Fermilab

Managed by Fermi Research Alliance, LLC for the U.S. Department of Energy Office of Science

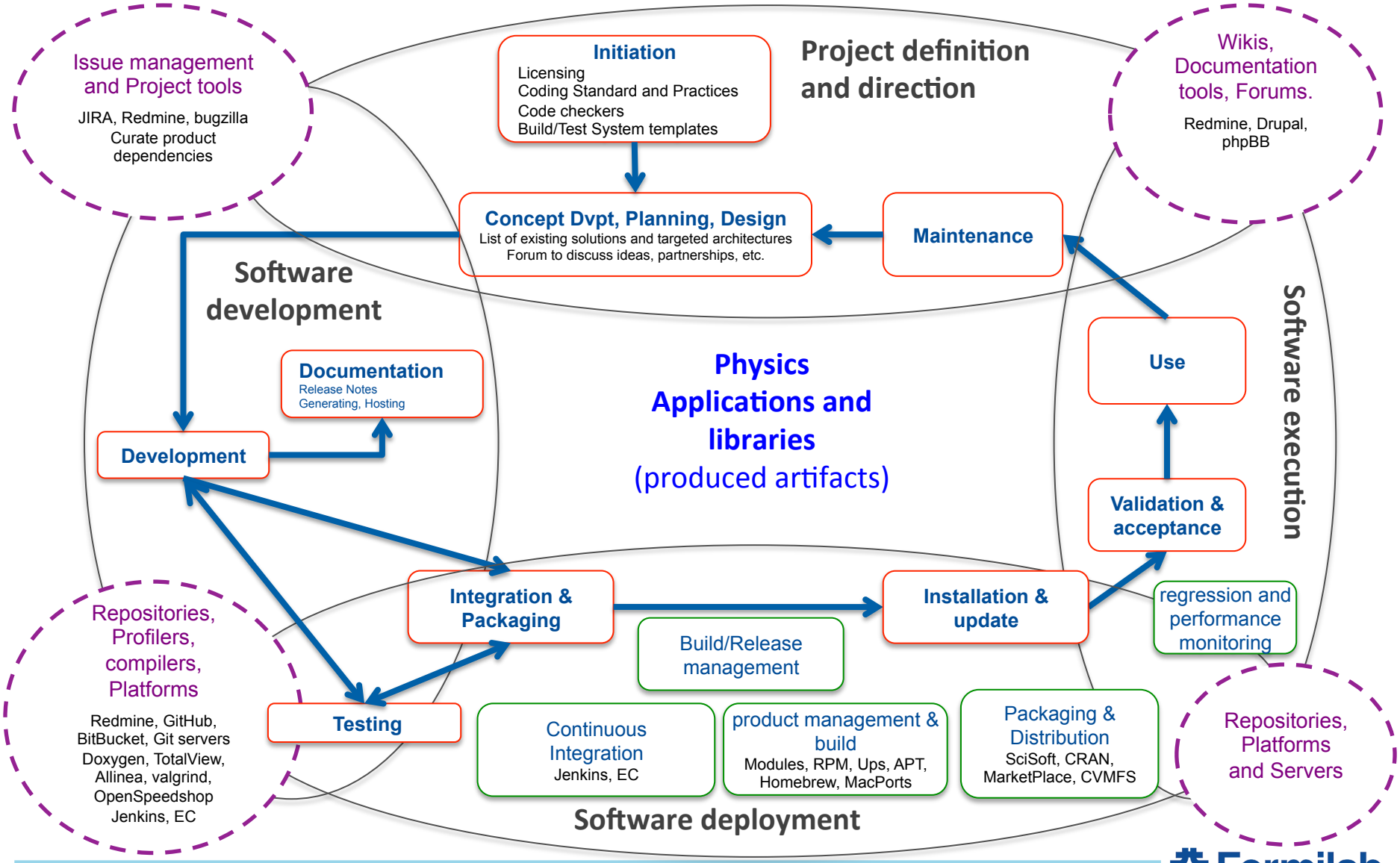
Fermilab and the HSF

Philippe Canal

HEP Software Foundation Workshop

20 January 2015

Project Life Cycles and Collaboration opportunities



Fermilab

HEP Software/projects

- Wide involvement in many areas of HEP software, for example:
  - ***ROOT***
  - ***Geant***
  - Frameworks
    - ***CMSSW, LArSoft , art, artdaq***
  - Generators
    - ***GENIE, Pythia, HepMC***
  - Storage and communication
    - ***Frontier, dCache, XRootD***

Common Services @ FNAL

- Build & release services
  - Distribution site with model for accepting and indexing contribution
  - Continuous build integration
  - Product management tools and guidelines, and instructions for use
- Aid in development and maintenance of best practices and standards suggestions
  - Shared repository use
  - Interface design, Coding
- Support Simulation validation.
- Open source and licensed products
  - ***Jenkins, Redmine, git***
  - ***Totalview, Allinea*** (profiler and debugger)

Build & Release Services

- Product management tools and guidelines, and instructions for use

- Distribution site with model for accepting and indexing contribution
  - ***SciSoft*** Distribution server (tarball download)
  - Management of product distribution manifests/dependencies
    - Download only the missing packages/versions
  - Product pull, install: scripts, ***ups, CVMFS***
  - Build and release requests are done on-demand

- Release and Packaging Services
  - Consistent builds across all dependent packages
  - Products rebuilt only if they are (directly or indirectly) changed
  - Used for Fermi products (and all their dependencies)
    - ***art, LArSoft, artdaq_core*** (a subset of artdaq), ***nutools***

