

# The Cluster Integration Toolkit

(An Extensible, Portable, Scalable Cluster Management Software Implementation)

## Cluster World Conference and Expo

25 June 2003

**James H. Laros III**

**Principal Member of Technical Staff**

**Sandia National Labs**

***jhlaros@sandia.gov***



Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.





# Overview

---

- **Goals**
- **Implementation**
- **Installation**
- **Tools**



# Goals

(Don't assume)

---

- **Broaden scope**
  - Originally developed to provide a solid foundation to the Cplant™ run-time system.
  - Target clusters in general rather than specific clusters used for specific purposes
  - Architectures (topologies and components) we have now...
  - Architectures we will have...
  - Architectures we haven't thought about yet!!



# Goals

(cont.)

---

- **Automate as much of the integration process as possible**
  - **Discovery process**
  - **Configuration file generation**
- **Ease Cluster Management**
  - **Supported by Unix system administrators not “cluster experts”**
  - **Same interface for every cluster, large or small**
- **Disk-less node support**
  - **Almost no work in this area when we started, especially to the scale we were targeting (5-10k nodes)**
- **Don't modify kernel (use commodity)**
- **Separation of Cluster Management and Run-time system**



# Implementation

---

- **Design**
- **Language**
- **Modules**



# Implementation

## (Design)

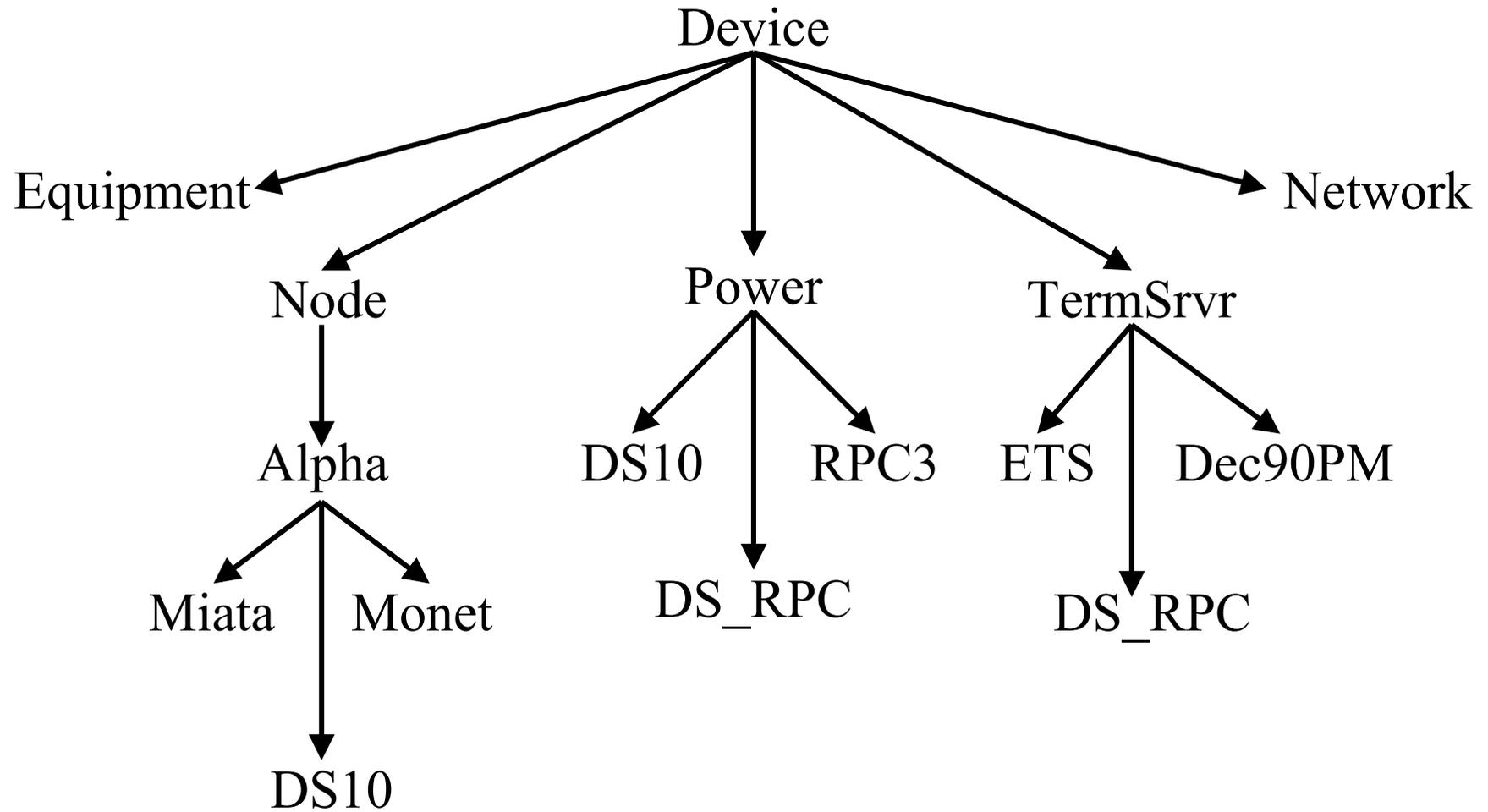
---

- **Device Class Hierarchy**
  - Hierarchical organization of devices and their capabilities
- **Persistence Object Store**
  - Representation of the Physical Cluster (hardware and topology)
  - Instantiated Objects from Device Classes
  - Linkage describes the specific topology of the cluster
  - Provides foundation for tools
- **Layered Utilities**
  - Capabilities constructed in “layers” range from
    - Low-level orthogonal utilities
    - To high-level user interfaces



