RDA-PRAGMA Sprint
-- PRAGMA 30 @ Philippines

Quan (Gabriel) Zhou, Nadya Williams, Aimee Stewart
Jason Haga, Beth Plale

2/4/16

Objectives

- Assess recently released tools and best practices from RDA for contribution to PRAGMA services. Carry out assessment through 2 phase demo.
- Demo: verify lineage of projection data objects, and enable rerun when new data exists
- Enhancements to PRAGMA testbed: Provide common persistent identifiers and landing pages to VMs and datasets of Lifemapper
- Feed results back to RDA

Demo 1 principles and constraints

- Minimal metadata for objects (VM images, data) associated with persistent ID so that when persistent ID is found, its metadata can be consulted (and understood) efficiently to make decisions.
- Use of Persistent IDs for objects (software, data).
- Utilize RDA tools (Data Type Registry and PID Information Types service) in demo to evaluate benefit for PRAGMA community.
- Get data services architecture in place
- Constrained for one software object type, one output type, and one application

Demo Phases

- Phase 1 (Jan 2016, PRAGMA 30 Manila)
  - Use static GBIF subset for Southeast Asia as input to Lifemapper,
  - Input datasets bundled into VM.
  - User has ID of two projection result datasets (both result sets have same internal ID (e.g., 317)), and uses RDA services to determine whether they came from the same VM or from the primary VM and its clone
- Phase 1.5 (Mar 01, 2016, RDA P7 Tokyo):
  - Define minimal metadata needed for software objects and data objects.
    - Minimal metadata must be sufficient to distinguish one VM from the other, and one projection result from the other
  - Define type definition for both minimal metadata definitions; register with Data Type Registry
  - Associate two properties with the PID (handle): URL to landing page and pointer to the type definition that describes the minimal metadata

Demo Phases

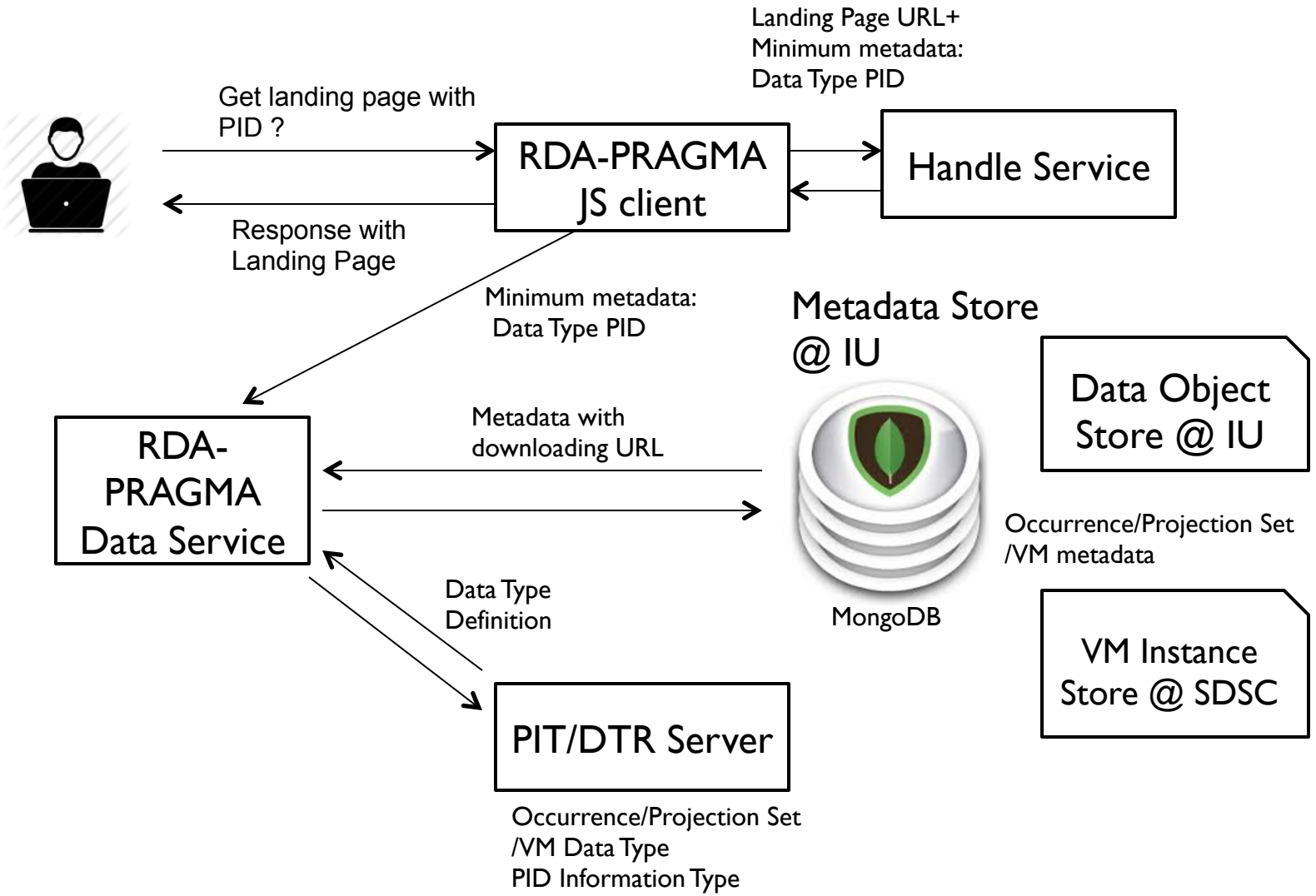
- Phase 2 (Sep 2016, PRAGMA 31):
  - Demo: After seeing change to iDigBio input dataset, use new PRAGMA data infrastructure to identify, download, and faithfully replay run with new iDigBio input dataset to visually compare before and after.
  - Input datasets ingested dynamically into VM (workflow dynamically accesses iDigBio.)
  - Rocks roll for PRAGMA-RDA data service (includes data service, client, PID Information Types service, Data Type Registry service) but not handle service

