
http://openhpc.community

ISC BoF#24

# OpenHPC: A Community Supported HPC Repository

Karl W. Schulz
OpenHPC Project Lead, Intel

ISC Conference
Frankfurt, Germany
June 22, 2016

ISC High Performance
The HPC Event.

# Outline

- Motivation and brief history for this new community effort
- Update on project governance structure
- Stack overview
- What's coming
- Open Discussion/Feedback

# Motivation for Community Effort

- Many sites spend considerable effort aggregating a large suite of open-source projects to provide a capable HPC environment for their users:
  - necessary to build/deploy HPC focused packages that are either absent or do not keep pace from Linux distro providers
  - local packaging or customization frequently tries to give software versioning access to users (e.g. via modules or similar equivalent)

- OpenHPC is focused on establishing a centralizing community effort to:
  - provide a collection of pre-packaged binary components that can be used to help install and manage HPC systems throughout their lifecycle
  - implement integration testing to gain validation confidence
  - allow and promote multiple system configuration recipes that leverage community reference designs and best practices
  - provide additional distribution/integration mechanisms for leading research groups releasing open-source software
  - foster identification and development of relevant interfaces between supported components that allows for simple component replacement and customization

# a brief History…

- ISC'15 (June 2015) – BoF discussion on the merits/interest in a *Community Supported HPC Repository and Management Framework*
  - discussed convenience of distribution via standard Linux package managers
  - feedback at the time was that most parties were interested in CentOS/SLES
  - consensus that "modules" or equivalent needed to provide capable end-user development environment
- SC'15 (November 2015) – Follow on BoF for a *Comprehensive Open Community HPC Software Stack*
  - initial seeding of OpenHPC and a 1.0 release
  - variety of interested community members assembled thru the Linux Foundation to work towards establishing a formal, collaborative project
- Nov'15 – May'16
  - Linux Foundation working group collaborating to define participating agreement, initial governance structure and solicit volunteers

➨ *June 16, 2016 – Linux Foundation announces technical, leadership and member investment milestones with founding members and formal governance structure*

