Lifemapper Server Virtualization

Aimee Stewart (KU) astewart@ku.edu
Nadya Williams (UCSD) nadya@sdsc.edu

PACIFIC RIM APPLICATIONS AND GRID MIDDLEWARE ASSEMBLY

PRAGMA

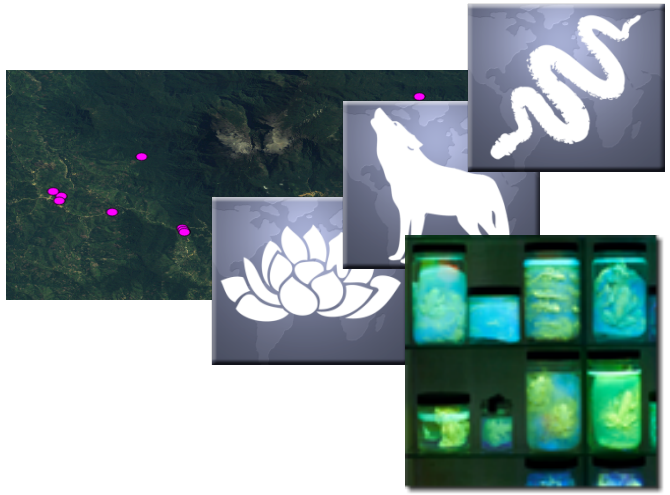
… Is Way Harder Than We Thought

THE UNIVERSITY OF KANSAS

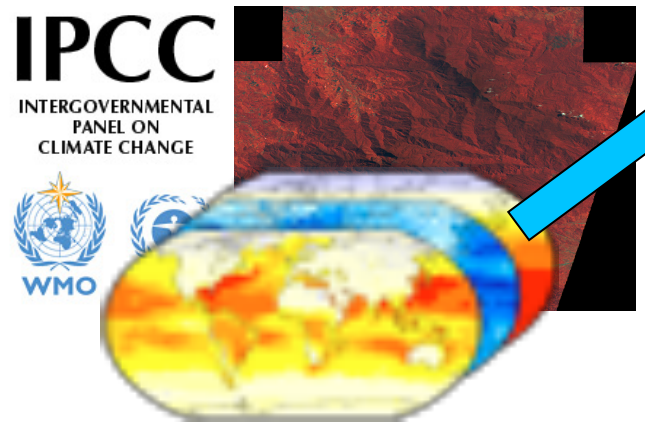
Lifemapper

- Data library
  - Climate
    - Observed
    - IPCC Predicted Future Climate
  - Species
    - Occurrence Points
    - Potential habitat maps
- Tools
  - LmSDM: Species Distribution Modeling
  - LmRAD: Range and Diversity