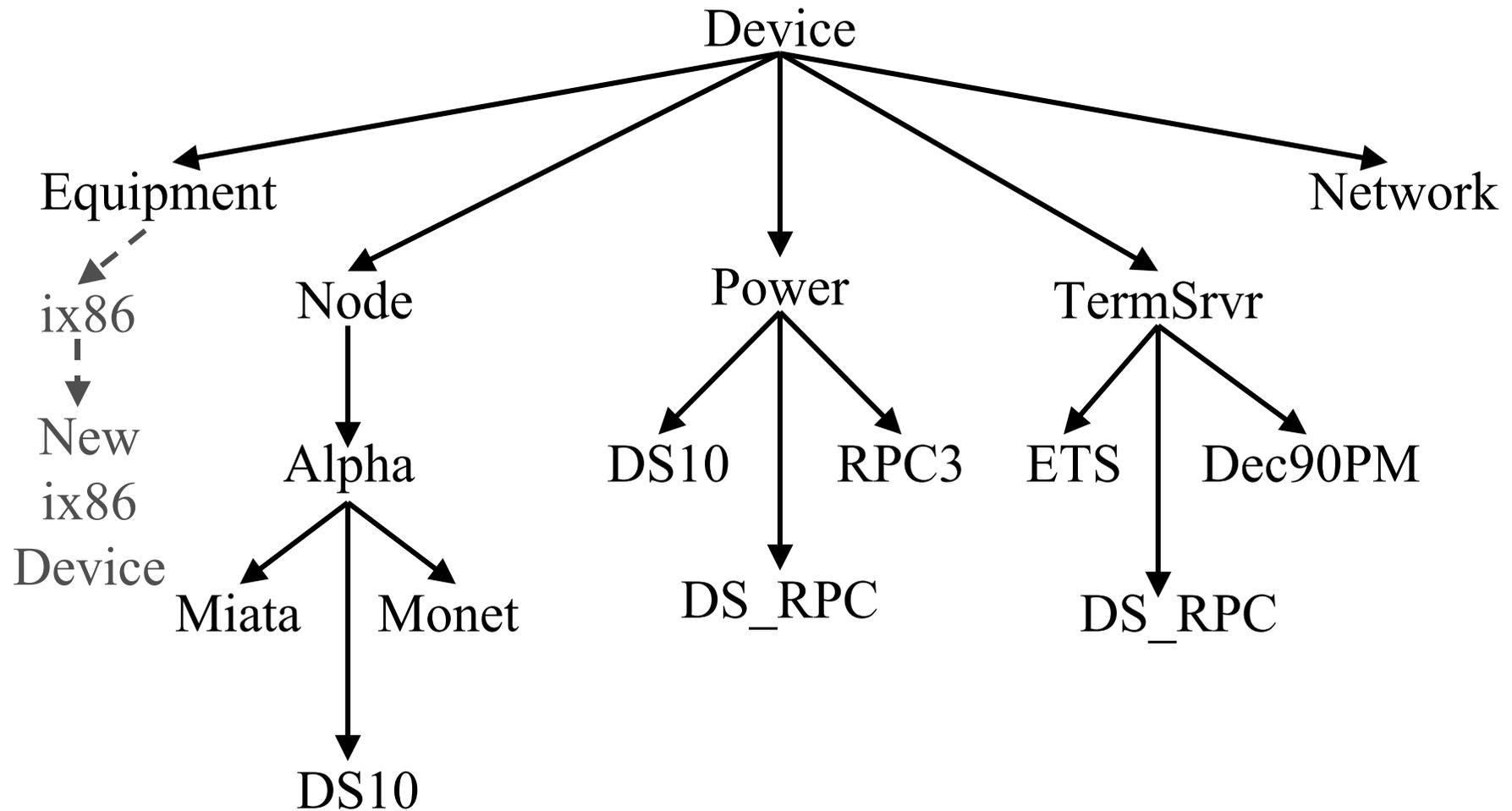
# Device Class Hierarchy

---





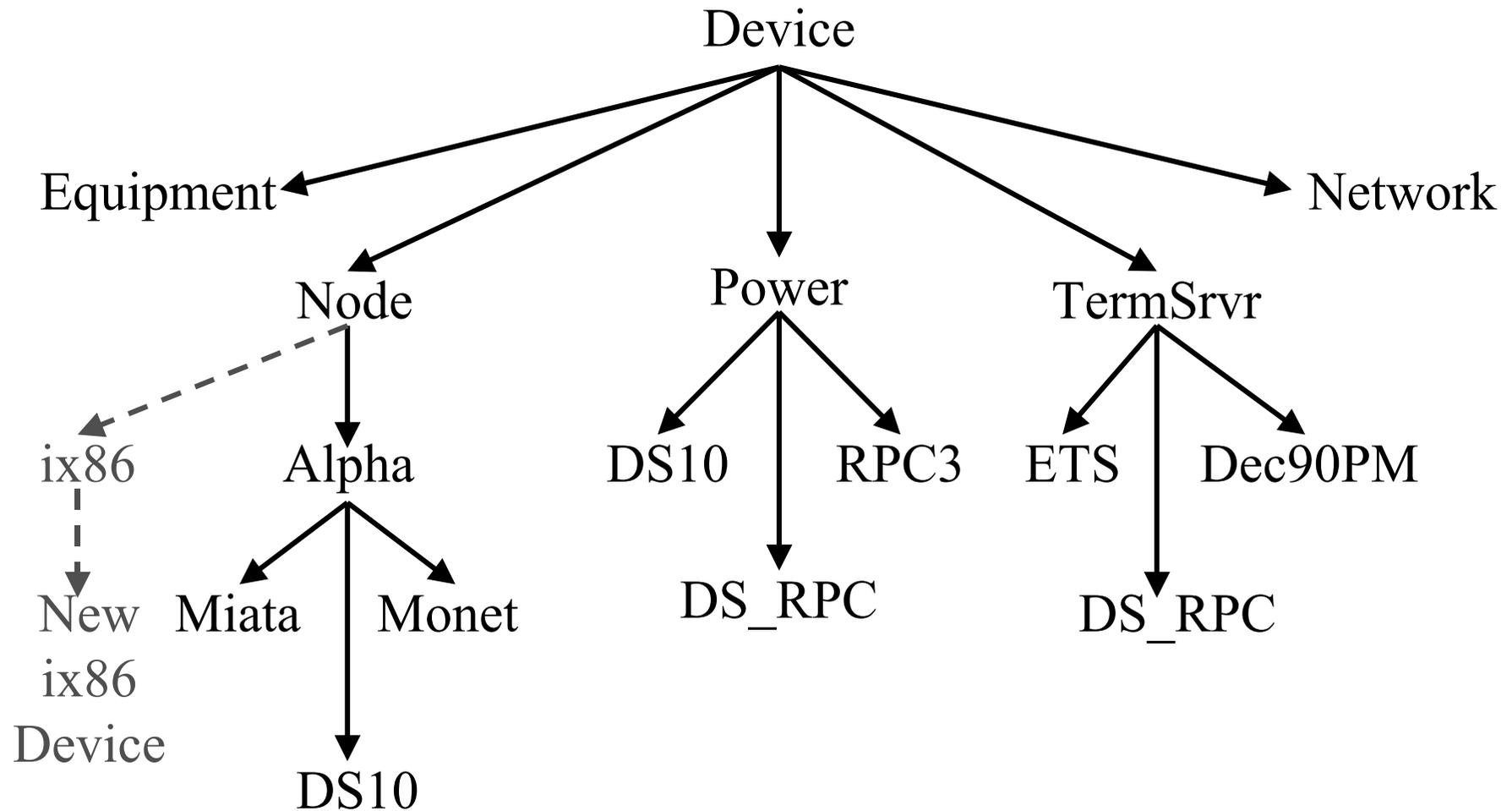
# Adding A Device Class





# Adding A Device Class

(cont)





# Implementation

(Design Cont)

---

- Discussed in detail:  
***“An Extensible, Portable, Scalable Cluster Management Software Architecture”*** – published in the proceedings of the International Conference on Cluster Computing (Cluster 2002) October 2002.



# Implementation

## (Language)

---

- **Why Perl**
  - **Portability**
  - **Supports Object-oriented as well as other programming paradigms**
  - **Numerous open source contributions (CPAN)**
  - **Quick clean implementations**



# Implementation

## (Modules)

---

- **CIT separated into modules**
- **Pick and choose modules you need to install**
  - **Base**
    - **Core of CIT**
    - **Contains low to high-level commands and libraries**
  - **Site**
    - **Site specific or potentially site specific libraries and commands**
  - **Config, Disk-less, Disk-full, etc**



# Installation

---

- **Initial Installation of CIToolkit Software**
- **Database (Persistent Object Store)**
- **Hardware Installation**
- **Cluster Software Customization and Configuration File Generation**
- **Initialization and Discovery**
- **Troubleshooting and Diagnostics**
- **Installation of the Run-time Environment**



# Installation

## (Initial)

---

- **Pre-hardware install benefit**
  - Can even be used to design your cluster and work out topology, number of devices needed, IP scheme etc.
- **Uses “make”**
  - **Small number of variables that require definition**
  - **Generates other config files**
    - For other installation processes like cfengine
    - For the installed CIToolkit
  - **Majority of CIToolkit just copied into place**
    - At least the portion that we have contributed!
    - Exceptions for things like Myrinet or other 3<sup>rd</sup> party contributions



# Installation

(“Database” Persistent Object Store)