# OpenHPC: Project Members

THE LINUX FOUNDATION

Argonne National Laboratory

Altair

ARM

Atos

Avtech Scientific
attaining the vision

BSC Barcelona Supercomputing Center
Centro Nacional de Supercomputación

INDIANA UNIVERSITY BLOOMINGTON

CEA

CINECA

CRAY

DELL

FUJITSU

Hewlett Packard Enterprise

intel

BERKELEY LAB

Lawrence Livermore National Laboratory

lrz

Lenovo

Los Alamos NATIONAL LABORATORY EST. 1943

PARTEC CLUSTER COMPETENCE CENTER

PSC PITTSBURGH SUPERCOMPUTING CENTER

RIKEN

sgi

SUSE

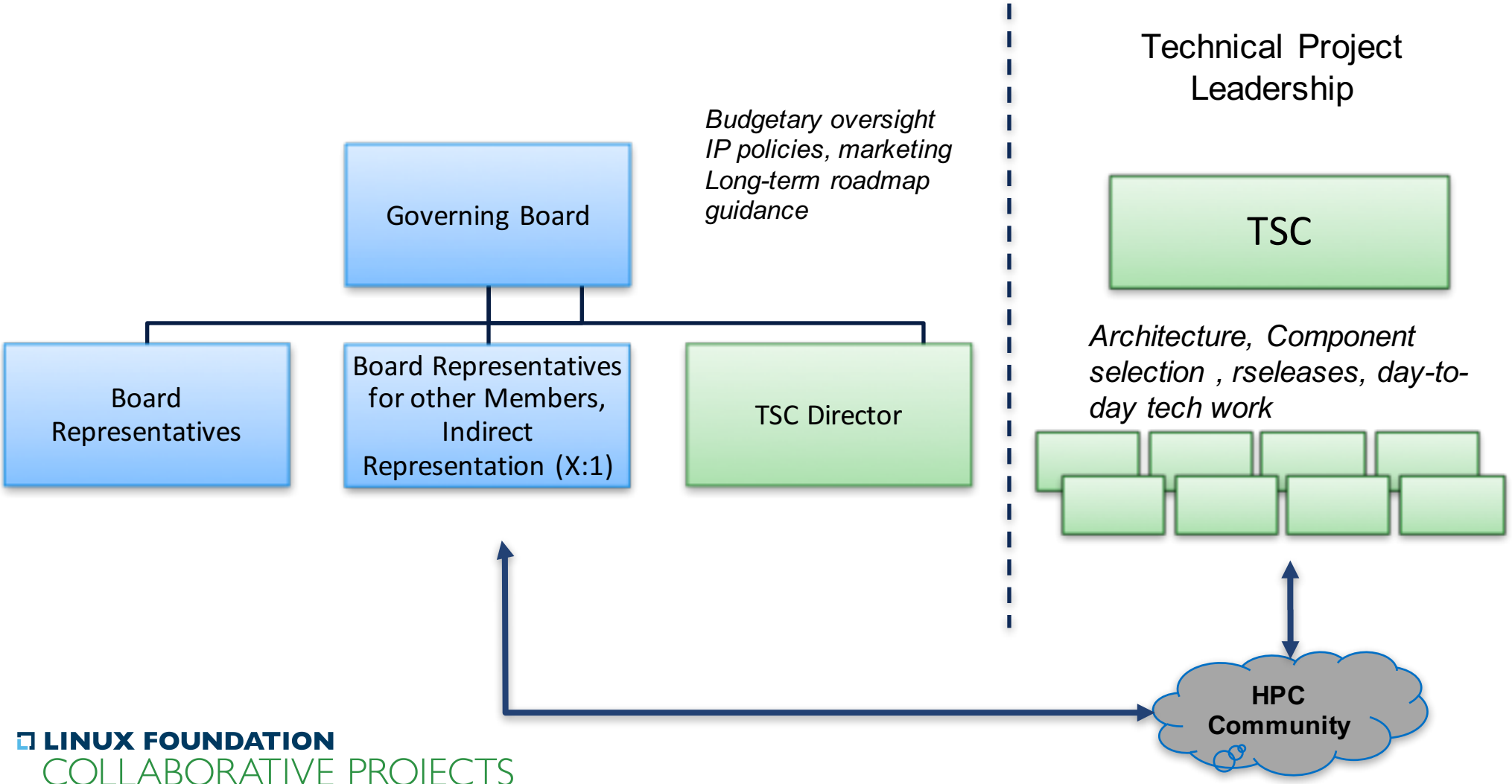
UNIVA

*Mixture of Academics, Labs, OEMs, and ISVs/OSVs*

Project member participation interest?
Please contact Jeff ErnstFriedman
jernstfriedman@linuxfoundation.org

