The Research Database as a Service

Daniel Tosello
Research Data Analyst
La Trobe University eResearch
July 2013

The Research Database as a Service

- Rationale for development, Setting the Scene
  - Why did we decide to build this system?
- How does it relate to Cloud systems?
  - Metaphors, Terminology and Paradigms
  - The As A Service model
- How is it different?
  - The traditional research database
  - On demand services
  - The eResearch Ecosystem
- How to develop your own
  - Components, Workflows and Technical details

The Research Database as a Service

At La Trobe University eResearch, we are on a mission

It’s not just about databases
It’s not just about metadata

Rationale for development

From the eResearch Perspective:

- Challenges in Data Management
  - Changing requirements in grant funding
  - ARC & NHMRC
- Resource limitations
  - Limited time to invest in development
  - Some development virtually necessary to provide coherent functionality
- Long Tail & Silo data
- Consideration of Researcher priorities
- Meeting common needs

Rationale for development

The eResearch Perspective – continued

- Acquiring passive records relies on initial input from researchers
- Maintaining active data relies on *ongoing* input from researchers
- Collating results requires them to be available
- Flexibility of application
  - Different types of projects
  - Web Applications
  - Generic structured data collections
  - Old, Long tail or siloed datasets

Rationale for development

From the Researcher Perspective:

- A great variation of requirements
  - Disciplinary level
  - Project level
- Tools requirements must be met
- Sharing and collaboration of data is both desirable and a long standing challenge
  - Often peers are not at the local institution
  - Security and capacity issues
- Time limited
- New data management funding requirements are creating a need

Rationale for development

What Researchers actually want

- Collaboration
- Sharing already collected, siloed data
- Collaboration
- Convenience
- Collaboration

Rationale for development

What Researchers definitely don't want

- Extra overhead
- Stress

Rationale for development

With that in mind, ideally…

- The system should track Researcher's metadata for them
  - Researchers shouldn't have to spend so long entering metadata
  - Metadata should always be sourced automatically where possible
  - Metadata should be automatically distributed to relevant repositories for curation
- Integrate technology into workflows without visibly modifying them
- Preference application platforms with accessible APIs
- Maintain a group capable of performing implementation, updates and modifications as necessary

Setting the Scene

At La Trobe –

- Underlying ICT infrastructure and resources
  - Active Directory
  - Traditional Storage (NAS, SAN)
  - Database Administrators
  - Enterprise database software
  - Enterprise Applications
  - Process oriented, potentially lacking agility

Setting the Scene

At La Trobe –

- Existing data management infrastructure
  - Fedora (VTLS)
  - ReDBox
  - Library Data Curation Team
  - Early attempts to integrate data management

Setting the Scene

No singular system could meet the needs of ALL researchers.

However, the required functionality can be provided by a combination of different packages linked to a central authorization index