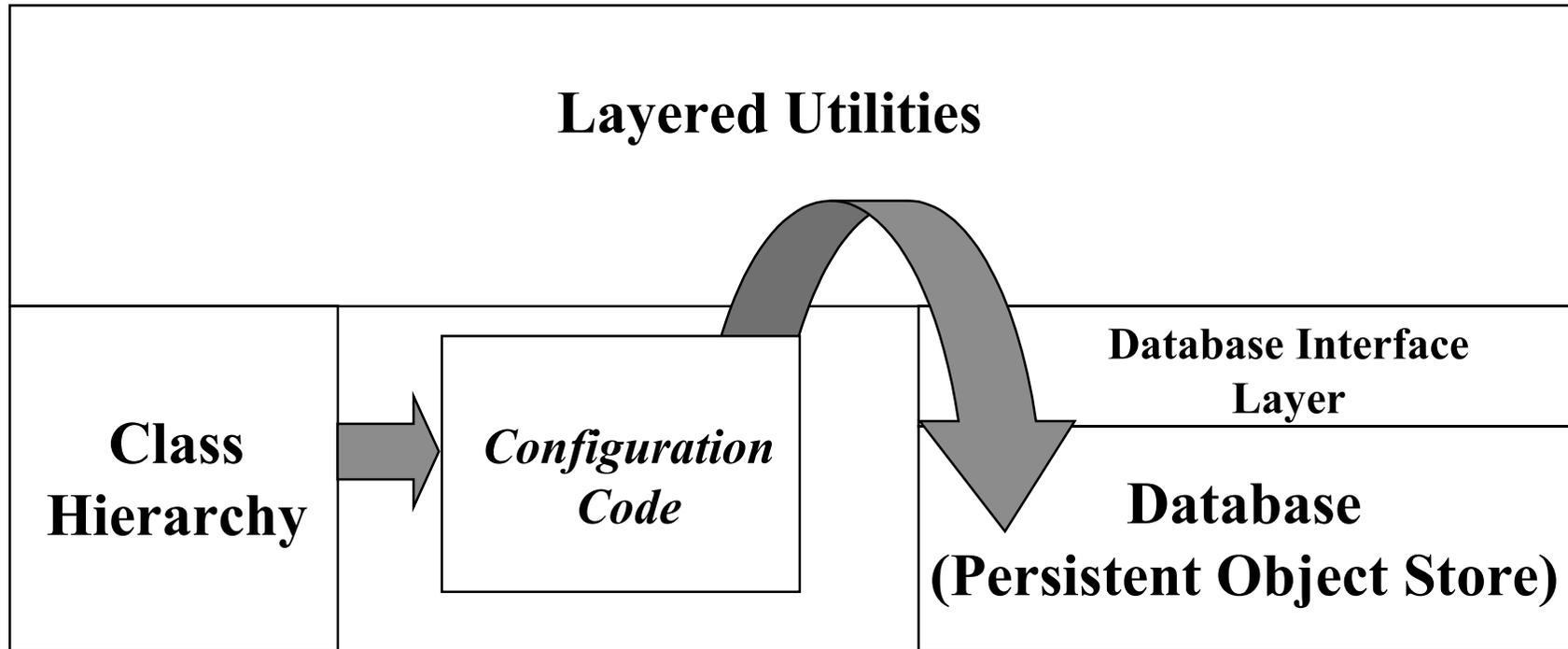
---

- **Review** – represents the devices and topology of “your” cluster
- **Programatic generation of database**
  - **Most flexible**
  - **Basically unlimited capability of description**
- **Library Interface**
  - **Somewhat more limited**
    - **But not realistically**
  - **Command line interface to utilize libraries**
    - **For initial generation**
    - **Subsequent manipulation**
    - **Can be leveraged by other tools (GUI interfaces)**
    - **Conversions from other formats (flat file, power-point, etc.)**
  - **Most importantly**
    - **Approachable for first time users, non-programmers, etc.**