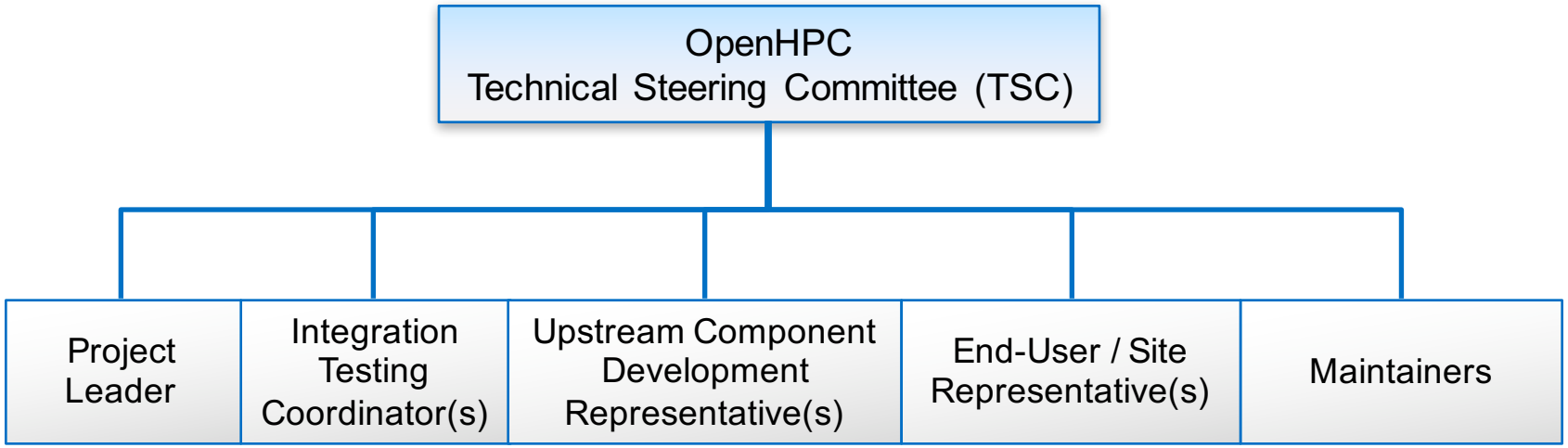
# Community Governance Overview

## *Governing Board + Technical Steering Committee*




LINUX FOUNDATION
COLLABORATIVE PROJECTS

# OpenHPC TSC - Role Overview



# OpenHPC TSC – Individual Members

- Reese Baird, Intel (Maintainer)
- Pavan Balaji, Argonne National Laboratory (Maintainer)
- David Brayford, LRZ (Maintainer)
- Todd Gamblin, Lawrence Livermore National Labs (Maintainer)
- Craig Gardner, SUSE (Maintainer)
- Yiannis Georgiou, ATOS (Maintainer)
- Balazs Gerofi, RIKEN (Component Development Representative)
- Jennifer Green, Los Alamos National Laboratory (Maintainer)
- Eric Van Hensbergen, ARM (Maintainer, Testing Coordinator)
- Douglas Jacobsen, NERSC (End-User/Site Representative)
- Chulho Kim, Lenovo (Maintainer)
- Greg Kurtzer, Lawrence Berkeley National Labs (Component Development Representative)
- Thomas Moschny, ParTec (Maintainer)
- Karl W. Schulz, Intel (Project Lead, Testing Coordinator)
- Derek Simmel, Pittsburgh Supercomputing Center (End-User/Site Representative)
- Thomas Sterling, Indiana University (Component Development Representative)
- Craig Stewart, Indiana University (End-User/Site Representative)
- Scott Suchyta, Altair (Maintainer)
- Nirmala Sundararajan, Dell (Maintainer)

***https://github.com/openhpc/ohpc/wiki/Governance-Overview***

# Stack Overview

- To start, we have assembled a variety of common ingredients required to deploy and manage an HPC Linux cluster including provisioning tools, resource management, I/O libs, development tools, and a variety of scientific libraries:
  - the delivery mechanism is via standard package managers (ie, *there are public OpenHPC repositories)*
  - general use model starting from bare metal typically goes like:
    1. install BaseOS on administrative host (master)
    2. enable OpenHPC repo → `rpm/zypper install ohpc-release*.rpm`
    3. use installation recipe (or provided scripts) to install desired packages to support back-end system and augment with desired development tools, resource manager, etc
  - of course, relevant bits can also be pulled into existing systems or other environments (e.g. cloud)

# Stack Overview

- Packaging efforts have HPC in mind and include compatible modules (for use with Lmod) with development libraries/tools
- Endeavoring to provide hierarchical development environment that is cognizant of different compiler and MPI families
- Include common conventions for env variables
- Development library install example:

```
# yum install petsc-gnu-mvapich2-ohpc
```

