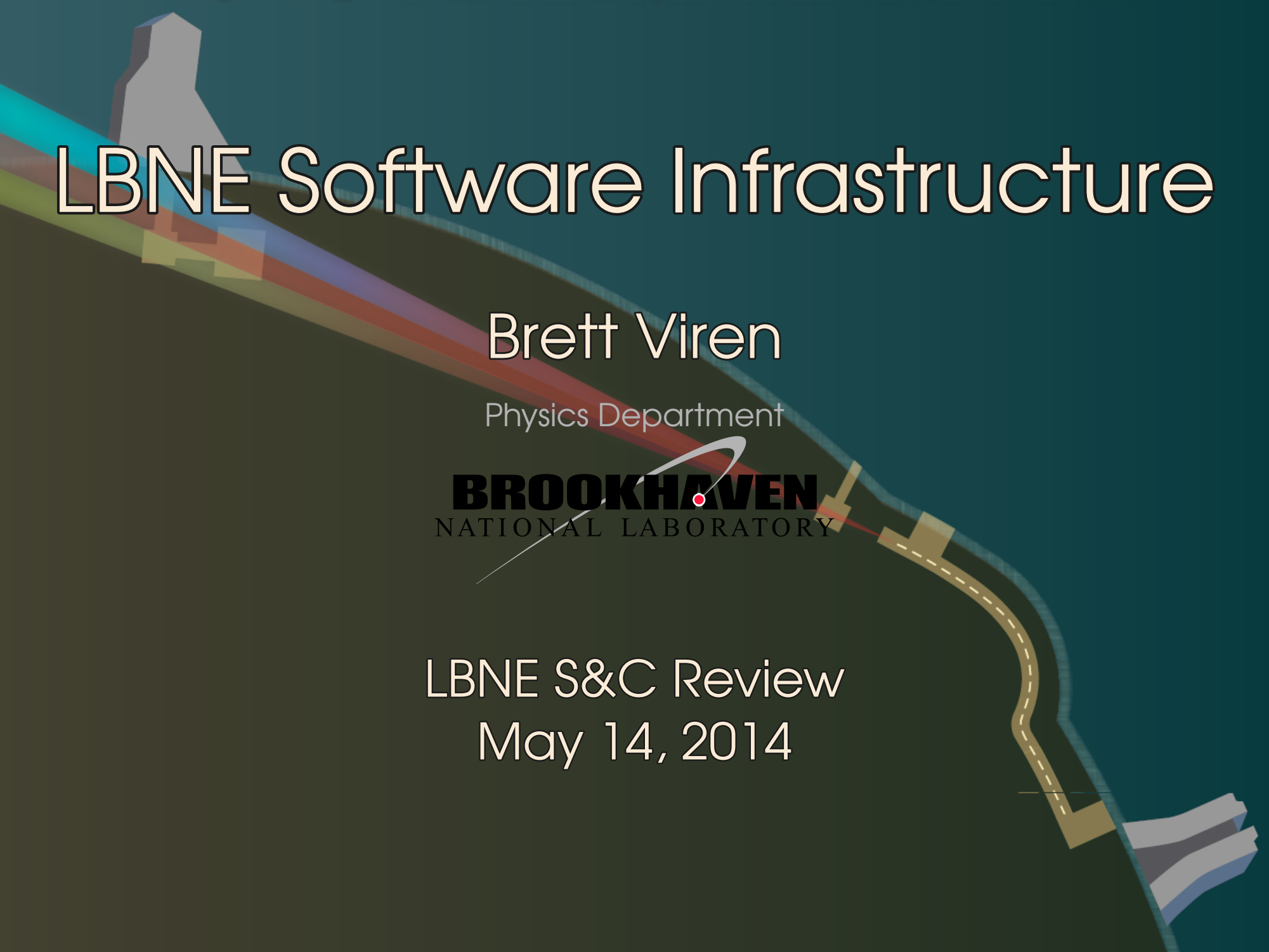
This is a draft.

If you see

things like this

they are todo notes.



LBNE Software Infrastructure

Brett Viren

Physics Department

BROOKHAVEN NATIONAL LABORATORY

LBNE S&C Review

May 14, 2014

Outline

Introduction

Software Infrastructure Topics
- Revision Control
- Software Building
- Release Management
- Software Distribution
- Continuous Integration

Summary

LBNE Software Landscape - Suites

LBNE currently has these partly independent software suites

- **lar** far detector simulation and reconstruction based on *art*/LArSoft.
- **g4lbne** neutrino flux simulation using Geant4/ROOT application.
- **fastmc** parameterized simulation and analysis using GENIE and GLoBES used for both ND and FD studies.

LBNE Software Landscape - Fermilab

- Many packages in common with other Fermilab IF experiments (eg, *art*/LArSoft)
- Leverage expertise and effort in Fermilab Sci. Computing Division (SCD)
  - optimization of effort over many experiments leads to competition for SCD attention
- Current reality is an entrenched, Fermliab-centric computing model which does not satisfy LBNE requirements:
  - Must be easy to build LBNE software from source at all institutions.
  - Must support prevailing collaborator platforms (not just Sci. Linux).
  - Self-determination of versions of packages used.
  - Long term support and retaining of releases.
  - Understood code provenance.
- Some small progress in reorienting Fermilab to LBNE requirements.