# Database Creation

---





# Installation

## (Hardware)

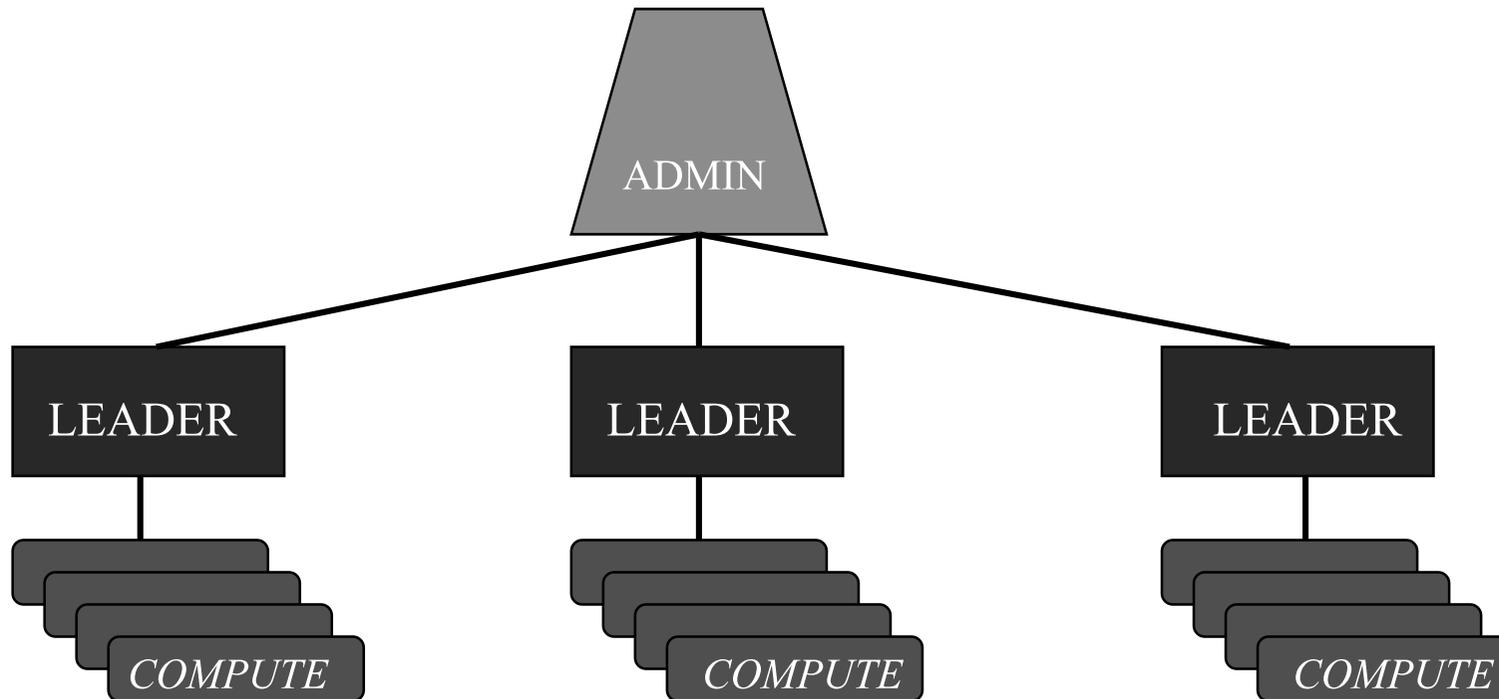
---

- **Not much to automate**
- **Can help with some initial configuration**
  - **CIToolkit on Linux PDA**
- **Diagnostics performed at many steps during installation**



# A Basic Architecture

---





# Installation

## (Cluster Software Customization and Configuration File Generation)

---

- **Disk-less Cluster Install**
  - **Generate bootable hierarchy**
    - Hierarchical Infrastructure to support a disk-less cluster using NFS (yes I did say NFS)
    - Automates OS install
      - Base is standard RH image
    - Config file generation
      - Ex. Hosts, dhcpd.conf etc.
    - Specificity ranges from general (entire cluster) to granularity down to the node level
  - **Supports multiple hierarchies**
    - Allows support for multiple OS versions
- **Can be leveraged for Disk-full installations also!**



