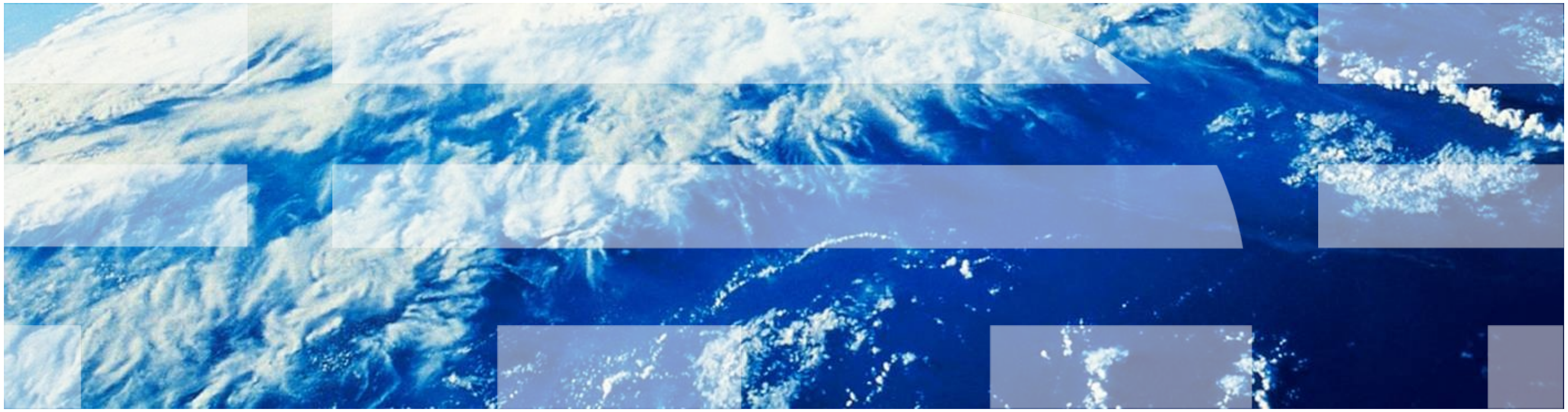
Using read/write Linked Data for Application Integration - Towards a Linked Data Basic Profile

Arnaud J Le Hors, IBM Linked Data Standards Lead



Agenda

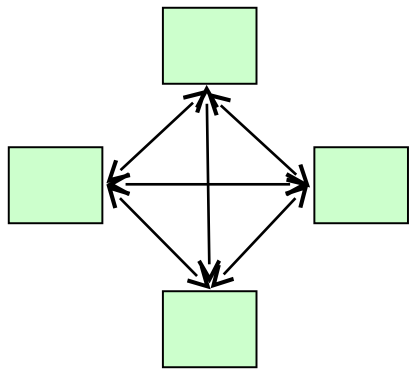
- IBM Rational's ALM Integration Challenge
- Linked Lifecycle Data
- Problems encountered using Linked Data
- A proposal: Linked Data Basic Profile
- Looking Forward

IBM Rational's Integration Challenge

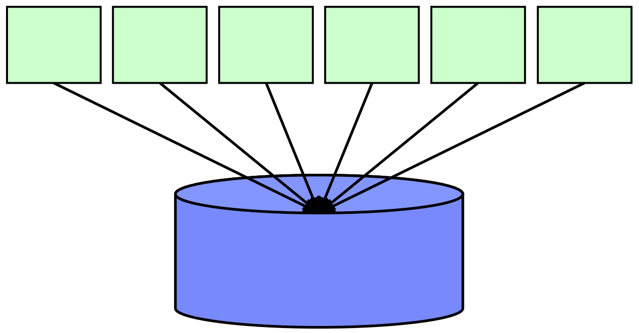
- Application Lifecycle Management (ALM) customers requirements:
  - More tightly integrated end-to-end ALM solution
  - Support for distributed development teams around the world
  - Broader solutions
  - Greater agility and lower cost of ownership

- Need to support integration between various tools from various parties
  - Source Code Management from Vendor A
  - Defect/Change Management from Vendor B
  - Requirements Management from Vendor C
  - ...
  - Homegrown components