Software Infrastructure Topics

The **Software Infrastructure** (SI) group is a division of the LBNE collaboration **Software and Computing Organization** responsible for:

revision control assure that the software that LBNE develops is tracked in a version control system

software building assure all collaborators can easily build and install LBNE software on their computers

release management defining releases of LBNE software suites and mechanisms for producing them

software distribution provide methods for distributing ready-to-use software to institutions and the Grid

continuous integration build and validate LBNE software continuously and promptly after any changes are made

Revision Control

LBNE developed software source code is required to be in a Version Control System (VCS)

- All major applications and framework modules already covered
  - **git** as become the de-facto standard and
  - some major packages were migrated away from CVS/SVN
- Developers of "personal" software encouraged to use VCS
  - Require all LBNE source code for official results in a VCS
- Fermilab provides VCS (git/CVS/SVN) via **Redmine**
  - Most LBNE software projects make use of this service
  - Some experimental, or non-LBNE-specific software kept elsewhere (eg, GitHub)

More details on Redmine in E. Berman's talk in the afternoon

Software Use and Development Model

- Support both per-site and per-user installation.
- Automated installation from source or binary.
  - Special deployment mechanisms where needed (eg, Grid/CVMFS)
- Versioned "base" releases of multiple software *suites*:
  - suites defined by their LBNE packages and requirements
  - capture all build options and aggregate of package versions
  - allow, but not require, inter-suite package overlap/reuse, trade off between sharing and bloat (eg, *art* needs non-OS GCC)
  - a'priori lax requirements on supported OS (Unix-like)
- Retain build/installation dependency information for user environment setup.
- Developers leverage an installed base release
  - Development package builds override base release packages via appropriate user environment and build settings.

Base Release Software Building - worch

The **worch** tool is used for building LBNE software suites.

- High degree of automation of download, configuring, building, installing and post-installation user environment configuration.
- General purpose, detailed control over installation layout and support for multiple environment management systems (eg, Environment Modules, UPS)
- Exercises package's native build system (autoconf, CMake, SCD scripts)
- Captures entire software suite's installation and version information in a concise text configuration
  - enables simple release management (more below)
- Extensible to new packages and patterns of use
  - currently focused on building from source
- Was developed for LBNE, but general purpose and in used outside LBNE.

Current Build Status - art/LArSoft

Installation from Fermilab-built binaries relatively simple.

Significant trouble building *art*/LArSoft from source

- underlying build system is CMake but entwined with Fermilab software infrastructure (UPS),
  - currently not able to build without whole-cloth replication
- layers of build scripts from SCD are somewhat improved in recent months, still only semi-automated and add unwanted complexity
- Fermilab releases not tested in a green field
  - many release-bugs left for installer to discover (see CI section below)
- Only Sci. Linux well supported by SCD scripts
- **worch**-based automation on top of SCD scripts developed
  - \+ used for one successful build of LArSoft at BNL's RACF (on SL6)
  - \- not maintainable, SCD scripts are to much of a moving target

UPS-free Build of *art*-related Packages

An experimental, UPS-free CMake-based native installation method for *art*-based packages is in development.

- work is by LBNE in collaboration between BNL and Warwick University with consultation with *art* group
- so far, just the first *art* package upgraded
- vastly reduces the building and installation effort and complication
- will give LBNE control over detailed package versioning
- lowers the bar to use *art*/LArSoft at collaborator institutions and new platforms

Current Build Status - other suites

g4lbne

- migrated from CVS to git
- requires Geant4, ROOT and their dependencies
- new, native CMake build system developed (not UPS-encumbered)
- **worch**-based build automation
- user environment managed via Environment Modules

fastmc

- requires ROOT, GENIE, GLoBES and their dependencies
- release and installation not currently managed
- initial **worch**-based build in development

Release Management

Leverage **worch** for release management:

- Each suite has a worch configuration file set
- Maintain file set in VCS (Redmine/git)
- Release starts with a branch of this repository
- Suite release includes specifying both a code version and a build version string
- Simple script to automate build ("just download and run")

Status

LArSoft initial release using this methodology on top of SCD scripts.

g4lbne partially implemented, but still builds "HEAD"

fastmc not yet

Software Distribution - Grid

See M. Kirby's talk on Grid and Distributed Processing.

Software Distribution - Local

Current status

- **worch**-based, locally-built suites
  - UPS-based for *art*/LArSoft, last-step binary install
  - EM-based for g4lbne
- Pure-binary, UPS-based install via manual download and unpack

Future plans

- Automate binary install via **worch** for both UPS-encumbered and UPS-free suites.

Continuous Integration

CI = automatic and continuous rebuilding and validating code as it changes in order to catch problems ASAP.

- Need for a CI system recognized by NOvA and later requests from LBNE led to SCD looking for a more general solution.
- Set of requirements developed with major contributions from LBNE
- CI group formed with members from SCD and LBNE, leveraging prior experience in CI in Daya Bay and CMS.
  - The group refined the requirements, identified candidate technology, evaluated test installations against requirements
- Status: selected technology (Jenkins) and out for wider comment from experiments and software projects.

This CI group serves as an excellent model for how LBNE should make other significant technology decisions.

Continuous Integration - cont

Some requirements:

- For use by Fermilab-associated experiments and software projects
- CI clients around the world and on different platforms.
  - Fermilab will supply and manage some CI client hardware for experiment use.
  - LBNE plans to run CI clients on all major platforms at multiple institutions.
- Incremental and green-field building and validation.
- Various triggering: nightly, by commit, by tag, ad-hoc.

Some knock-on benefits

- Promote successful builds:
  - nightly snapshots
  - release builds
- Form CI group going forward to share strategy and consolidate testing of common software to avoid duplication

Summary

- LBNE software infrastructure effort is underway.
- Near term focus on getting LBNE software into collaborators hands.
- Some build issues are being worked on in LBNE and with Fermilab SCD.
- A continuous integration service has been identified and will soon be deployed after very well run evaluation working group effort.