# Installation

## (Initialization and Discovery)

---

- **Discovery**
  - **Discovers information about the cluster**
  - **Uses topology and device information contained in database**
  - **Populates database with information about the specific device**
    - **Like MAC address**
      - **Used later in configuration file generation**



# Installation

(Troubleshooting and Diagnostics)

---

- **Can be helpful in all stages of install and subsequent maintenance.**
- **Usually found in diag module**
- **Good example of integrating open-source and 3<sup>rd</sup> party software**
- **Both locally written and open-source utilities benefit from the CIToolkit infrastructure**



# Installation

(Installation of the Run-time Environment)

---

- **Review – separation of infrastructure and run-time environment**
  - Any run-time
  - Single entry point to initialize run-time
  - Virtual machine concept for logical partition of physical machine
    - Can run multiple run-time systems
    - Multiple versions of same run-time system
    - Combinations



# Tools

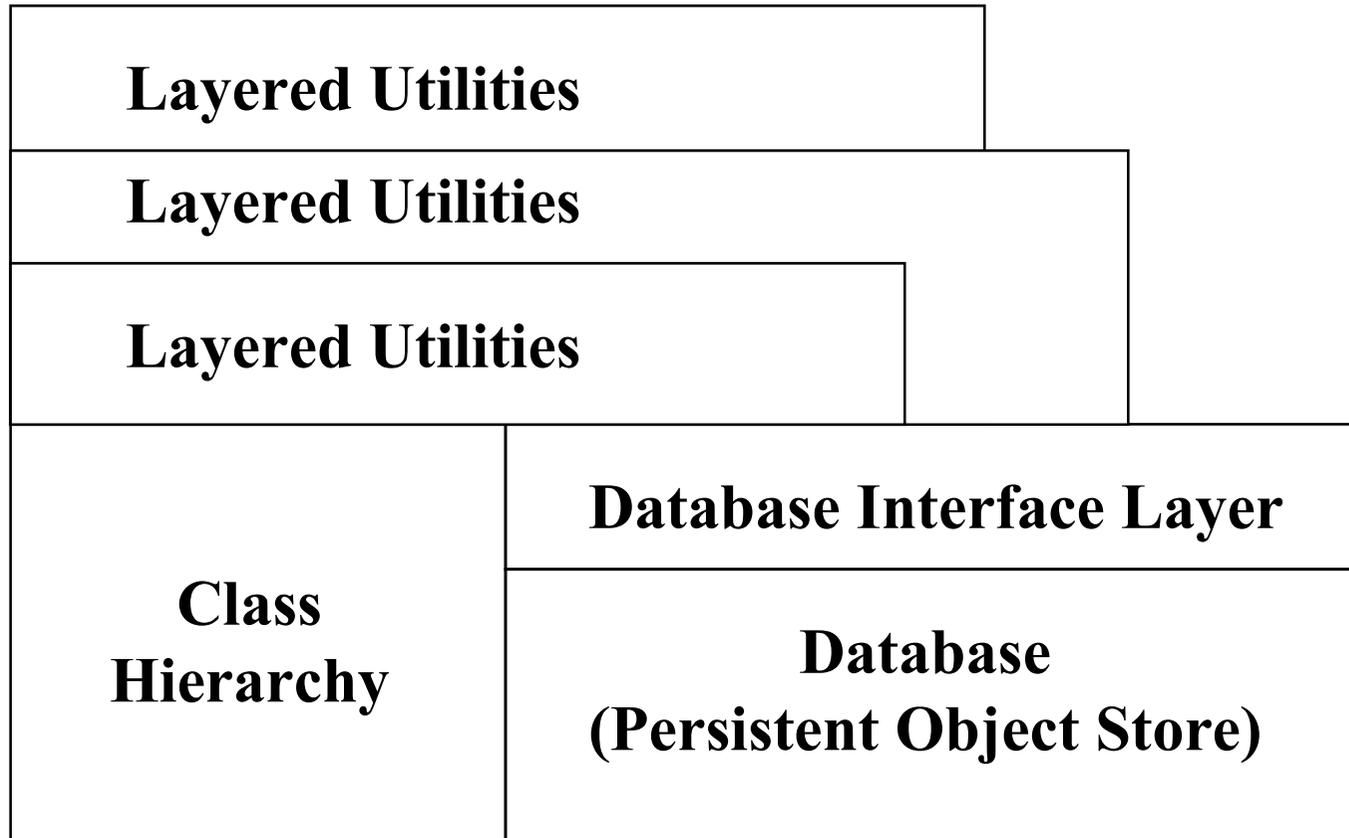
---

- **Database Tools**
- **Configuration Tools**
- **Low-level Operation Tools**
- **High-Level Operation Tools (Csuite of Tools)**
- **Additional High-Level Tools, Interfaces, and Daemons**



# Layered Utilities

---





# Database Tools

---

- **Tools for creation and subsequent manipulation of database**
  - **Dump and restore utilities**
    - Useful for displaying information about objects, groups etc.
    - Create flat file backup
    - Re-import for purposes of restore, binary format change, etc.
  - **Object editing**
    - Change, add, delete information about a device object
  - **Group (collection) generation**
    - Can be temporary or permanent
    - Groups can be generated for many reasons
      - Command execution
      - Inventory



# Configuration Tools

---

- **Generate files needed for cluster to operate**
- **Linux config files**
  - /etc/hosts
  - /etc/dhcpd.conf
- **Linux config files specific to each node**