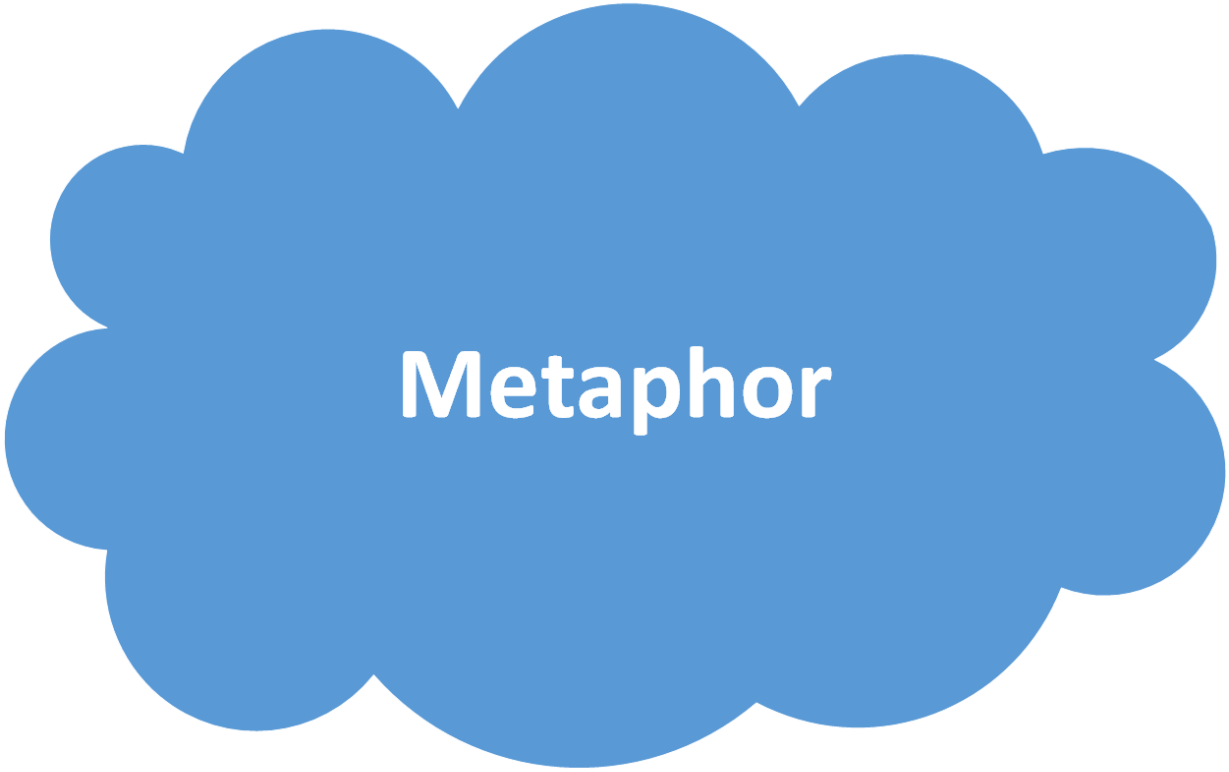
Setting the Scene

No singular system could meet the needs of ALL researchers.

However, the required functionality can be provided by a combination of different packages linked to a central authorization index

But as an aside…

The Cloud



The Cloud (What it is)

- Obfuscation as a management technique
  - Black box philosophy
  - APIs
  - A common functional approach in software
- Extending this paradigm to hardware
  - Platform independence
  - Distributed storage, redundancy
  - Allows for massive automation of parallel processes
  - Immensely powerful for some tasks when managed properly

The Cloud (…and what it isn't)

- Does not reduce the need for computing power
  - Marginally increased requirements and overhead
  - Potential for optimization
- Abstracts storage, but does not reduce reliance on physical media
- Does not magically resolve latency issues
  - Data must be physically near to compute
  - Geographic distances can be a challenge

The Cloud (…and what it isn't)

- Not actually logistically simpler
  - Logistic load shifted to cloud provider
  - Logistic capacity *also* shifted to cloud provider
  - *Disastrous* performance and reliability if complex demands aren't met
- Capacities shouldn't be taken for granted in research oriented organizations
  - Providing good services gives researchers an edge

The Cloud (…and what it isn't)

So the cloud isn't the answer?

Back to the situation at hand…

Some problems worth solving.

- Databases require ongoing maintenance & support
- Administration must be performed on the whole stack
- ICT reluctant to allow free reign on managed systems, with fair reason
- Hardware is expensive and requires administration too

All these factors represent significant barriers to the average researcher

How is it different?

Traditional databases

- Run on monolithic database servers
- Guarded fiercely by ICT
  - Secure
  - Potentially hard to access legitimately
  - By extension, potentially hard to use for collaboration
  - ICT reluctant to engage with content

How is it different?

Databases in the Software as a Service paradigm

- Run on distributed systems owned by large entities
- Out in the world
  - No direct control over physical relationship between data and compute
- Can be truly secure but only with additional overhead
  - Node to Node encryption
  - Extra resources for authorization management

How is it different?