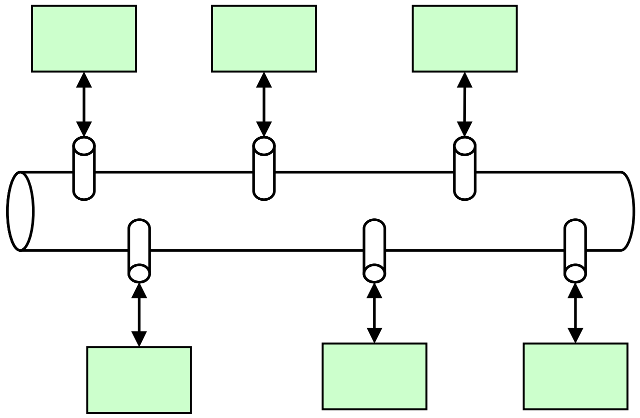
Traditional approaches not wholly satisfactory



- Point to point communication via APIs
  - Tight coupling
  - N^2
  - Requires lockstep versioning



- Central Repository
  - Requires schema coordination across tools
  - Requires "central planning" by consumer (deploy one database) as well as provider (integrate tools to a single schema)



- Central hub/bus
  - Multiple copies of data in different tools
  - Sensitive to changes in data model in any participating tool
  - Sensitive to changes in the process

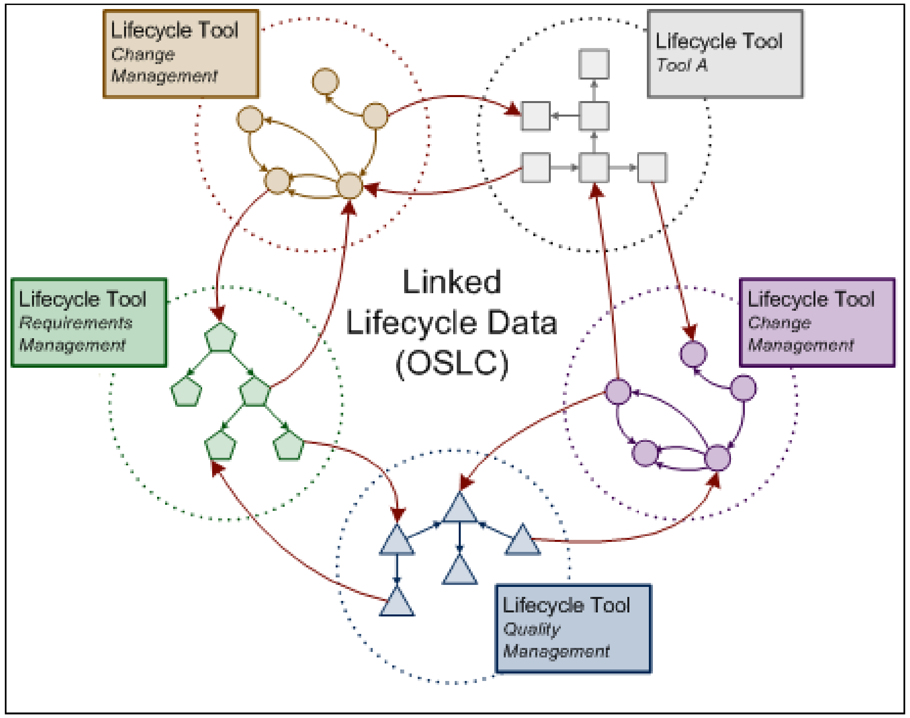
Characteristics of an Ideal Solution

- Distributed: to match today's work environment
- Scalable: to support an unlimited number of products and users
- Reliable: to work across a wide range of connectivity profiles
- Extensible: to work with a wide variety of resources
- Simple: to avoid fragility of tight coupling and keep the barrier to entry low
- Equitable: to be open to everyone

Solution: Linked Lifecycle Data

- **Inspired by Internet principles, implemented with Internet technologies**: simple interfaces for exchange of resources
- **Loosely coupled**: everything is a “resource” linked together with URLs
- **Technology neutral**: treats all implementations equally
- **Minimalist**: defines no more than necessary for exchange of resources
- **Incremental**: deliver value now, add more value over time
- **Openly published standards**: free to implement and irrevocable

If the entire Web can connect like this, why wouldn't the same idea work for ALM?