Lifemapper VM ID scheme

- Lifemapper VM can be uniquely identified by the following 5 attributes
  - Host IP
  - Rocks version number
  - SpeciesDataset ID
  - EnvironmentalDataset ID
  - Github roll tag

- Proposed ID scheme: single common ID (UUID, DOI, handle) with attributes as part of the minimal metadata stored to RDA Persistent Information Type Service

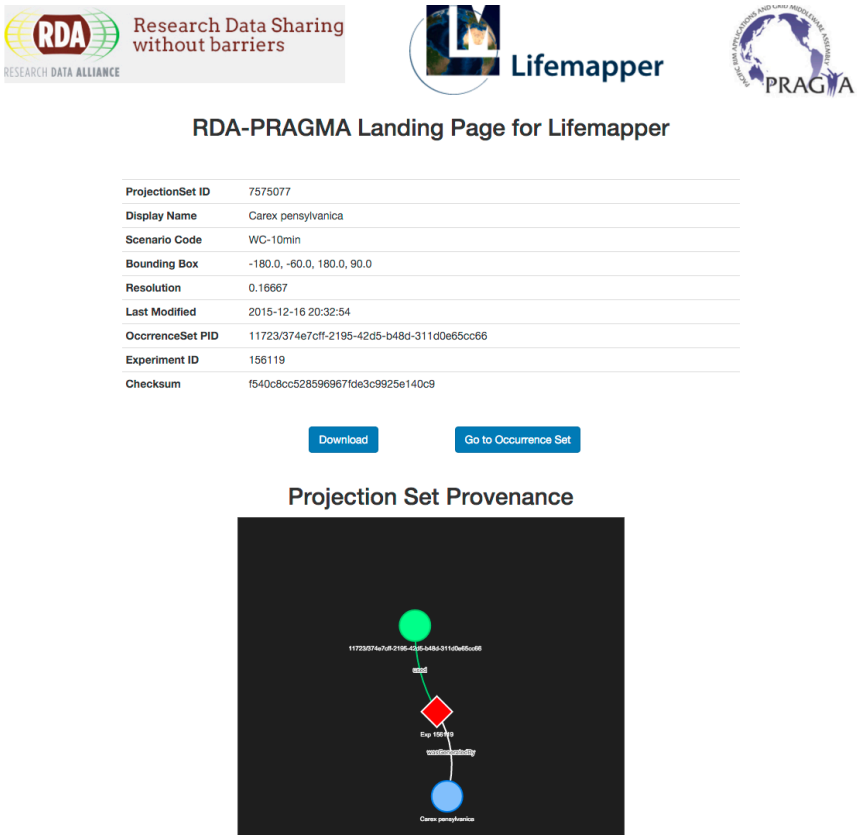
RDA-PRAGMA Data Service



New architectural components

- PRAGMA-RDA Data Service
  - Stores metadata, objects, and landing pages
  - Maintains metadata about both data sets and VMs
  - Assigns unique handle to incoming objects
  - Displays landing page for each object
  - Interacts with RDA PIT/DTR service
- RDA PIT/DTR service: stores type information about minimal metadata that allows interpretation of the metadata.
- Handle service : obtain/resolve handle PID (at CNRI)

