

# **Beyond Annotea**

*Stephen Crawley, Ron Chernich, Jane Hunter  
eResearch Group,  
School of ITEE, Univ. of Queensland*

# Acknowledgements

This work is part of a National eResearch Architecture Taskforce (NeAT) project, supported by the Australian National Data Service (ANDS) through the Education Investment Fund (EIF) Super Science Initiative, and the Australian Research Collaboration Service (ARCS) through the National Collaborative Research Infrastructure Strategy Program.

Many thanks to ALA and Aus-e-lit for motivating the work, and using our software in their sites.

# Overview

- Annotea
- Implementing Danno & Dannotate
- Lessons learned
- Scaling up

# Annotea Basics

- Protocol for managing annotations on web resources
  - HTTP based, RESTful.
  - Annotations/Replies represented in RDF with simple/generic RDF schemas.
  - Simple queries to fetch annotations for resource, replies for annotation.
  - GET, POST, PUT and DELETE for CRUD.
  - Annotations are separate from resources.

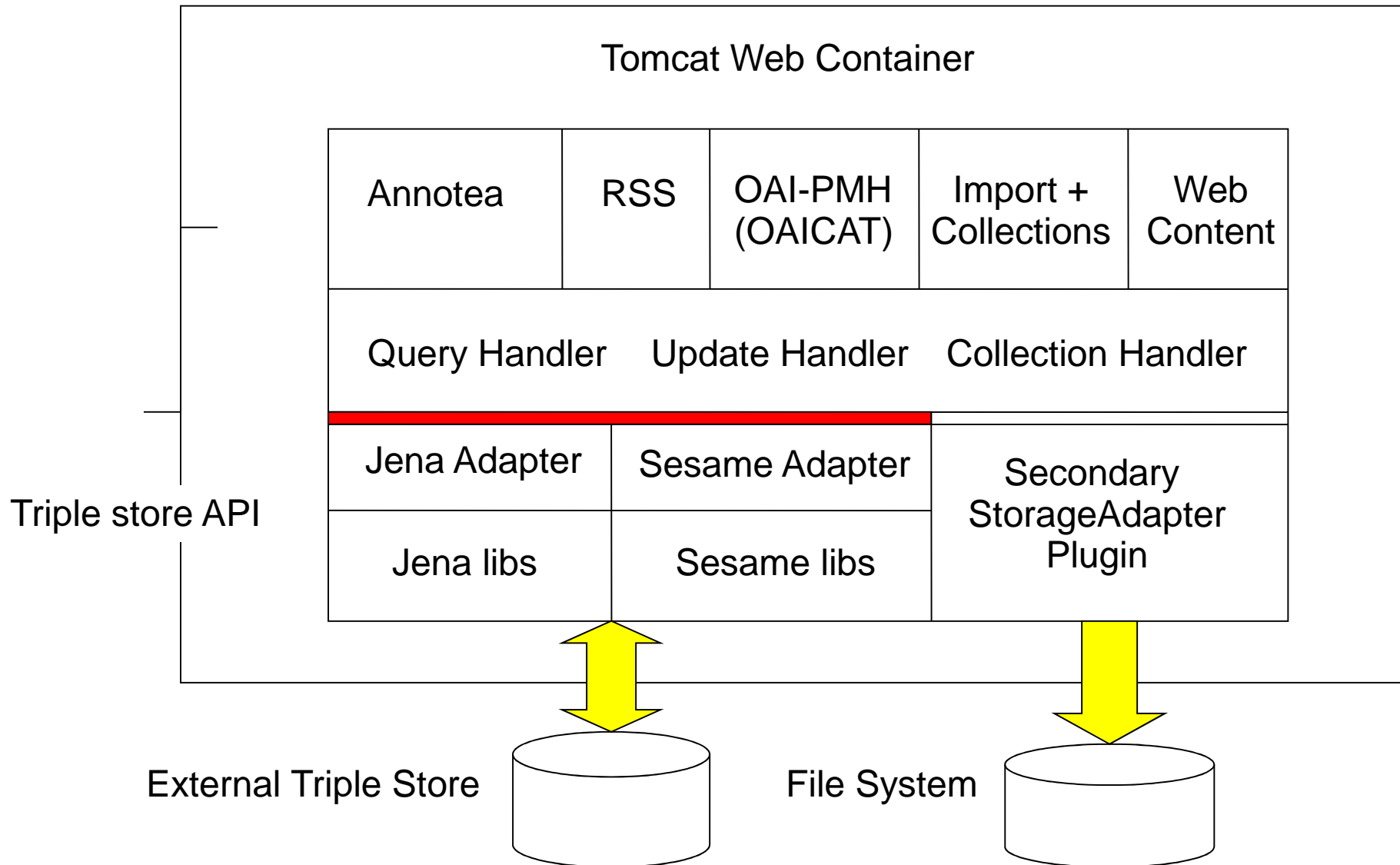
# Annotea History & Status

- Produced by W3C working group, circa 2000
- Documents are draft only.
  - Not endorsed standards.
  - Not standard track.
  - Objectively not “standard quality”.
- No current Annotea standardization effort.
- Implementations exist, but no major products / websites support Annotea.

