**Biodiversity Expedition – Lifemapper**

Biodiversity research uses species distribution modeling to understand how changes in climate and environment could affect the living organisms and how different species geographical distribution can change.

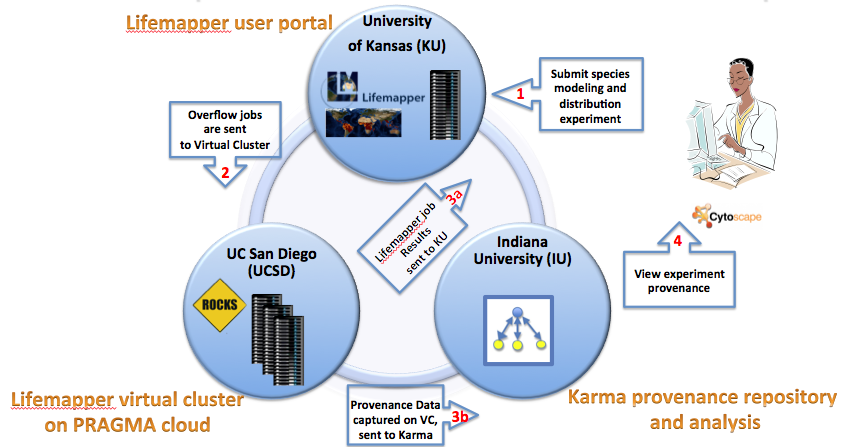
Lifemapper has been developed by the KU Biodiversity Institute since 2000 and today it provides a collection of computational and web services that together with the software modules enable species distribution modeling and mapping over geographic ranges (<http://news.ku.edu/2014/06/27/lifemapper-shows-where-earths-organisms-live-today-and-might-go-tomorrow>).

Lifemapper is a computer infrastructure producing an archive of species distribution maps calculated from public specimen data and a suite of data and tools for biodiversity researchers. Lifemapper analysis and modeling tools include *Species Distribution Modeling* (LmSDM) which tries to predict the geographic distribution of species based on where they have been found and the environmental conditions in those locations and *Range and Diversity Modeling* (LmRAD) that supports Macro-Ecological modeling.

The infrastructure consists of four independent sub-systems that communicate with each other to process biological data: 1) LmDbServer for data management; 2) LmCompute for computations; 3) LmWebServer for communications and 4) client applications.

For PRAGMA 25, three PRAGMA members (KU, IU and UCSD) built a framework for enabling Virtual Lifemapper Provenance Cluster (VLPC). The VLPC was instantiated at UCSD site, run Lifemapper species distribution modeling jobs created on the KU server and sent the jobs’ provenance information to the Provenance Karma server at UI site. For this distributed site approach we have: (1) streamlined and automated the build process for the LmCompute component to allow easy deployment of a virtual cluster at SDSC; (2) extended the Lifemapper code on the cluster to produce process-specific provenance logging; (3) enabled automated provenance collection for Lifemapper running jobs on SDSC's cluster.

Using this framework we can enable execution of Lifemapper jobs as seen on Figure1:



**Figure: Lifemapper Jobs**

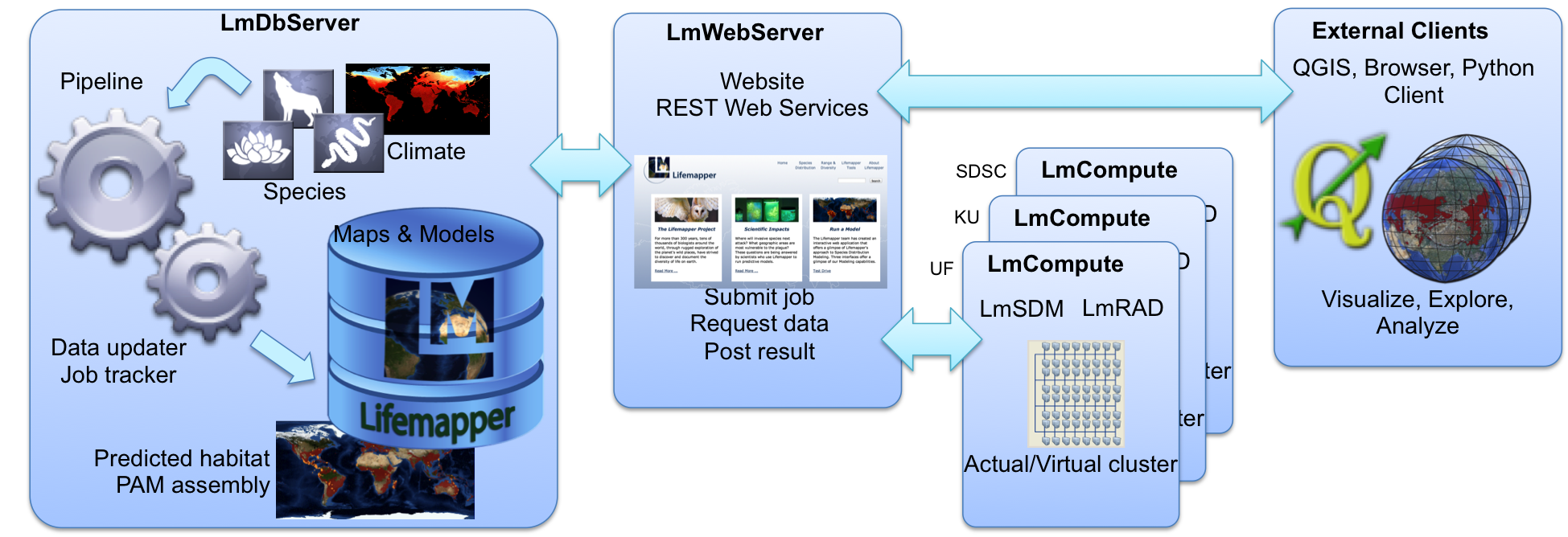
1. A user submits the species modeling and distribution experiment job using KU web services server.
2. A virtual cluster with installed LmCompute and provenance collection modules gets the overflow jobs from the KU server.
3. After the Lifemapper job is finished on VLPC, the job results are sent back to KU server and the provenance events are sent to the Karma provenance server at IU for ingest.
4. The provenance collection framework provides users with a provenance-enabled Cytoscape visualization plugin to access and visualize the provenance information stored in the Karma server. The purpose of provenance in this project is explanatory visualization to increase understanding how scientific applications use distributed computing resources, and provide the provenance about data lineage and data quality information to verify data reliability and improve data reusability.

The designed virtual environment must be capable of fast deployment and reproducibility. Virtual clusters can be easily instantiated as needed and can run Lifemapper jobs once KU server authorizes the cluster. We can build new clusters with the desired software environment or we can instantiate new clusters from a single authored cluster image using automated virtual cluster deployment process “pragma\_boot”. Deploying a ready-made cluster images as needed at multiple sites or at different hosting environments can provide scientists with on-demand availability of cluster resources.

For PRAGMA 26 we expanded on our previous work and virtualized the data management (LmDbServer ) and web services (LmWebServer ) sub-systems of Lifemapper. The broader purpose of this project is to increase the scalability, availability and flexibility of Lifemapper to enable scientists to assemble multi-species macro-ecology experiments or perform other LM-facilitated data processing on unique datasets, restricted-use data, very large datasets, or other special-needs data.

We needed to decouple the web and db servers from the KU-specific implementation and to create configurable, deployable servers components that will enable a fast turnaround form a server software update to a server’s availability (see figure below). The following four steps were identified as needed to make the decoupling a reality:

1. Create the build process for the components, enabling an independent instance of the Lifemapper server;
2. Pre-populate the system with software parameters and static environmental data for species distribution modeling;
3. Begin the archive creation workflow by starting the Lifemapper data pipeline application with a list of accepted scientific names used for requesting species data;
4. Configure one or more LmCompute instances to look to this new server for job requests.



**Figure: Lifemapper components**

The goal is enable individual projects, groups and scientists to run independent instancesof one or more Lifemapper components. Having an ability to start one’s own instance of Lifemapper will allow:

* + Lifemapper to scale for greater computational and data storage needs.
  + Scientists with specific data access conditions to control where their data is deployed and facilitate data processing of
    1. unique datasets,
    2. restricted use data,
    3. very large datasets.
  + Backup or failsafein case any tier needs maintenance or fails
  + Running Virtual Biodiversity Experiments in the field at the time of data collection.

During the code base refactoring we discovered that pipeline job creation on the server had inconsistencies in the data and metadata creation, cataloging, and updating. We were able to find and address most of the inconsistencies in the deployed installation, successfully seeding the database but we were not able to restart the data pipeline and had to defer the remaining few fixes. In summary, we succeeded with 3 out of 4 our servers decoupling goals:

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

The final goal would have brought everything together and started creating populating species data from GBIF, then computing SDM maps for all these species. This illustrates the complexity of the virtualizing of real science application.

We will work on finishing the pipeline work and finalizing the Lifemapper servers virtual cluster images.

We plan to use heterogeneous datasets from different sources and extend cluster deployment at different sites using pragma\_boot, and test other deployment options such as EC2 and VirtualBox.