→ Artifacts such as defects, change requests, and tests become resources exposed as RDF that can be linked to each other

→ Tools simply access the resources via HTTP following the Linked Data principles

Problems encountered using Linked Data

- Tim Berners-Lee's four principles are a terrific foundation:
  1. Use URIs as names for things.
  2. Use HTTP URIs so that people can look up those names.
  3. When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL).
  4. Include links to other URIs so that people can discover more things.
- But in practice, this doesn’t go far enough.
- Rational had to come up with its own set of conventions on how to address the problems it faced when they actually are common usage patterns applicable beyond the ALM space:
  - How do I create a resource?
    - It seems obvious that you use POST to create, but what do you POST to?
  - Where can I get the list of resources that already exist?
  - Which vocabulary do I use?
  - Which media types do I use?
  - When containers get big, how do I split the information into pages?
  - How do I specify ordering?

Linked Data Basic Profile proposal

- A set of rules that clarify and extends Tim Berners-Lee's four basic rules focusing on the following concepts:
  - Basic Profile Resources (BPR)
    - HTTP and RDF techniques to use to read and write linked data
  - Basic Profile Containers (BPC)
    - A BPR to which you POST to create new things, GET to find existing things
    - Similar to what AtomPub does for XML
  - Paging
    - A mechanism to get the content of a BPC in chunks
  - Ordering
    - A mechanism to specify the order in which the content of a BPC is sorted

Some Linked Data Basic Profile rules

1. BP Resources are HTTP resources that can be created, modified, deleted and read using standard HTTP methods.
2. BP Resources use RDF to define their states.
3. You can request an RDF/XML representation of a BPR and possibly other reps (e.g., Turtle)
4. BP clients use Optimistic Collision Detection on update.
5. BP Resources use standard vocabularies such as Dublin Core.
6. BP Resources set rdf:type explicitly.
7. BP Resources use a restricted number of standard data types.
8. BP clients expect to encounter unknown properties and content.
9. BP clients do not assume the type of a resource at the end of a link.

Some Basic Profile Container rules

1. BP Containers are BP Resources
2. Clients can retrieve the list of resource members of a BP Container using GET
3. New resources are created using POST
4. Any resource can be POSTed to a BP Container – not just BP Resources
5. After POSTing a new resource to a BP Container, the new resource will appear as a member until it is deleted.
6. The same resource may appear in multiple BP Containers.
7. Clients can retrieve information about a BP Container without retrieving a full representation of its content, including its members.

Basic Profile Container Example

```
@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix bp: <http://open-services.net/ns/basicProfile#>.

<http://example.org/container1> a bp:Container ;

  dcterms:title "A very simple container";

  rdfs:member

      <http://example.org/container1/member1>,

      <http://example.org/container1/member2>,

      <http://example.org/container1/member3>.
```

Basic Profile Container Example with Pagination and Ordering

```
<http://example.org/netWorth/nw1/assetContainer>
  a bp:Container;
  dcterms:title "The assets of JohnZSmith";
  bp:membershipSubject <http://example.org/netWorth/nw1>;
  bp:membershipPredicate o:asset;
  bp:containerSortPredicates (o:value).

<http://example.org/netWorth/nw1/assetCont?firstPage>
  a bp:Page;
  bp:pageOf    <http://example.org/netWorth/nw1/assetContainer>;
  bp:nextPage  <http://example.org/netWorth/nw1/assetContainer?p=2>.

<http://example.org/netWorth/nw1>
  a o:netWorth;
  o:asset
    <http://example.org/netWorth/nw1/assetContainer/a1>,
    <http://example.org/netWorth/nw1/assetContainer/a2>.

<http://example.org/netWorth/nw1/assetContainer/a1>
   a o:Stock;
   o:value 50.

<http://example.org/netWorth/nw1/assetContainer/a2>
   a o:Cash;
   o:value 100.
```

Looking Forward

- Linked Data has the potential of becoming an important application integration model in the enterprise.
- But some significant issues need to be addressed:
  - Better definition and documentation of best practices, e.g. Basic Profile.
  - A few important specification gaps and problems:
    - PATCH
    - Security – Authentication, Access control
    - Shapes/constraints/validation
    - Cross-server query
    - URL changes
    - Server cloning
- W3C activity:
  - IBM submitted Linked Data Basic Profile 1.0 with DERI, EMC, Oracle, Red Hat, SemanticWeb.com, along with Cambridge Semantics and Siemens.
  - W3C proposes to launch the Linked Data Basic Platform WG to "produce a Recommendation for HTTP-based (RESTful) application integration patterns using read/write Linked Data."