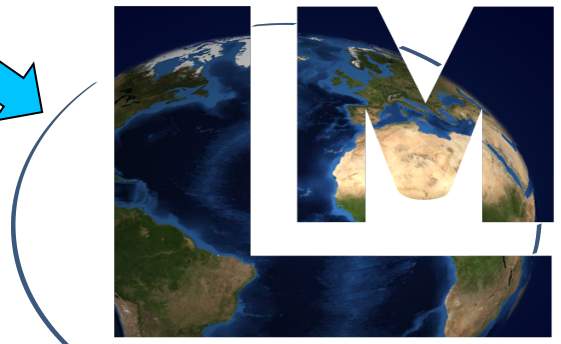
LmSDM:
Species Distribution Modeling

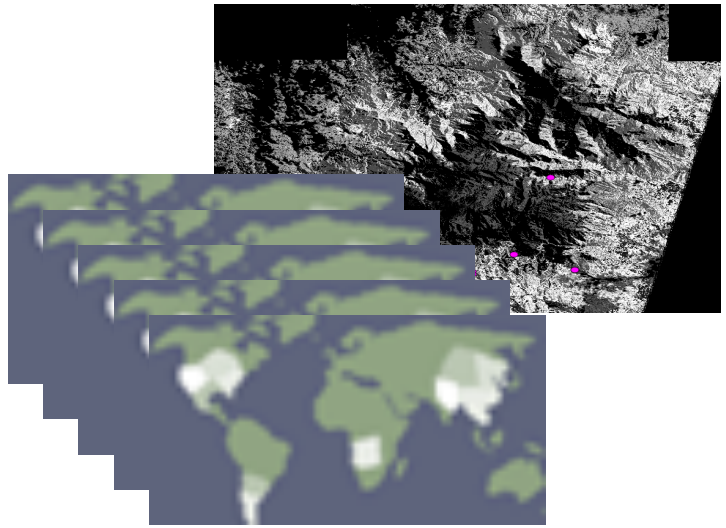


Species Occurrence Data



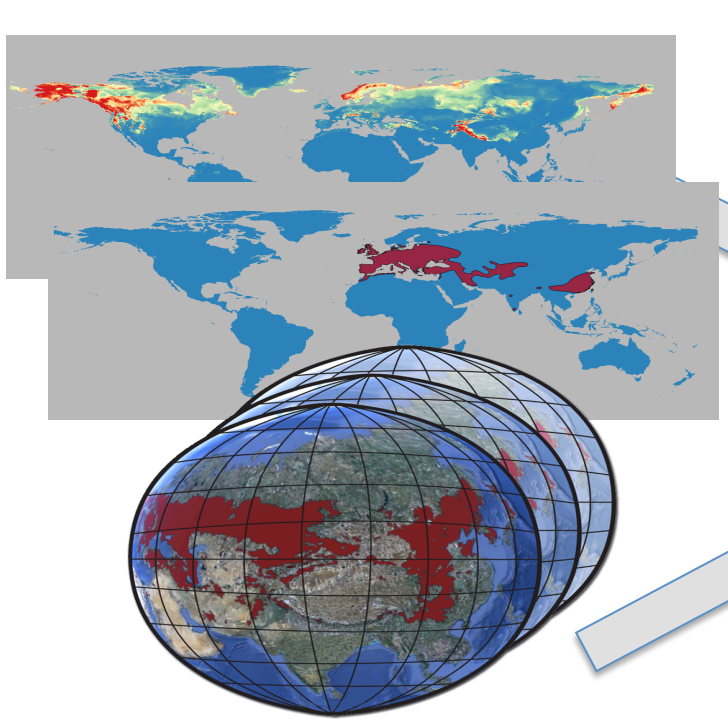
Environmental Data



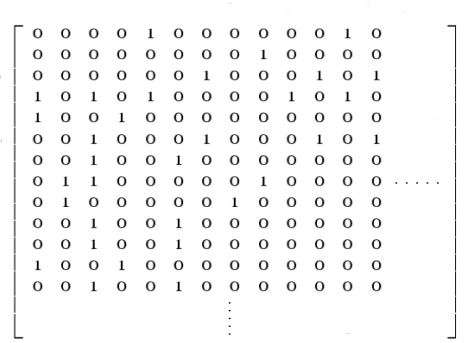


Potential Habitat

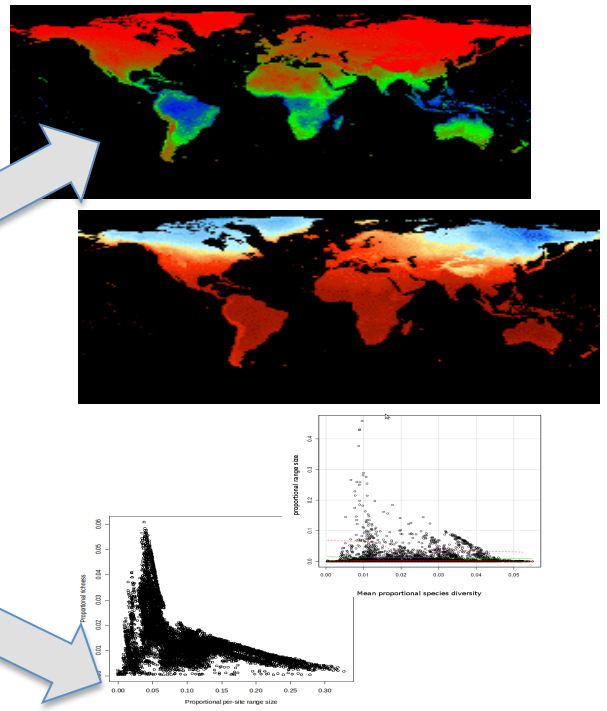
LmRAD:
Range and Diversity



Species Habitat Data



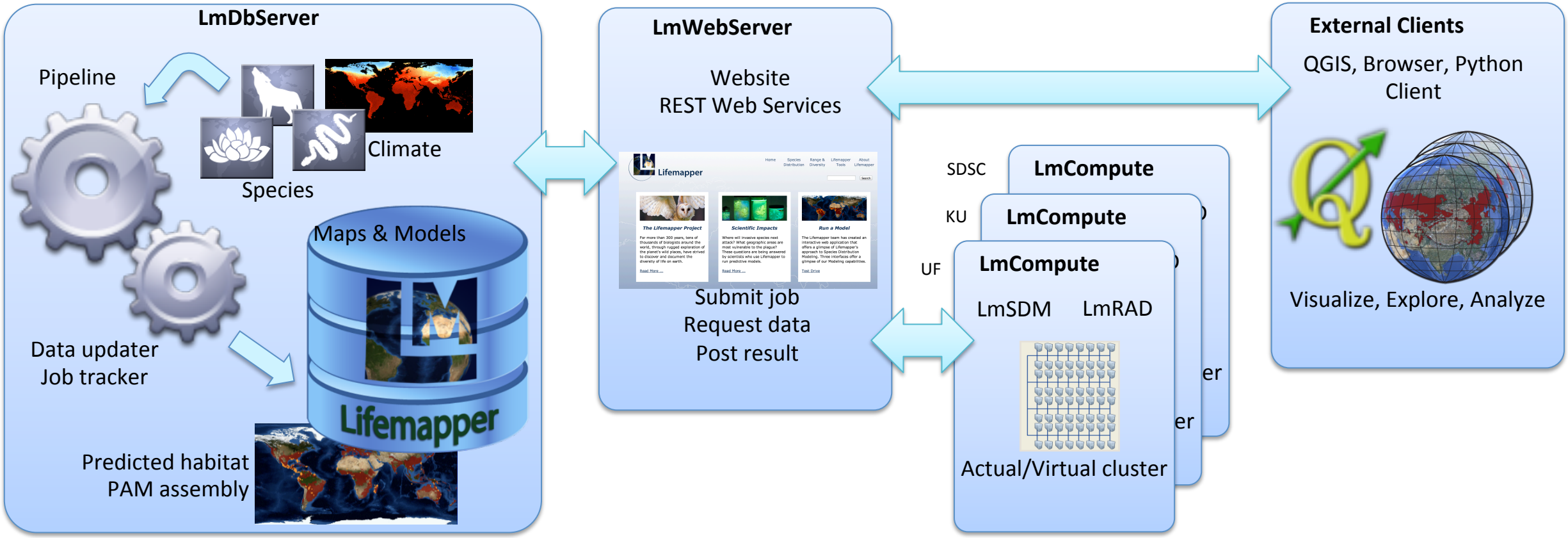
Presence Absence Matrix (PAM)



Multi-species analyses

Range and Diversity
Quantifications

Milestones: Lifemapper refactoring



PRAGMA 26 VBE Goals

PRAGMA 26 Goals: Decouple Lifemapper Server Components from KU

- *Enable configuration through initialization files*
- *Seed the database with site-specific inputs*
- *Connect the virtualized cluster (PRAGMA 25) to compute jobs*
- *Start the data pipeline to create computational jobs*

Issues:

- *We resolved KU-specific configuration problems*
- *Discovered data-specific bugs, not encountered in fully populated original installation*

Some further PRAGMA goals:

- **Instantiate Lifemapper with Mount Kinabalu data at UF**
- **Customize and extend data workflow through configuration**

Lifemapper Server Virtualization

Domain scientist's viewpoint:

1. Extend previous Lifemapper work to enable data management (LmDBServer) and web services (LmWebServer) components virtualization
2. Increase the availability and flexibility of Lifemapper to enable scientists to
   - *Assemble multi-species macro-ecology experiments*
   - *Perform other LM-facilitated data processing on:*

