Wire-Cell Toolkit ↔ Phlex Framework

Brief update on initial integration work

Brett Viren

May 14, 2026

Topics

- Phlex history and status
- WCT / Phlex comparisons
- WCT / Phlex integration design
- Status and near future
- Strategy thoughts

Some selected Phlex history

- FNAL decided to replace *art* with an entirely new framework for DUNE.
- FNAL R&D funded Kyle Knoepfel to develop experimental **meld** framework.
- FNAL/DUNE is now funding development of **Phlex** based on lessons from meld R&D.
- **Phlex** latest release 0.2.0 made on March 31, active development.

Phlex status today

High-quality package, still (intentionally) limited-scope

- Easy, portable build story, Spack packaging, documentation.
- Trivial example algorithm / application code at level of: “add two numbers”.

Status of Phlex ecosystem

~~Phlex provides file I/O, Geant4, geometry, data model and everything you need!~~

There are essentially **no Phlex applications yet**.

> *To make an apple pie, you must first invent the universe. –Carl Sagan.*

Need to invent things just to give a working context for the WCT/Phlex integration.

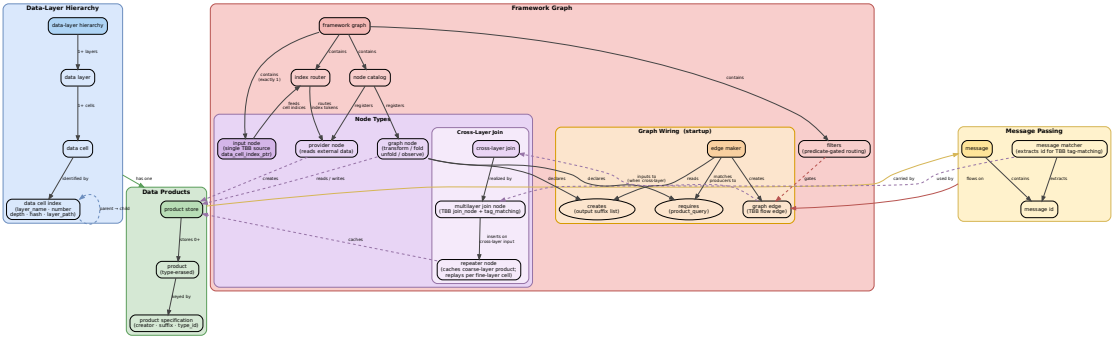
- But, in this **green field** lies potential opportunity for **first-movers**....

Wire-Cell Toolkit / Phlex comparisons

Wire-Cell Toolkit DFP design



Phlex DFP design



Wire-Cell Toolkit / Phlex differences

- **WCT DFP design** is much simpler but (ironically) more flexible than **Phlex**
  - Some WCT subgraphs are (now) impossible for Phlex (hydra, async fans)
- **WCT data** lives on graph edges, **Phlex** data lives in a central store.
  - **Phlex** is more complex but does better **data provenance** (who made what data).
- **WCT design** uses dynamic abstract interface, **Phlex** uses hard-wired C++ static typed.
- **WCT services** are retrieved by nodes, **Phlex** passes **resources** just like input data.
- **WCT nodes** are C++ only, **Phlex** also supports **Python** nodes!

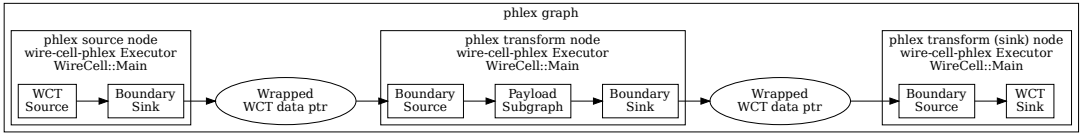
The differences pose no major problems to WCT/Phlex integration.

- Some minor issues to deal with, but also some things are a breeze.
- The WCT/LArSoft integration was both harder and easier.

Wire-Cell Toolkit / Phlex commonalities

- Both adopt the “data flow programming” (DFP) paradigm.
  - A computational **graph** where **nodes** execute by exchanging data along **edges**.
  - Intel oneTBB `flow_graph` underlies the graph execution of both.
  - WCT+Phlex can share a thread pool (a **big win** for MT running).
    - ★ Torch is still a thorn.
- Both use Jsonnet as the primary configuration language.
  - Both can also accept JSON. Phlex also accepts YAML and yes even FHiCL.

Wire-Cell Toolkit / Phlex integration design



Executor embed and execute **WCT-graph-as-phlex-node**.

Boundary converters wrap/unwrap WCT data to/from Phlex-compatible types.

Somewhat similar in concept as `larwirecell` integration package.

Status of `wire-cell-phlex`

Exists[1] and is fully functional for WCT sim and sigproc jobs.

- Includes Phlex+WCT configuration derived from `dunereco` for PDHD and PDVD.
  - Factored into **detector-generic job config** and **common-shaped detector-specific objects**.
- Exposes native WCT data objects to Phlex graph as first-class data.
  - Thin wrappers and converters.
- Good unit test and integration test coverage.
- Documentation: README and other markdown files.
- For now, lives at `https://github.com/brettviren/wire-cell-phlex`

[1] This work took three days and about 200M tokens.

Known todo items

- Add `wire-cell-phlex` packaging to `wire-cell-spack`.
- Internal testing by WC core folks to try to break things.
- Extend to support WCT imaging, clustering, pattern recognition.
- Extend the detector-factored config to FDVD/FDHD variants.
- Plan “public relations” for how to introduce `wire-cell-phlex` to broader DUNE.

Known unknown todo items

Current use of WCT in Phlex is contrived just to give some job context.

- No experience yet in how to “play well with others”.

DUNE has no data model (“object lingua franca for nodes to speak”)

- I have a proto-design proposal that I will give next week at the collab meeting.

Expect need for some “fat” converters when the other major players join:

- I want to push development of these on to others to the extent I can.

Comments on future and strategy for spending our effort

Green field first-mover advantage

Get `wire-cell-phlex` accepted as "real" ASAP to force its de-facto standardization.

- Want WCT data objects to be considered first-class citizens in a Phlex graph.
- Let the late comers write their own data converters to consume WCT objects.

The new, fast development paradigm → broader vistas?

Given the (relative) ease of getting this far, it makes me set my sights higher.

- I guess we can integrate Geant4+Phlex, eg with `edep-sim-phlex`, in ≈ 1 week.

But, if we do this, do we get any real profits or just acquire more support debt?