- End user interaction example with above install: (assume we are a user wanting to build a PETSC hello world in C)

```
$ module load petsc
$ mpicc -I$PETSC_INC petsc_hello.c -L$PETSC_LIB –lpetsc
```

# Development Infrastructure

# OpenHPC Development Infrastructure

*What are we using to get the job done….?*

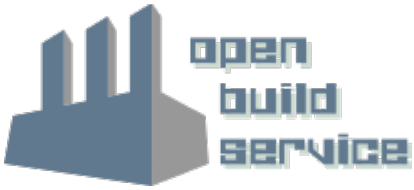
## The usual software engineering stuff:

- GitHub (SCM and issue tracking/planning)
- Continuous Integration (CI) Testing (Jenkins)
- Documentation (Latex)


https://github.com/openhpc/ohpc

## Capable build/packaging system

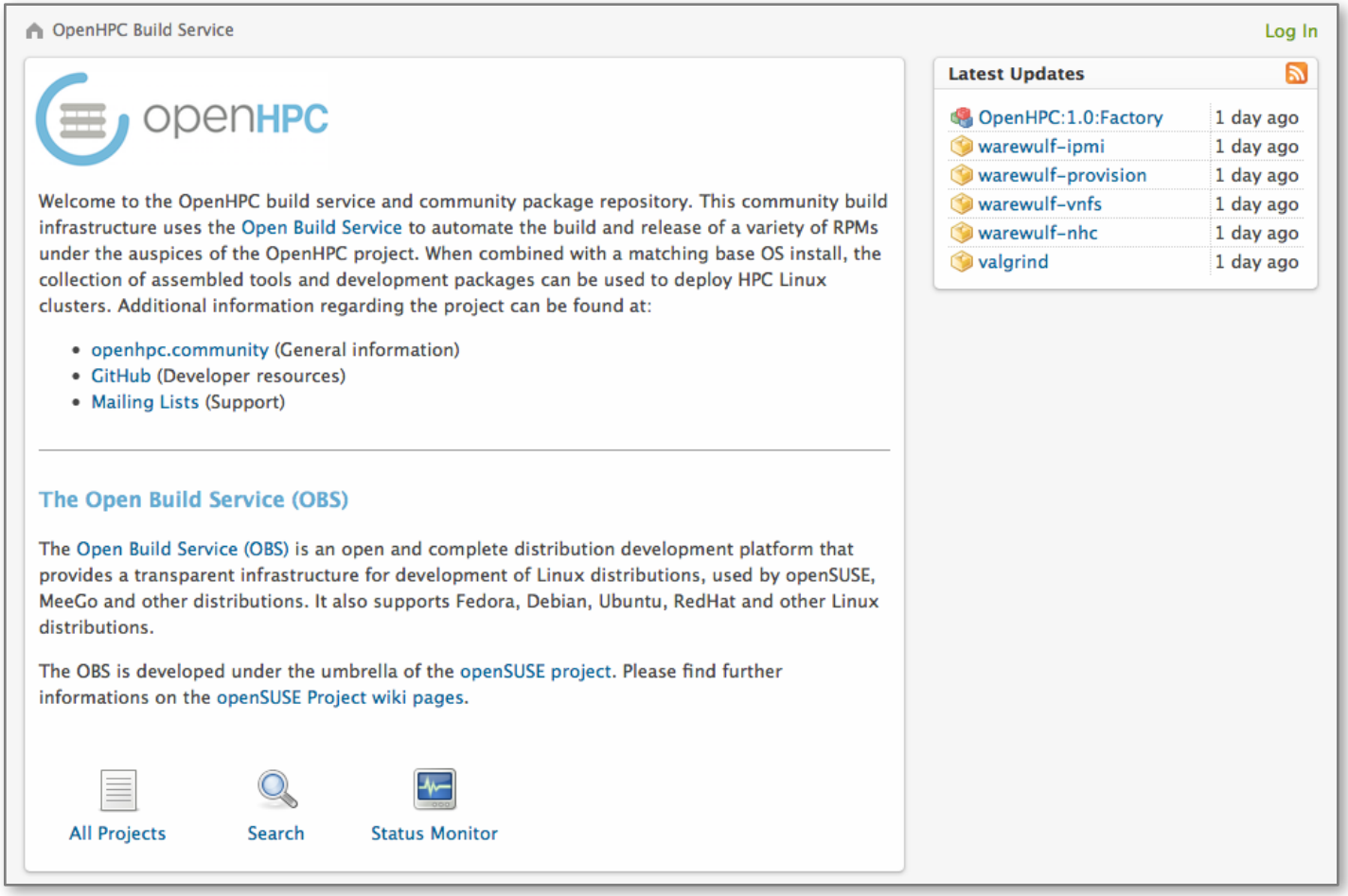
- At present, we target a common delivery/access mechanism that adopts Linux sysadmin familiarity - ie. **yum/zypper** repositories for supported distros
  - ultimately delivering RPMs
  - [base] + [update] repositories to support life-cycle management
- Require flexible system to manage builds for multiple distros, multiple compiler/MPI family combinations, and dependencies across packages
- Have engineered a system using Open Build Service (OBS) which is supported by back-end git
  - git houses .spec files, tarballs, patches, documentation recipes, and integration tests
  - OBS performs automated builds and dependency analysis