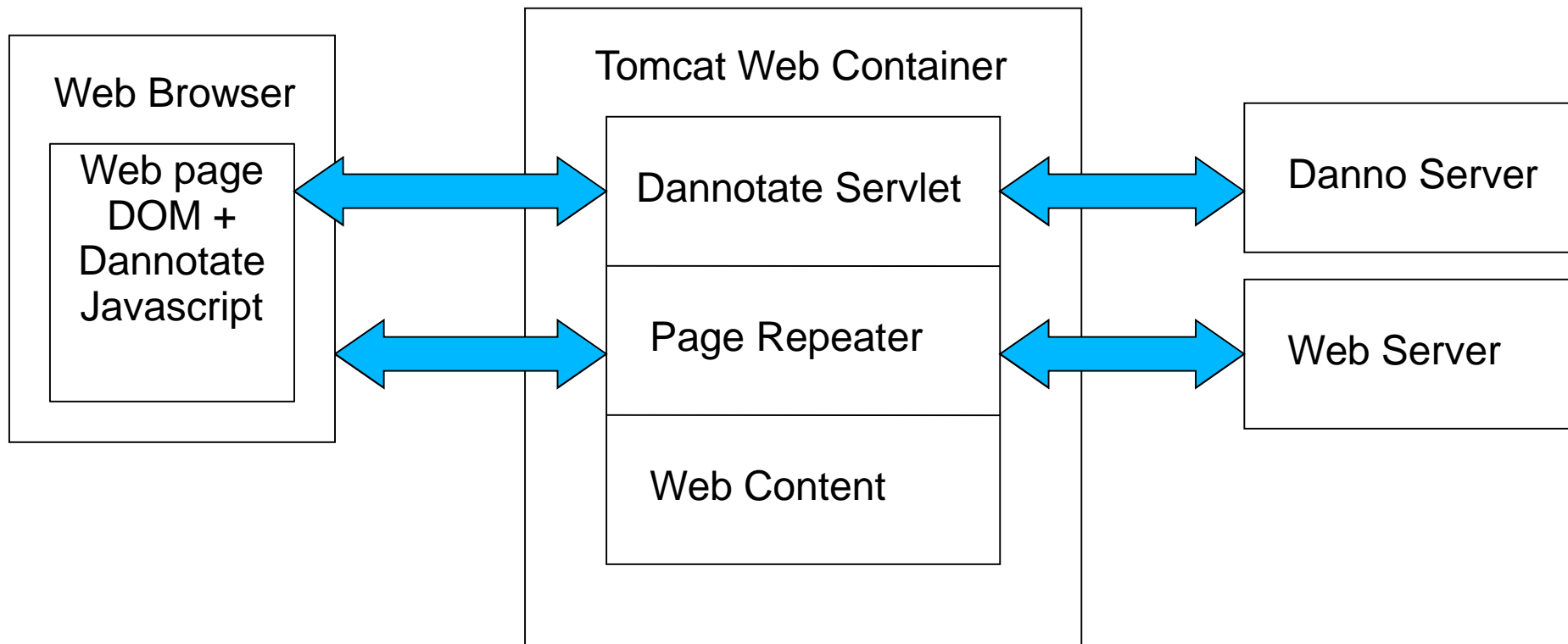
# Danno & Dannotate

- Danno annotation server:
  - Implements Annotea protocol + extensions.
  - Supports fine-grained access control, collection management, OAI-PMH harvest, etc.
  - Designed to be highly configurable, extensible.
- Dannotate annotation client:
  - Runs in multiple browser (not IE7 or earlier)
  - Not a browser plugin / addon – no user install step.
  - Text, image & map annotations.
  - Context sensitive annotation, annotation of data.

# Danno Architecture



# Dannotate Architecture



# Danno / Dannotate in use

- Aus-e-lit project.
  - Uses Danno with custom annotation tool.
  - Free-text indexing of annotations.
- Atlas of Living Australia.
  - Uses Danno. Will use Dannotate.
- 3-D model Annotation project.
  - Uses Danno with custom annotation tool.
- AAF-enable Annotation project.
  - Will use Danno + Dannotate + ...

# Lessons – Annotea issues

- General:
  - Failure to create a standard.
  - Too much “specification by example”.
  - Specifications should be based on real experience.
- Specific:
  - No specification of the annotation model.
  - Dubious handling of annotation “bodies”.
  - Sketchy RDF schemas. Poor separation.
  - Security / access control not addressed.

# Is OAC the answer?

- Open Annotations Collaboration (OAC).
- Superior Annotation model:
  - Multiple annotation targets, contents.
  - Improved contexts: fragments, constrained targets.