Unique datasets
Restricted use data
Very large datasets

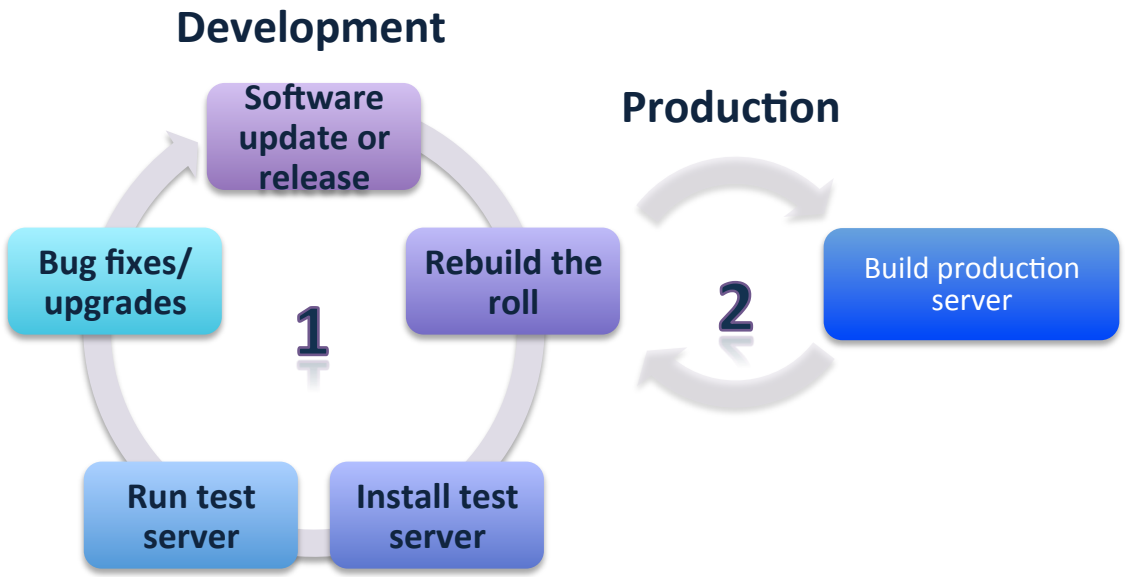
Cyber-infrastructure viewpoint:

1. Continue practical use of PRAGMA cloud infrastructure
2. Encapsulate the complexity of software build/configure in ROCKS rolls
3. Create a complete system as an end-to-end solution
4. Reduce cost of installing/configuring/replicating

What we are trying to do

- **Increase availability and flexibility of Lifemapper Server as a complete system**
  - reduce cost of installing/configuring/replicating and ease burden of integrating hardware and software
- **Enable a fast "workflow" from software update to server availability:**
  - Minimize time spent on software build and configuration
  - Automate most hands-on tasks.
  - Essential: have test cases for all installed components and their configuration
- **Prepare for greater quantity and quality of data and complexity of operations**
  - From low resolution climate data to high resolution satellite imagery for Mt. Kinabalu
  - From simple single-species SDM experiments to multi-species macro-ecology experiments with more species

This work is a part of PRAGMA's "Resources and Data" working group



Lifemapper

http://lifemapper.org

https://github.com/lifemapper/

Rocks

http://www.rocksclusters.org

Pragmagrid GitHub

https://github.com/pragmagrid/lifemapper

https://github.com/pragmagrid/lifemapper-server

Why Automate

Manual hands-on tasks:

- Software packages build
- Custom scripts
- User work-around
- Repetitive tedious functions
- Tracking errors, exceptions and problems

Time

Automation is no longer nice t… …t have that allows to:

- Know your system real-time status: what i… …nfigu…ation, data population
- Robust system: can reliably build and r…
- Address complexity of configuring – n…
- What issues need to be documented?