  - /etc/sysconfig/ifcfg-eth0
- **Easy to add new configuration tools**
- **Bootable hierarchy**
  - Reference: *“Implementing Scalable Diskless Cluster Systems using the Network File System (NFS)”*



## Low-level Operation Tools

---

- **Single function utilities intended to be leveraged by higher level commands.**
- **Examples:**
  - power
  - status
  - boot
  - console
- **Do what their names imply, not much more.**



## High-Level Operation Tools (Csuite)

---

- **More flexible, user-friendly**
  - **cboot**
  - **cstatus**
  - **ccmd**
- **Abstract complexity away from user**
- **Flexibly parse user input in common way**
- **Leverage roles, groups etc.**
- **Site specific things like “names” isolated**
  - **Site module**
- **Apply parallelism where appropriate to achieve scalability**
- **Leverage low-level utilities for core functionality**



# Additional High-Level Tools, Interfaces and Daemons

---

- **Command/Status Daemons**
  - Adds robustness and reliability
  - More opportunity for local customization
- **GUI's**
  - Perl/Tk based
  - Curses Based
  - Web Based
  - CHITS (Cluster Hardware Issue Tracking System)
  - Others easy to implement
- **Watch-console**
  - Enhance console monitoring capability



## Future Work

---

- **Develop additional systems management tools**
- **Data Capture utilities**
- **Support for additional system architectures, new hardware**
- **Disk-full solution**
- **Other disk-less solutions**
- **Add-on functionality for select run-time systems**
- **Anything to ease Cluster Integration and Management**
- **Prove concepts on more than 1861 nodes**



## Conclusions

---

- **Take this task seriously**
  - What you put in is what you will get out
  - The larger you get the more it matters
- **Rigorous approach pays off**
- **Focus on similarities**
  - Pays off with flexibility
- **So far our design and implementation have fulfilled our stated goals**



# Questions???

---

[jhlaros@sandia.gov](mailto:jhlaros@sandia.gov)

<http://www.cs.sandia.gov/cit>