BACKUP

Basic Profile – Restricted set of data types

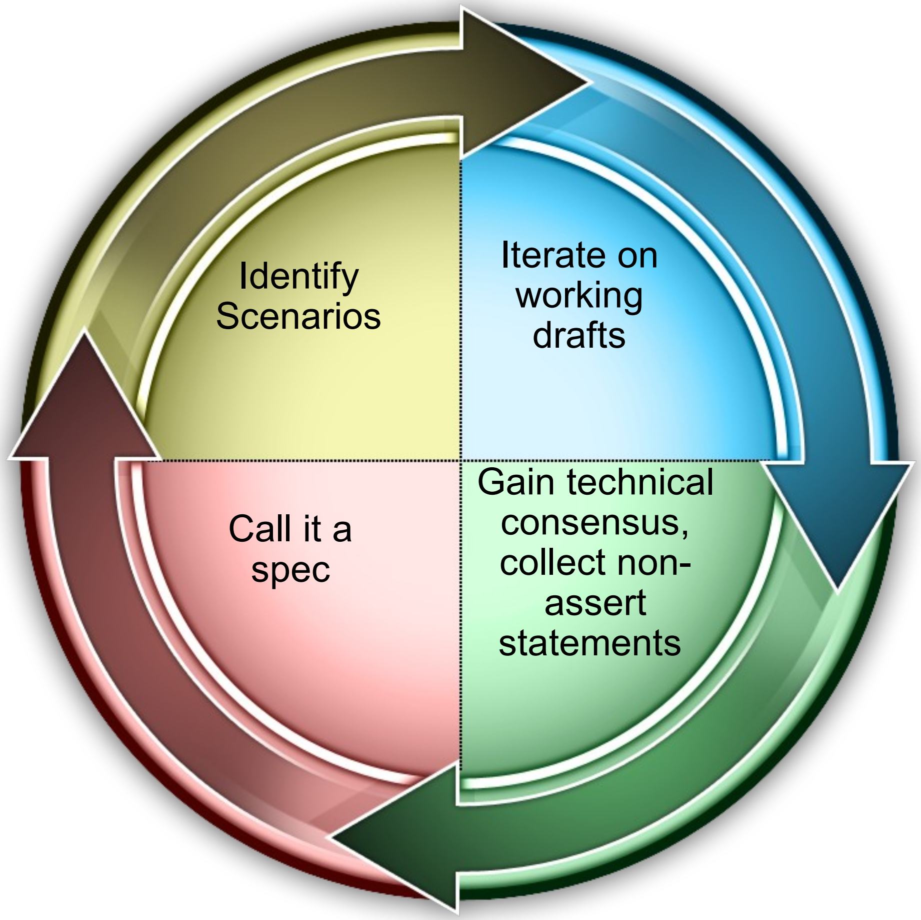
7. **BP Resources use a restricted number of data types.**
   - **Boolean:** A Boolean type as specified by XSD Boolean http://www.w3.org/2001/XMLSchema#boolean, reference: XSD Datatypes.
   - **DateTime:** A Date and Time type as specified by XSD dateTime http://www.w3.org/2001/XMLSchema#dateTime, reference: XSD Datatypes
   - **Decimal:** A decimal number type as specified by XSD Decimal http://www.w3.org/2001/XMLSchema#decimal, reference: XSD Datatypes
   - **Double:** A double floating point number type as specified by XSD Double http://www.w3.org/2001/XMLSchema#double, reference: XSD Datatypes.
   - **Float:** A floating point number type as specified by XSD Float http://www.w3.org/2001/XMLSchema#float, reference: XSD Datatypes.
   - **Integer:** An integer number type as specified by XSD Integer http://www.w3.org/2001/XMLSchema#integer, reference: XSD Datatypes.
   - **String:** A string type as specified by XSD String http://www.w3.org/2001/XMLSchema#string, reference: XSD Datatypes.
   - **XMLLiteral:** A literal XML value http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral

ALM integration requirements

- Create a link between ‘artifacts’ in different tools
  - E.g Test links to requirement. Development task links to requirement
- Within one tool, create an “artifact’ in another
  - E.g. Within a Test tool, log a defect using test results
- Across tools, share common concepts
  - **People** form a **team** to work on a **project** to produce the next **release** of a **product**
- Be able to query across information in multiple tools, e.g.
  - Show me all the critical requirements that don’t have test cases
  - Show me al the new test cases I should run on last night’s build