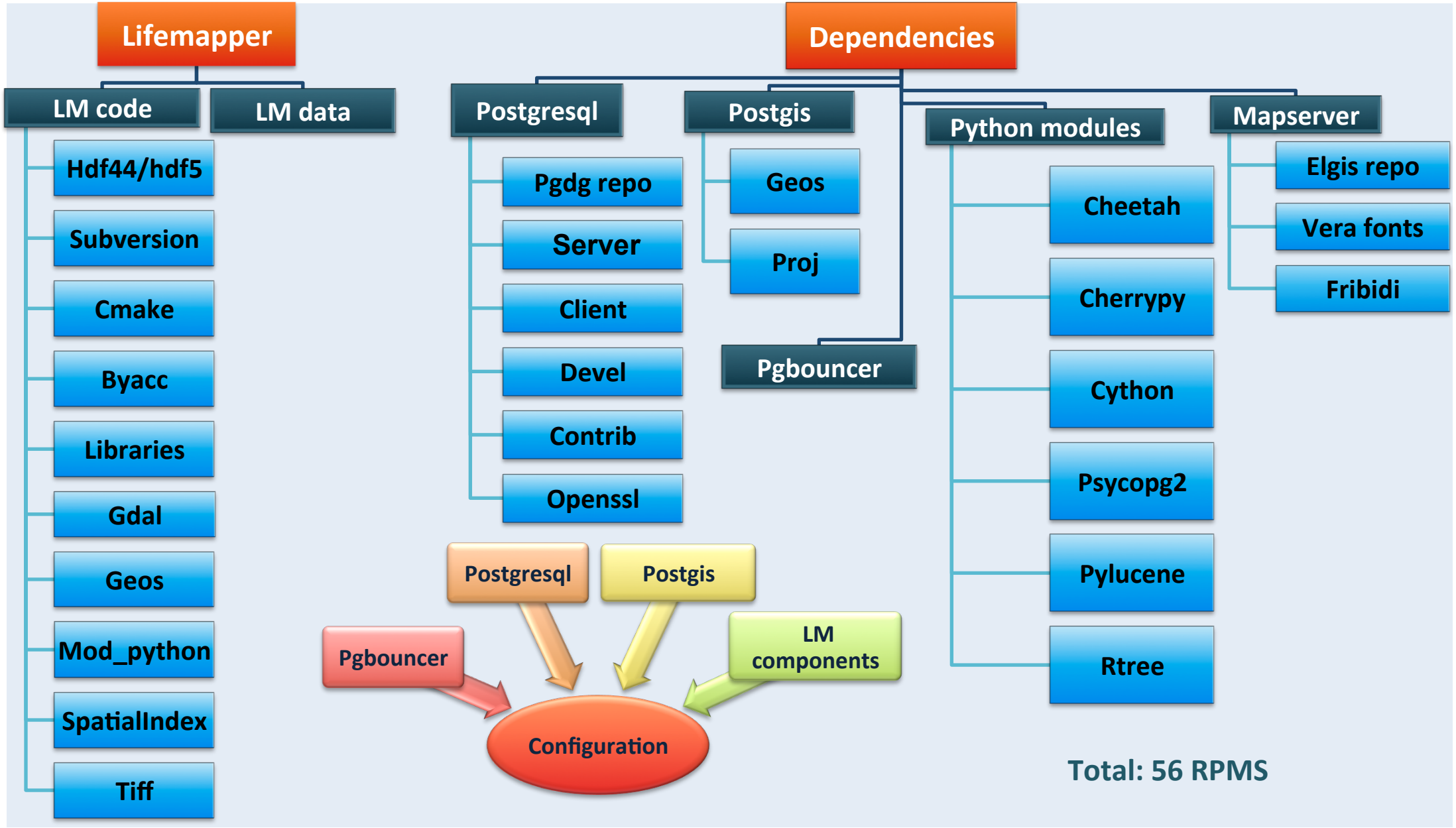
Time

Facilitate easy-to-use solution to provide seamless integration of hw/sw

- Allow to virtualize infrastructure to improve hw utilization and scalability
- Enable operational efficiency and flexibility: deployment and operation of virtual servers
- Enables to clone multiple servers