  - Annotation versions, time-related annotations.
  - <http://www.openannotation.org/spec/alpha3/>
- Missing from OAC (so far):
  - Client-server protocol for annotation tools.
  - Framework for application-specific annotations.
  - Standards track.

# Lessons - Danno

- No SPARQL equivalent for updating
- No standard Java APIs for RDF handling.
  - Sesame, Jena, etc all have different APIs.
  - Vendor tie-in or abstraction layer.
- Triple store performance issues.
  - Sesame, Jena are slow.
  - Proprietary implementations faster, but expensive.
  - Performance of closure operations.
  - Connection pool / transaction impl issues.

# Lessons - Dannotate

- Avoiding browser plugin / extension requires significant server-side support.
  - Same source problem.
  - Page repeater approach is fragile.
- Cross-platform Javascript is hard.
- Map annotations are hard, Map API specific.
- Live updates are a scalability issue.

# The Scaling Problem

- Danno handles 100 requests per second max\*.
  - 300 users x 10 browser tabs with live-updates\*\* ...
  - ... “Huston, we have a problem”.

\* Indicative only. This is from an old benchmark with no security, a small triplestore and no Dannotate.

\*\* Assume live-updates poll once every 30 seconds.)

- Scaling dimensions:
  - Number of annotated targets, annotations per target, replies per annotation
  - Number of concurrently active users
  - Annotation create/update/delete rates

