or ones that we worry about.

- Reproducibility (Req-4/5/6/7/9)(Req-14/15/16)
- Data representation (Req-2)(Req-21/22/23)
- Memory Management (Req-27/28/29)
- Concurrency and multiprocessing models (Req-10/13)(Req-17)
- Systematics and overlays (Req-32/37)
- Accounting (Req-41)
- I/O Layer and Serialization (Req-24/25)(Req-31)

Andrew's Top Three Worries

1. Memory in a distributed environment
   a) How do you keep it all coherent
   b) How do you fit in the footprint of the compute
   c) How do you keep from trouncing yourself when go off to your accelerator

2. Random numbers in a distributed environment
   a) How do you maintain reproducible random fields across different compute topologies
   b) How do you store the states and manage seeds

3. Exposure accounting w/ dynamic detector conditions

Questions?

I’ve tried to answer questions that were submitted before this workshop.

Please see the live doc for the responses.

Excerpts from Taskforce Executive Summary

- The following are excerpts from the executive summary that was presented to DUNE, it’s provided for reference.

Findings

- The Dune experiment requires a unified software framework that is capable of:
  - Initial processing of extremely **large readout data** (trigger frames) using discrete and matrix **transforms methods** with the capability to offload computations to **accelerator technologies** which maintaining a **coherent parallel execution** model.
  - **Subdivision** of **data and computations** across **multi-node** computing architectures while maintaining **a coherent data model** and **parallel execution model**.
  - **Partial read/write** of trigger record data under a coherent **parallel data** and **execution model**
  - Execution and tracking of processing cycle under different primary data atoms (i.e. looping over “slices” instead of “trigger records”)

Findings

- …capable of:
  - Accounting and aggregation facilities discretized at the primary data atom level
  - Providing modular I/O under multiple formats and protocols
  - Providing automatic management of serialization and compression of data objects for both persistent and transient data representations
  - Merging of multiple data sources (i.e. overlay patterns)
  - Joining of multiple data sources (i.e. friend referencing)
  - Simultaneous analysis of data from multiple detectors
  - Coherent systematic variation of simulation, data, hybrid overlays
  - Providing full provenance for data under both algorithmic and systematic variation

Findings

- …capable of:
  - Managing the the generation, accounting and propagation of random number fields in a manner that is fully reproducible under parallel execution.
  - Managing and providing modular access to external tools for machine learning and other algorithmic techniques in a manner that retains the coherence of the data and memory models.
  - Providing data representation reduction (“skimming/slimming”) while retaining full data product provenance
  - Providing bi-directional data product and event record linkage and association in both persistent and in core representations
  - Providing read and write access to non-POSIX storage and non-block storage devices for both persistable and in core data representations

Evaluation

However:

- The need for a coherent data and memory model under parallel and distributed execution differs significantly from current frameworks
- The granularity of accounting and shifts in “context” that are require are radically different from current methods
- The management of memory and accelerator access may be incompatible with current scheduling techniques