https://build.openhpc.community

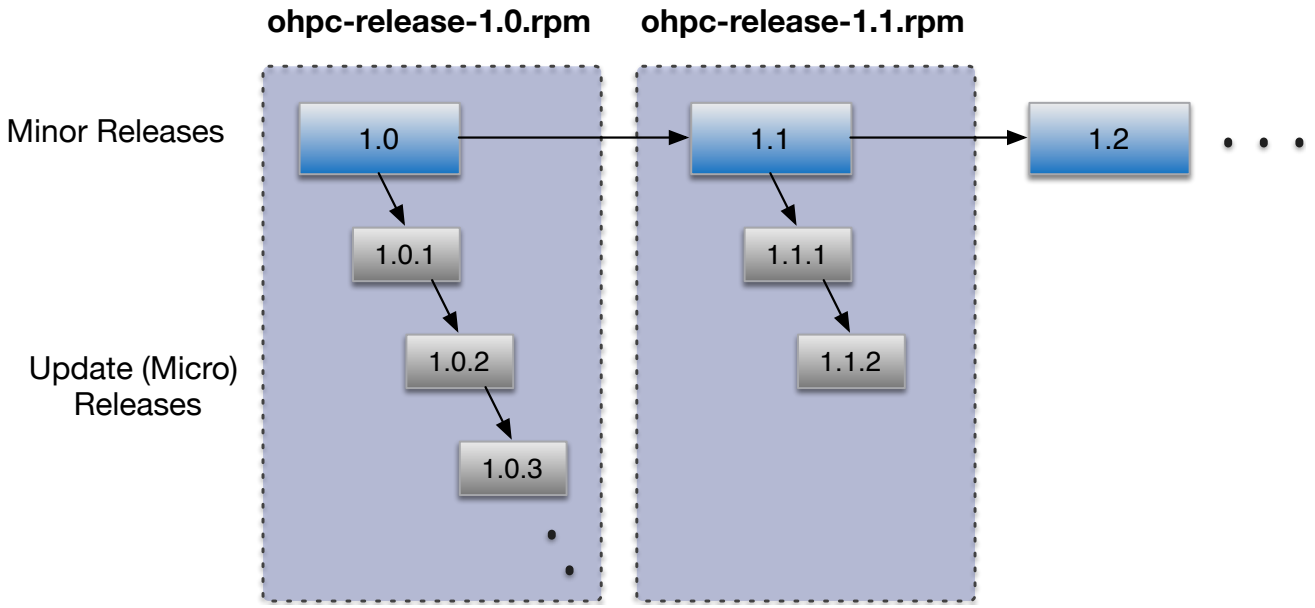
# Build System - OBS

https://build.openhpc.community




- Using the **Open Build Service** (OBS) to manage build process
- OBS can drive builds for multiple repositories
- Repeatable builds carried out in chroot environment
- **Generates binary and src rpms**
- Publishes corresponding package repositories
- Client/server architecture supports distributed build slaves and multiple architectures