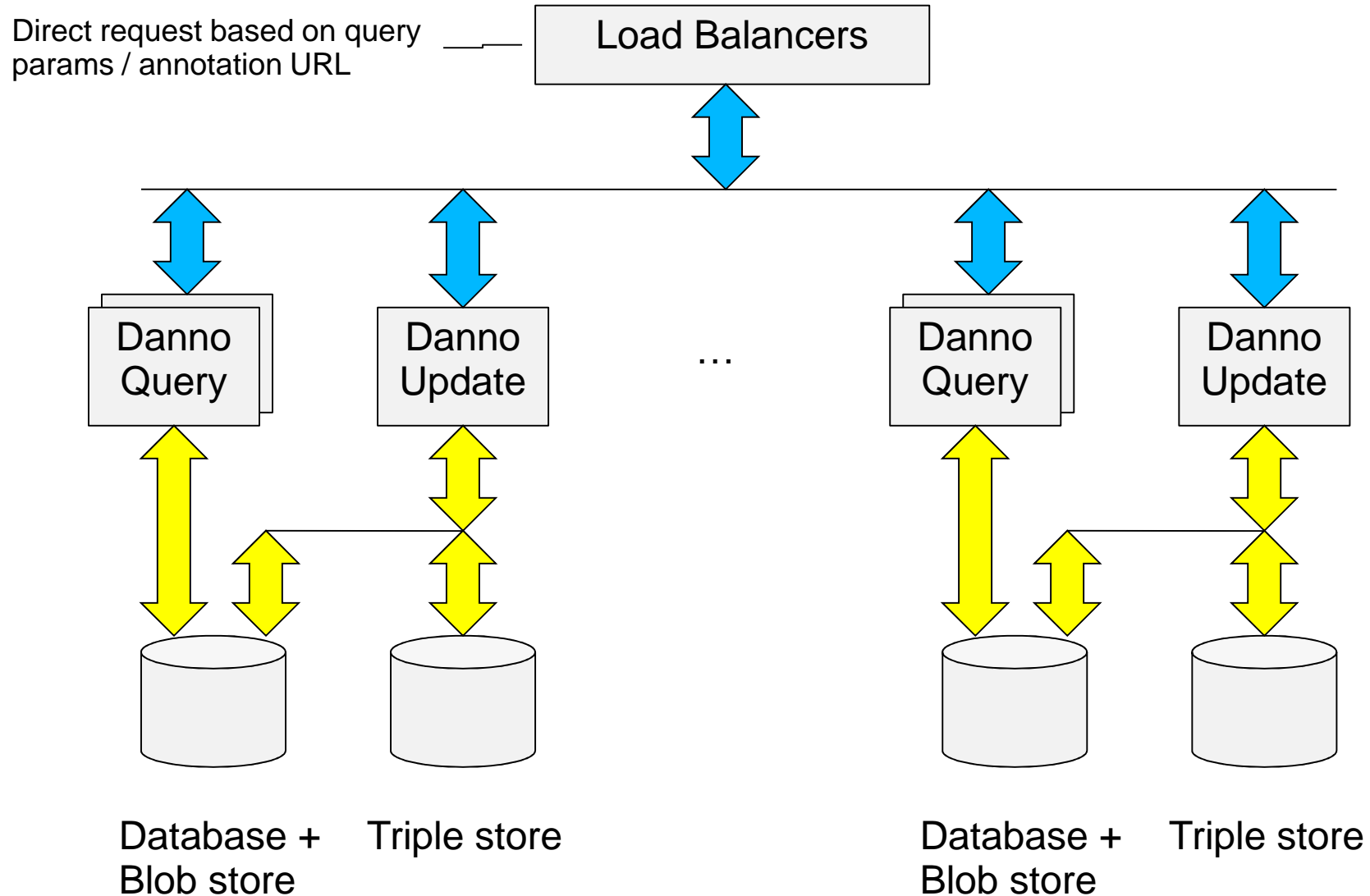
# Danno Performance

- Replace persistence abstraction layer.
  - Not RDF-centric. Not Annotea-centric.
- Use blob store to hold RDF/XML annotations:
  - Use triple store or multi-store implementations in deployments where inference is a requirement.
- Add custom SQL tables to support Annotea queries and access control:
  - Tables & indexes for supported queries.
  - Table for annotation metadata.

# Danno Scaling

- Partition annotations across multiple Danno servers based on target URLs
  - Load balancer dispatches based on annotation URLs and query params (e.g. annotation target).
- Database / blob store for primary persistence.
- Further measures:
  - Replicate / mirror Danno's persistence backend.
  - Use in-memory caches for active locuses.

# Scaled-up Danno



# Dannotate Performance

- Reducing the number of interactions:
  - User's browser <-> Dannotate servlet
  - Dannotate servlet <-> Danno
- Adaptive live updates:
  - reduce polling rate when load is high.
- Use of COMET for live updates.