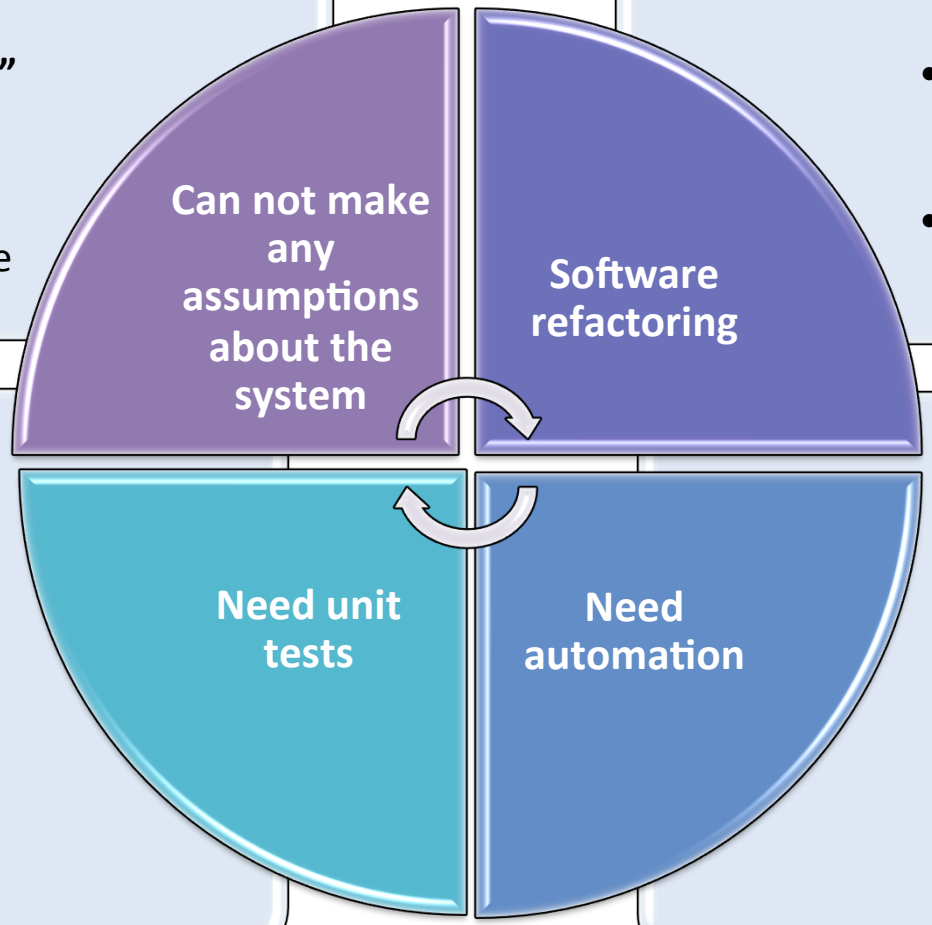
Time

Milestones: Lifemapper Server Roll



Issues Encountered

- **What are SW installation defaults and configuration**
- **Application "minimal resources" requirements** (i.e. what memory/disk/network/other is needed for application to work) changed with different hardware and data



- **Large changes in the build process**
- **Differences** in dependencies and configuration between Linux flavors: Ubuntu vs. CentOS
- **Simplify, harden and streamline code**

- **For build and post-install stages** for installed components and their configuration during the.
- **Datasets** to emulate application's run-time workflow

- Scripts for hands-on tasks (previously done only once):
- Software build
- Data population
- Configuration
- Monitoring
- Troubleshooting

Future Work

- Data integration – generalize more to use heterogeneous datasets from different sources
- Cloud approach – Extend and leverage with PCC and PRAGMA_Boot
- Virtualized environment testing
  - Storage system
  - Performance estimates
  - What I/O volume can we handle without degraded performance?
  - Check for "storage I/O blender effect"
- Need to define requirements for
  - performance
  - data storage
  - data management
- Need to "translate" application/data requirements when application is moved from physical to virtual servers
  - Is there a bottleneck?
  - What is needed for IO-intensive application (database)?

Acknowledgements

This work is funded in part by National Science Foundation and NASA grants

PRAGMA

US NSF 1234953

Lifemapper

US NSF EPSCoR 0553722

US NSF EPSCoR 0919443

US NSF EHR/DRL 0918590

US NSF BIO/DBI 0851290

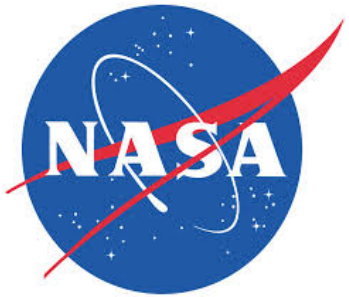
US NSF OCI/CI-TEAM 0753336

US NASA NNX12AF45A

Rocks

US NSF OCI-1032778

US NSF OCI-0721623



Questions?

Aimee Stewart (KU) astewart@ku.edu
Nadya Williams (UCSD) nadya@sdsc.edu