# Initial version history



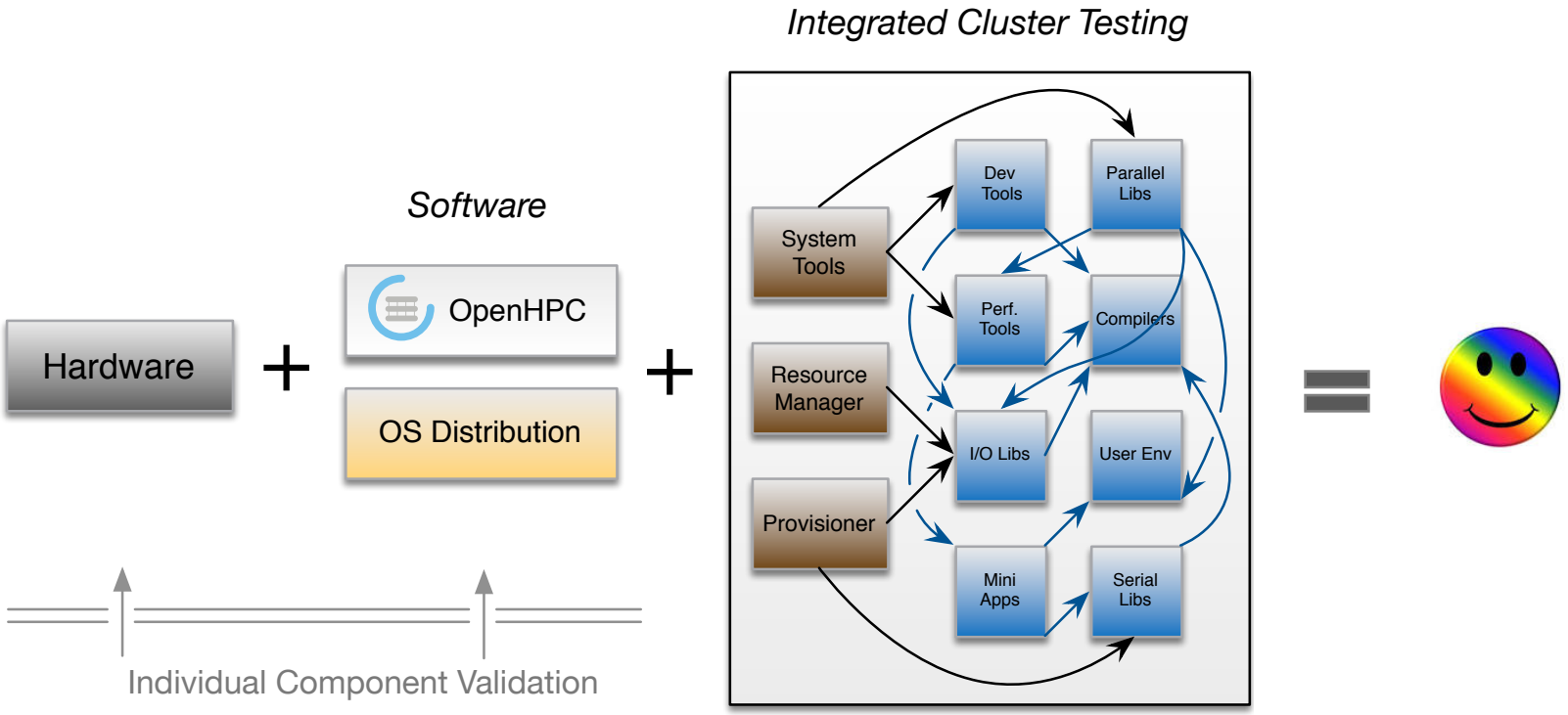

- 1.0: November 2016
  - initial release,
  - builds against CentOS base OS
- 1.0.1: February 2016
- 1.1: April 2016
  - include support for SLES12
  - improved suport for diskfull provisioning, component additions and version updates
- 1.1.1: June 2016