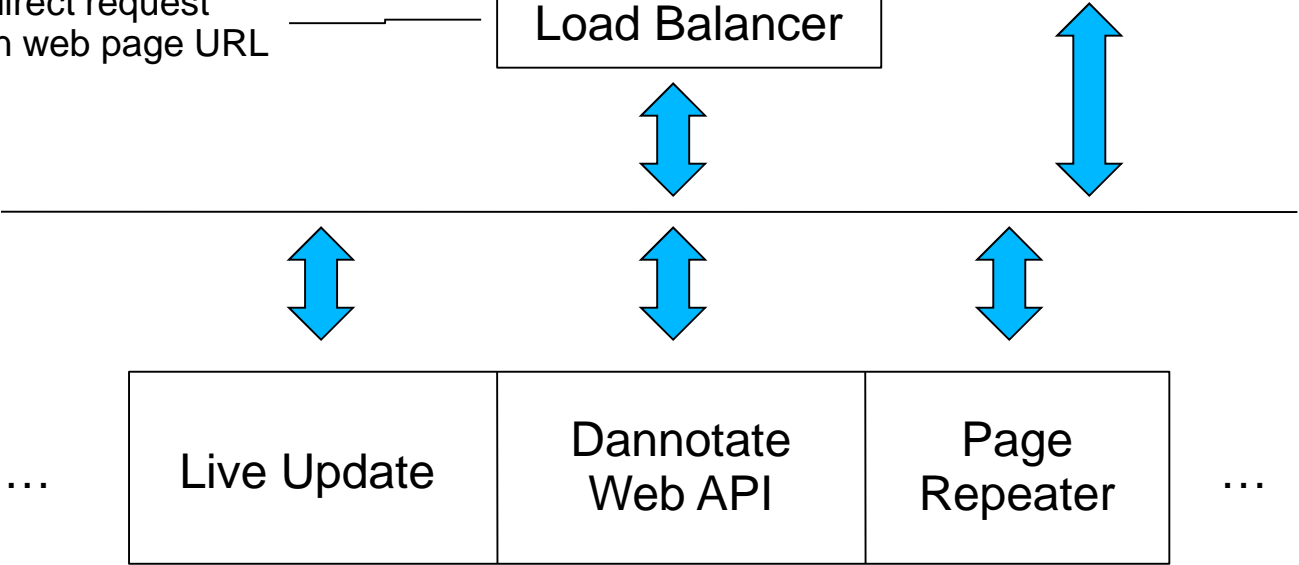
# Dannotate Scaling

- Danno / Dannotate servlet on different servers.
- Replicate servers running Dannotate servlet.
  - Load balancer dispatches to server based on client IP address.
- Use internal publish/subscribe network to eliminate live update load on Danno servers:
  - Danno servers publish update events for annotation / reply create, update and delete.
  - Dannotate servers subscribe to update events for relevant targets, annotations or replies.

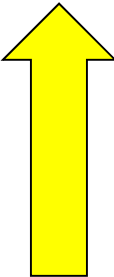
# Scaled-up Dannotate

Initially direct request  
based on web page URL

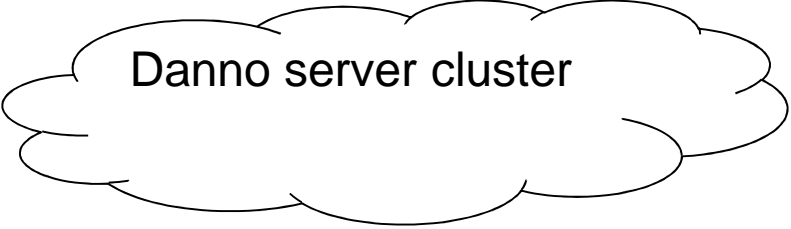
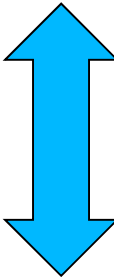
Load Balancer



Update events  
via pub/sub



Annotea+ requests



# Conclusions

- Only use a triple store if you **need** to.
  - Fixed queries + no inference => DB is faster.
- Annotea is problematic:
  - It is **not a standard**.
  - Specification incomplete + technical issues.
  - Nobody much implements it (see above)
- Open Annotation Colloboration
  - Shows considerable promise.
  - Needs to be standards tracked ... or else it will be ignored by industry. (My prediction!)