Landing Page Example

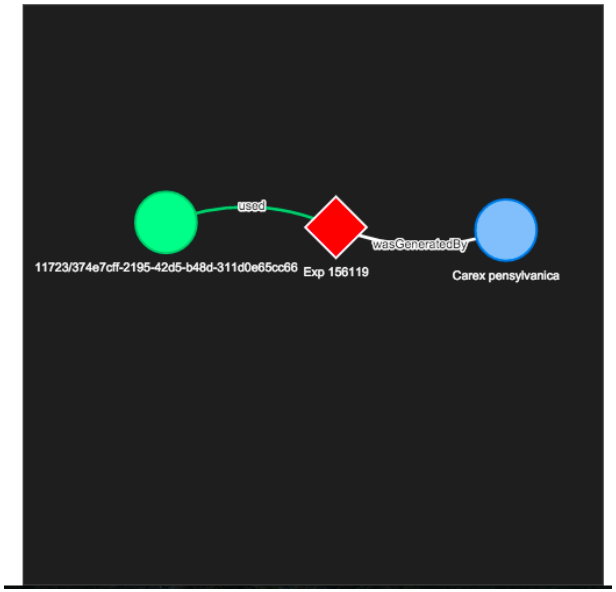


Demo URL:
http://hdl.handle.net/11723/d506d6e9-54f8-4c5c-9e95-054a26db24d1
Code base example:
https://github.com/Gabriel-Zhou/LMLandingpage

Provenance Role

- Reveal lineage between projection set and occurrence set in the context of Lifemapper experiment

- Provide reproducibility of Lifemapper experiment and determine trustworthiness of experiment output



Data Diff Service

Data Validation Service

Projection PID	11723/99f2c886-5a20-42e0-9220-e6a771f4e940

Validation VM PID	11723/24cf5abe-9beb-4994-81d2-3b7b7b45478b

Submit

VALID !

Validation Result:

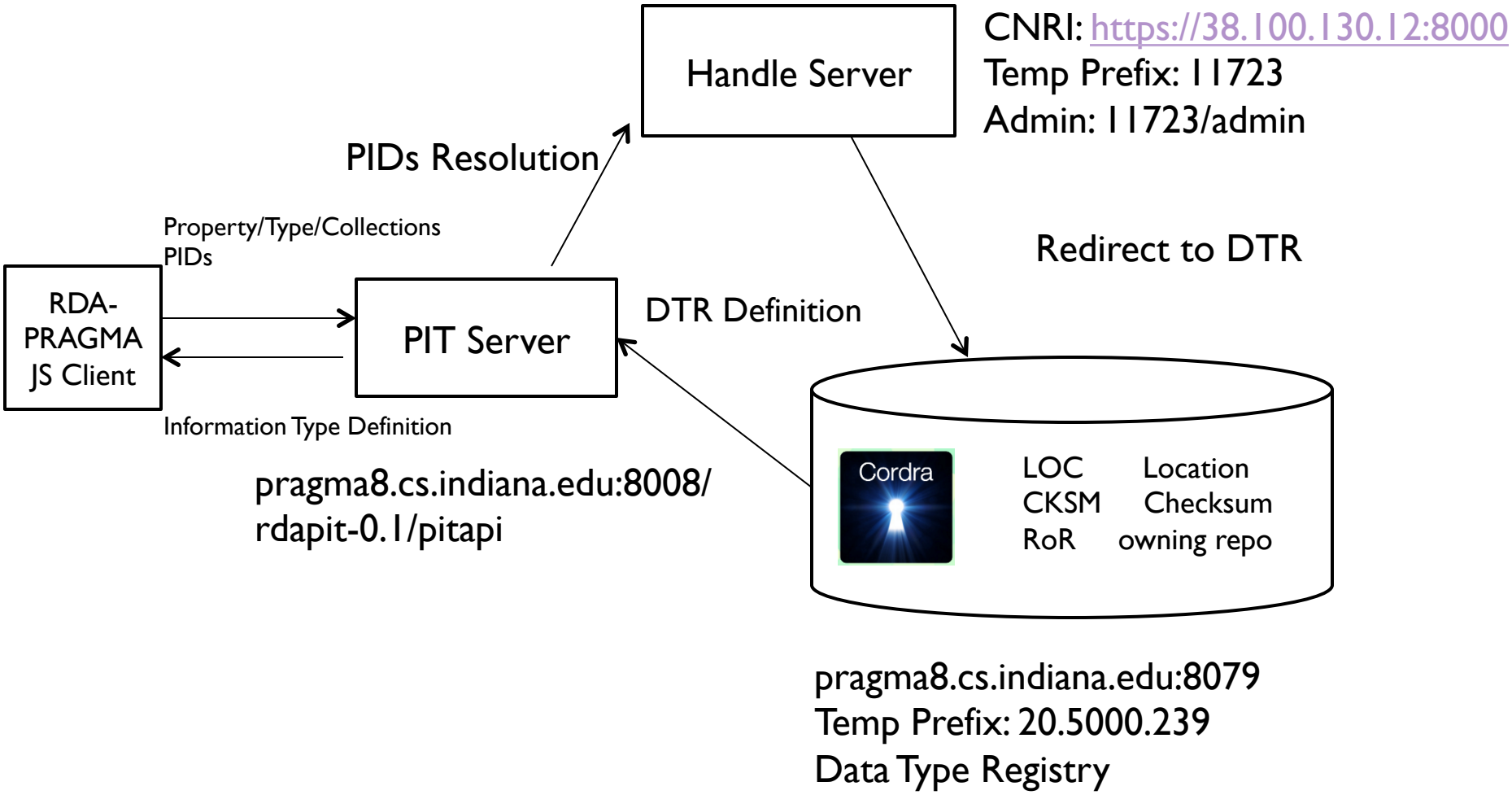
```
Retrieving Lifemapper VM information type...
Retrieving original VM of target projection set:11723/839e2528-0f79-4205-9022-3329302a1d14
Comparing validation and original VM metadata with LM VM information type...
Original VM manifest metadata:
rocks Version:6.2
Species Dataset:sorted_seasia_gbif
Environment Dataset:30sec-present-future-SEA
Roll Version:p29_1; LM Version:1.0.3.1w

Validation VM manifest metadata:
```

Information Type @ RDA PIT/DTR WG

- RDA PIT working group allows providers agree on a common API, register their information types in a common data type registry and agree on some core types;

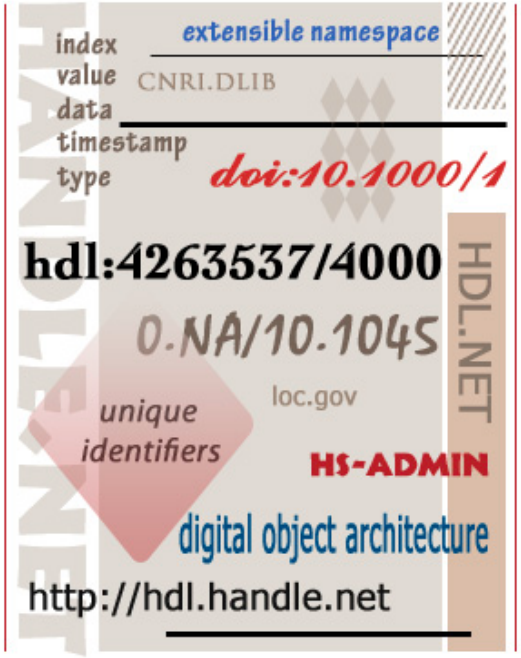
- We deployed the PIT/DTR services on PRAGMA IU nodes and registered information types of Lifemapper generated (projection) sets and Lifemapper VM instances with useful metadata units.