# Integration Testing

# Integration/Test/Validation

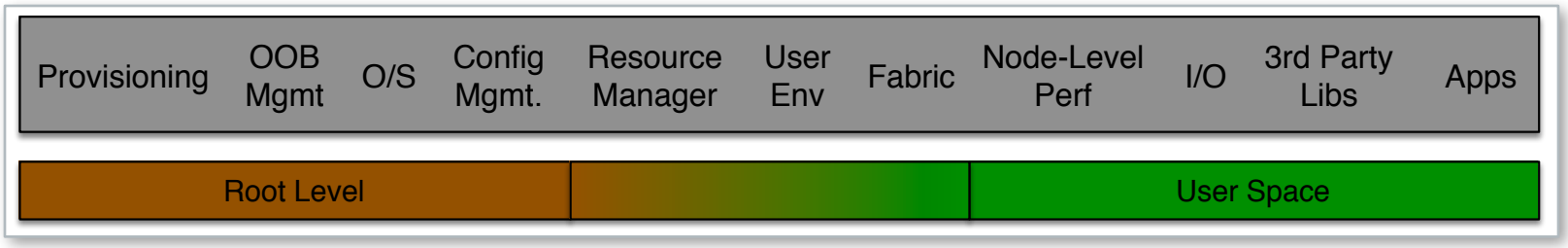
Testing is a key element for us and the intent is to build upon existing validation efforts and augment component-level validation with targeted cluster-validation and scaling initiatives including:

- install recipes
- cross-package interaction
- development environment
- mimic use cases common in HPC deployments
- upgrade mechanism



# Integration/Test/Validation

- To facilitate global efforts in diagnostics/validation, we have devised a standalone integration test infrastructure
- Intent was to create families of tests that could be used during:
  - initial install process (can we build a system?)
  - post-install process (does it work?)
  - developing tests that touch all of the major components (can we compile against 3rd party libraries, will they execute under resource manager, etc)
- Expectation is that each new component included will need corresponding integration test collateral
- These integration tests are included in GitHub repo



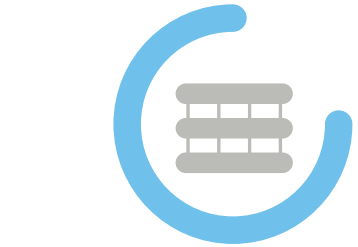
# What's Coming

- Some known big ticket items on the horizon for the TSC
  - establishing a process and prioritization/selection process for including new software components
  - establish minimum integration test expectations
  - establish packaging conventions:
    - naming schemes
    - dependency hierarchy management
    - installation paths
    - upgrade/rollback? mechanisms
  - roadmap timeline for next release (and cadence strategy for future releases)
  - addition of public CI infrastructure, roll out of additional architecture builds

# Open Discussion

# Discussion seed items….

- Things you like/hate with current direction?
- Importance of integrating R&D efforts that may be a different maturity points?
- Importance of developing abstraction layers to accommodate HPC use cases with alternate component choices
  - e.g. resource manager driven provisioning with alternate provisioners
- base distro choices?
  - CentOS
  - SLES
- Openness to alternate "package manager" equivalent?
- Ease of use ("turnkeyness")
- Cloud images?
- Package relocation?
- Other items you would like to the TSC priority list?




http://openhpc.community

# Thanks for your Time - Questions?

karl.w.schulz@intel.com

http://openhpc.community

https://github.com/openhpc/ohpc

https://build.openhpc.community (repo)