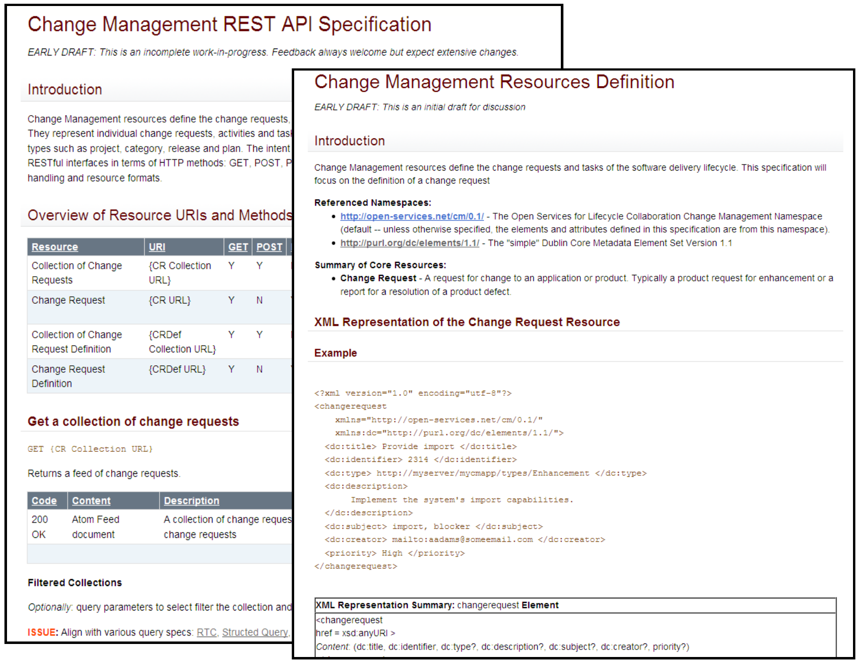
OSLC and Open Community
http://open-services.org

Open Services for Lifecycle Collaboration
open community. open interfaces. open possibilities



Identify Scenarios
Iterate on working drafts
Gain technical consensus, collect non-assert statements
Call it a spec

- Minimalist/additive approach
  - Not a “complete” definition for a given area
- Scenario driven scope
- Co-evolve spec and implementations
- Open participation around active core group



Change Management REST API Specification

EARLY DRAFT: This is an incomplete work-in-progress. Feedback always welcome but expect extensive changes.

Introduction

Change Management resources define the change requests,
They represent individual change requests, activities and tas
types such as project, category, release and plan. The intent
RESTful interfaces in terms of HTTP methods: GET, POST, P
handling and resource formats.

Overview of Resource URIs and Methods

Resource	URI	GET	POST
Collection of Change Requests	{CR Collection URL}	Y	Y
Change Request	{CR URL}	Y	N
Collection of Change Request Definition	{CRDef Collection URL}	Y	Y
Change Request Definition	{CRDef URL}	Y	N

Get a collection of change requests

```
GET {CR Collection URL}
```

Returns a feed of change requests.

Code	Content	Description
200 OK	Atom Feed document	A collection of change reques change requests

Filtered Collections

Optionally: query parameters to select filter the collection and

ISSUE: Align with various query specs: RTC, Structed Query,

Change Management Resources Definition

EARLY DRAFT: This is an initial draft for discussion

Introduction

Change Management resources define the change requests and tasks of the software delivery lifecycle. This specification will focus on the definition of a change request

Referenced Namespaces:
- http://open-services.net/cm/0.1/ - The Open Services for Lifecycle Collaboration Change Management Namespace (default -- unless otherwise specified, the elements and attributes defined in this specification are from this namespace).
- http://purl.org/dc/elements/1.1/ - The "simple" Dublin Core Metadata Element Set Version 1.1

Summary of Core Resources:
- **Change Request** - A request for change to an application or product. Typically a product request for enhancement or a report for a resolution of a product defect.