Integration & Validation Services

- Continuous build integration
  - Workflow for building set of related products, testing, packaging and posting of distributions
  - Consistent builds across all dependent packages
  - Guidelines and procedures for adding products
  - Facilities that permit external extensions (***Jenkins***)
  - Main build platforms: ***SLF5, SLF6, OSX***
  - Users (at various degrees)
    - ***art, LArSoft, artdaq_core*** (a subset of ***artdaq***), and ***nutools***
    - ***Mu2e***, ***LBNE***
    - ***Minerva, GENIE, artdaq, DarkSide***

- Validation
  - Monitoring of ***Geant4*** run-time, memory, and physics performance
  - Web server to host and inspect collected data
  - Requires somewhat dedicated hardware for reliability of results
  - Being extended to also cover ***GENIE***

Conclusion

- Many on-going HEP wide efforts/collaborations
- Existing common services
  - Build, Continuous integration & release services
  - Aid in development and maintenance of best practices and standards
  - Support Simulation validation
- Those services could be contributed to ***HSF***
  - either directly or through the ***US HEP Forum for Computational Excellence***
- What would improve those services (non exhaustive)
  - Extend platforms supported by the Continuous Integration facilities:
    - Need for additional nodes and extra variety of nodes.
  - Sharing of licenses and access to (specific) hardware (for development). For example:
    - ***Intel*** compiler/profiling tools licenses
    - Various version of processors, co-processors cards, HPC computing nodes
  - Finer grain distribution
    - For plugin and user contribution. See ***CRAN*** for example (or ***BOOT***).
  - More source code sharing (as opposed to copying) in fundamental tools
    - Requires curating, trust, communication, ease of download/connection

- Backup slides and helpers

Definitions

- Release management: This area is primarily concerned with providing the release management and product packaging and building services and tools for groups developing and using software infrastructure. The services include packaging for external products.

- Continuous integration: This area focuses on access and use to the Jenkins build services and associated database, testing interfaces, and web summary facilities that exist. This includes assistance with integrating software product builds and unit / integration test suites with Jenkins, and supporting the facilities and software infrastructure to permit the running of a product build within this context. This includes building development and integration candidate releases for products on-demand, triggered, or periodically. It is recognized that this system relies on the build system that is part of the software package.

- Product distribution: This area focuses on product distribution through central web and file system services. It organized and allows users to access versions of full binary and source distributions for experiments across platforms and build configurations.

- Validation control (future): This area focuses on testing and quality assurance. Of particular interest is the control and coordination of performance and validation runs. This is a newly proposed area. Performance evaluation for both checking accuracy and speed both fit into the testing area and can be triggered by the re-lease building (including development and integration builds). Testing of LArSoft would likely be a first candidate here. Future candidates for this area are the Geant4 performance runs and the GENIE validation runs. Both of these systems have high demands that will almost certainly exceed the capacity and scope of the current tools and resources, requires interactions with GRID or other batch-oriented computational resources.

Under consideration

- Documentation generation
  - It is reasonable to consider adding documentation generation services to the areas in the future. The purpose of this is to add, just like testing stages in a build, automatic documentation generation for software products that are using the continuous integration system. Two examples of products that could be included here for support and configuration are doxygen and LXR covering documentation embedded in source files and source code navigation.

Release Management (Common build, packaging & release management)	**Continuous Integration** (Common facilities for build and testing)	**Product Distribution** (Common distribution site)	**Validation Control** (Common tools for launching and summarizing validation runs)	
	• Workers • Servers • Jenkins • Web tools • Database	• CVMFS • Web Site • Storage	• Future servers • Storage • Workflow tools	**Facilities** (computation resources and software packages)
• Experiment production release control • SCD production release control • External product packaging • Product dependency management • Tools configuration	• Experiment Integration • Tools configuration • Interface support	• Filesystem maintenance • Tools support • Web index management • Distribution manifests	• Future monitoring and configuration • Workflow and GRID integrate	**Services** (configure and utilize facilities, support software)
• cmake external • MRB • UPS packaging • Production build tools • Distribution build tools	• CI test interface • Database • web summary	• Production distribution tools • Web management tools	• Future interfaces • Coupling tools	**Projects** (development of the software tools owned and maintained within BCIR)

While reviewing the integration, build, and release functions that SCD is currently performing within various projects and experiments, an obvious breakdown into areas was apparent. An area represents a distinct line of work with specific goals and objectives. Each area is further broken down into the aspects of facilities, services, and projects.

Release Management	Continuous Integration	Product Distribution	Validation Control	
• Tag release • Configure ACLs • Initiate build • Update Jenkins configuration • Complete release • Initiate packaging • Package external • Accept external • Configure build	• Integrate test & build summary • Integrate build • Initiate build • Check state • Check results • Check test summary • Adjust authority	• Configure ACLs • Retrieve bundle • Post packages • Update index • Build manifest • Define bundle	• Initiate validation • Check status • Configure resources • Adjust authentication • View summary • Transfer summaries	**Services**
• Maintain development make system • Maintain make conductor system • Main build coordination system	• Maintain CI test interactions • Maintain configuration of Jenkins • Maintain DB • Main web summary	• Maintain product pull • Maintain build coordinator • Maintain documentation	• Maintain interfaces	**Projects**

Operational scenarios

List of several of the functions that can be performed under the services and projects aspects of the four defined areas.

Customers/Users Partnerships

- Experiment and project relations and support
  - Already have been heading down this path for IF experiments
  - Have support arrangements and procedures in place with several experiments
- Collaborative development and contributions

Tools maintenance and support

- Relocatable UPS
  - UPS for binary distributions without needing root access.
- Build systems
  - cetbuildtools, cetpkgsupport
- MRB
  - Simplify the building of multiple products pulled from separate repositories
- Open source and licensed products
  - Jenkins, Redmine, git, cmake, make.
  - TotalView, Allinea (profiler and debugger), valgrind, and Openl SpeedShop.