RDA PIT/DTR Architecture



Handle Server Configurations

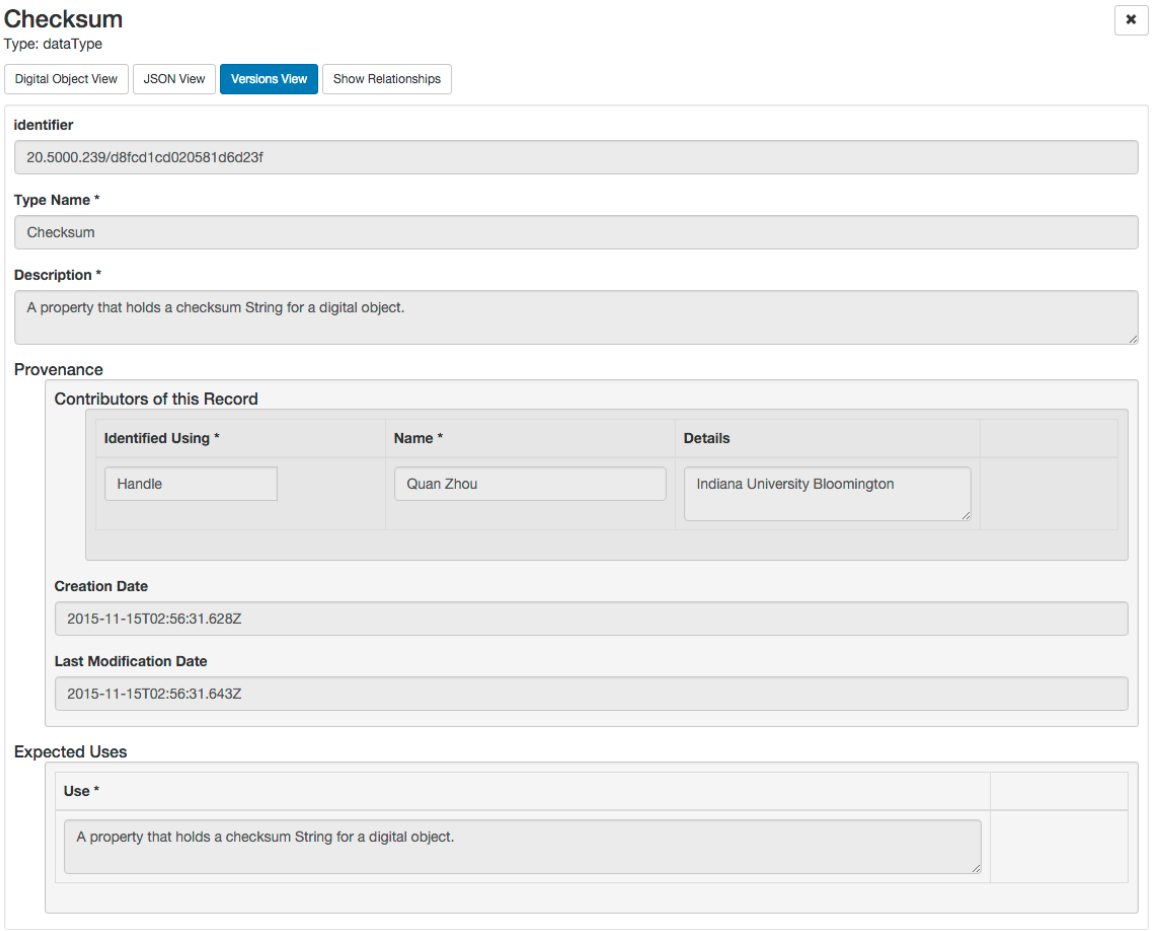
- CNRI hosted a handle server V8 instance for our evaluation;



- Handle instance configurations:
  - https://38.100.130.12:8000/
  - Handle prefix: 11723

CheckSum Information Type

PID: 11723/377739b4-14df-441a-b219-15881cf6ae52



More Information

- For more information, please visit the following URLs:
  - RDA PID Information Types Working Group
    https://rd-alliance.org/groups/pid-information-types-wg.html
  - CNRI Handle.Net Registry
    https://www.handle.net/
  - Lifemapper
    http://lifemapper.org/

IU Data Node Resources

- RDA PIT/DataType Registry Service
- PRAGMA-RDA Data Service
- Handle V8 service (generates PIDs, runs at CNRI)
- PRAGMA-ENT Mesh
- Open HathiTrust Corpus

Acknowledgement

- This project is funded by PRAGMA. (NSF OCI 1234983)

- We thank CNRI for hosting handle V8 server for evaluation PIT/DTR tool. We thank Tobias Weigel from RDA for all the instructions and discussions about RDA output.