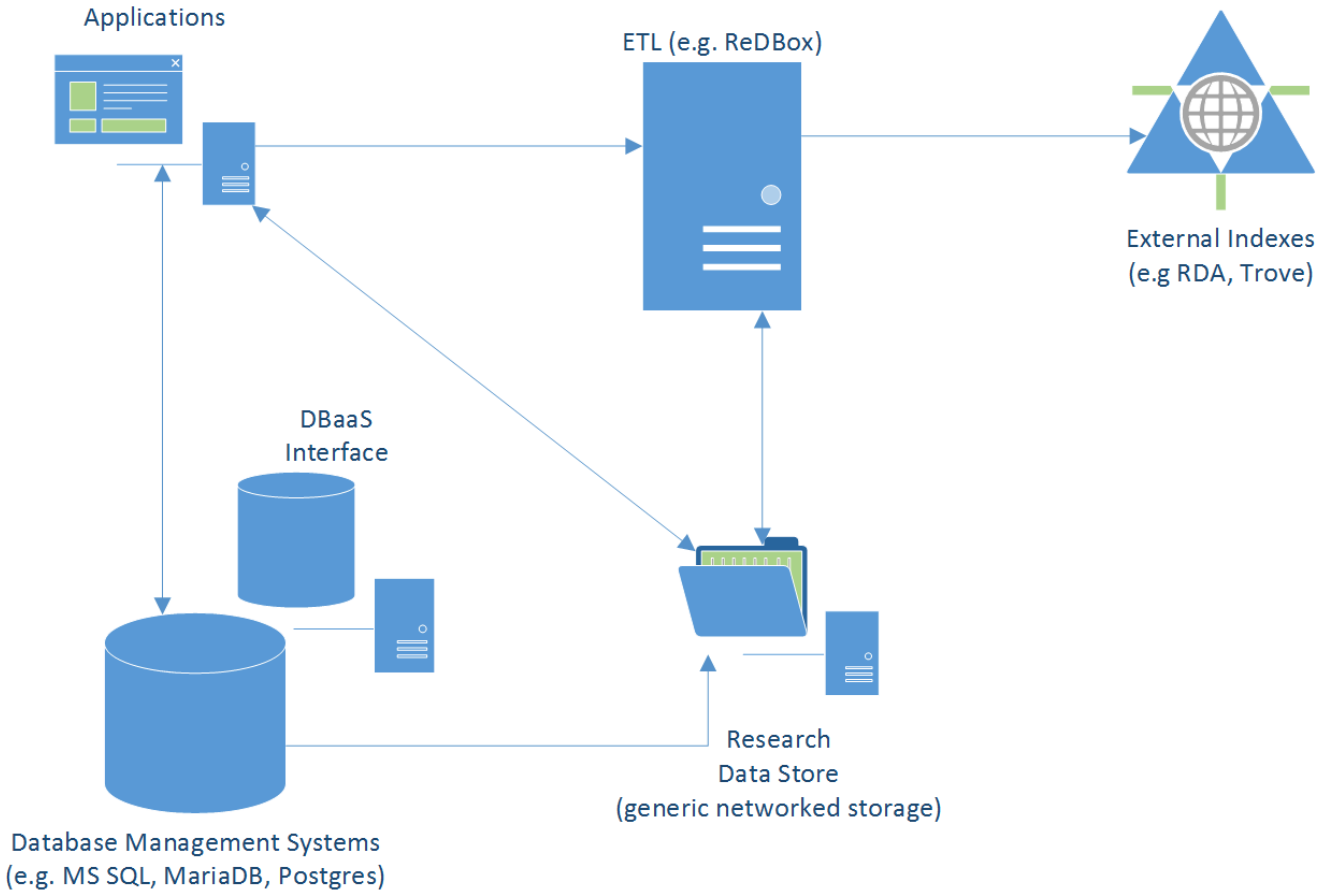
Our Implementation – The best of both worlds

- Surprisingly simple
- Using only common open source or enterprise supported software
- Conceptually separate responsibility for structure from content
- Using as much existing infrastructure and support as possible
  - Authentication methods
  - Structured storage and current DBA workflows
  - Existing applications
  - Institutional Metadata
  - No change to previous server or database administration procedure

The Jigsaw Puzzle

- Task oriented endpoints
  - Authentication (AD/LDAP)
  - MySQL/MariaDB, MS SQL, PostgreSQL databases
  - HTML based interfaces
  - Fedora/MyTardis repositories
  - ReDBox for metadata processing and transport
  - Web Applications
  - Research software packages

How is it different?



How is it different?

In a sense, it’s NOT that different

How is it different?

In a sense, it's NOT that different

It's simply a way of assembling existing technologies to mitigate some of the barriers to use and administrative overhead

How is it different?

Therefore, you can do it to!

How to develop your own

- Assess your current environment
  - Both hardware & software are relevant
  - Adapt where possible
- Assess your researchers' specialist needs
  - Dedicated software packages
  - Existing workflows
- Assess the strengths of your current team
  - Work with familiar technology where possible
  - Focus on outcomes
  - Try to avoid investing in excessively niche software unless necessary
  - Attempt to make relevant data available to other applications

How to develop your own

Systems to fill universal roles – Database Storage

- Structured Storage
  - MySQL/MariaDB
  - PostgreSQL
  - Microsoft SQL Server
- Unstructured Storage
  - SMB/CIFS
  - WebDAV
  - HTTP based Dropbox-style system (e.g. CloudStor+, OwnCloud)

How to develop your own

Systems to fill universal roles – Sources of Truth

- Authentication and Authorization
  - Institutional Authentication is *always* preferable where possible
  - Specifics of attaching applications and systems should be available
- Researcher Metadata
  - Institutional Repositories
- Dataset Metadata
  - A potential problem, Some local schema needed
  - Requirements will vary based on project types

How to develop your own

Systems to fill universal roles – Side note on database permissions

- Ensure that applications have unique database users
  - Safety & Security
  - Access monitoring

How to develop your own

Systems to fill universal roles – Methods of Interaction

- User Interface
  - HTML based interfaces
  - Datastreams
- Application Interfaces
  - ETL (e.g. ReDBox, Pentaho Kettle)
    - This is where the bulk of the work lies
    - Schema transformations

Thank you