XML Representation of the Change Request Resource

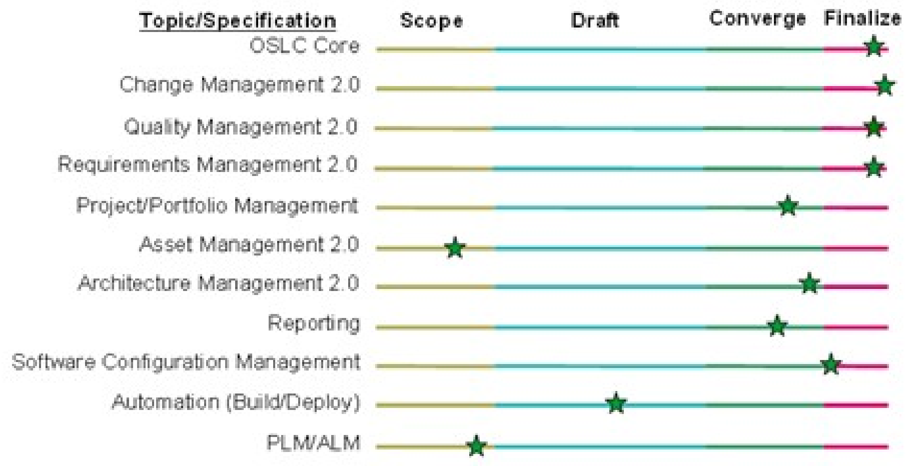
Example

```
<?xml version="1.0" encoding="utf-8"?>
<changerequest
    xmlns="http://open-services.net/cm/0.1/"
    xmlns:dc="http://purl.org/dc/elements/1.1/">
  <dc:title> Provide import </dc:title>
  <dc:identifier> 2314 </dc:identifier>
  <dc:type> http://myserver/mycmapp/types/Enhancement </dc:type>
  <dc:description>
       Implement the system's import capabilities.
  </dc:description>
  <dc:subject> import, blocker </dc:subject>
  <dc:creator> mailto:aadams@someemail.com </dc:creator>
  <priority> High </priority>
</changerequest>
```

XML Representation Summary: changerequest **Element**
<changerequest href = xsd:anyURI > *Content*: (dc:title, dc:identifier, dc:type?, dc:description?, dc:subject?, dc:creator?, priority?)

OSLC Community

- Eleven workgroups operating at open-services.net
  - Domain focused workgroups (e.g. CM, QM, RM)
  - Common issues and patterns (Core)
  - Solution oriented workgroups (e.g. PLM/ALM)
- Range of interests, expertise, involvement
  - 400+ registered community members (up from 70 people at RSC 2009)
  - Individuals from 34+ different companies have participated in OSLC workgroups (up from 5 companies at RSC 2009)
- Worthy of note
  - PLM/ALM Workgroup
    - Siemens leadership
  - Open source
    - Eclipse Lyo Project Proposal
    - Eclipse Mylyn project restructuring and positioning of OSLC
    - Mantis, Forges
  - Customer and integrator involvement
    - GM, Northrop Grumman, Tieto, Integrate Systems



Accenture
APG
Black Duck
Boeing
BSD Group
Citigroup
EADS
Emphasys Group
Empulsys
Fokus Fraunhofer
Galorath
General Motors
Health Care Services Corp
IBM
Institut TELECOM
Integrate Systems

Lender Processing Services
Northrop Grumman
Oracle
QSM
Rally Software
Ravenflow
Shell
Siemens
Sogeti
SourceGear/Teamprise
State Street
Tasktop (Eclipse Mylyn)
Thales
Tieto
TOPIC Embedded Systems
UrbanCode
WebLayers

Basic Profile Non-member Properties

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix dcterms: <http://purl.org/dc/terms/>.

<http://example.org/container1>

  a bp:Container;

  dcterms:title "A Basic Profile Container of Acme Resources";

  bp:membershipPredicate rdfs:member;

  dcterms:publisher <http://acme.com/>.
```

Shapes (Validation, constraints)

- RDFS and OWL do inferencing
  - “If it walks like a duck, and quacks like a duck, it must be a duck”
- In software (as in real life) it is also useful to go the other way
  - “If it wants to be a good duck, it must walk like one and quack like one”
- Everyone in software knows this model
  - Database Schemas
  - Object-oriented programming class definitions
- This model is actually very important for writing practical software
- Need to support more than one perspective on a duck
  - It’s a food-source, a source of insulating materials, a hazard to aircraft, a bird

Problem - this is a gap in RDF technologies

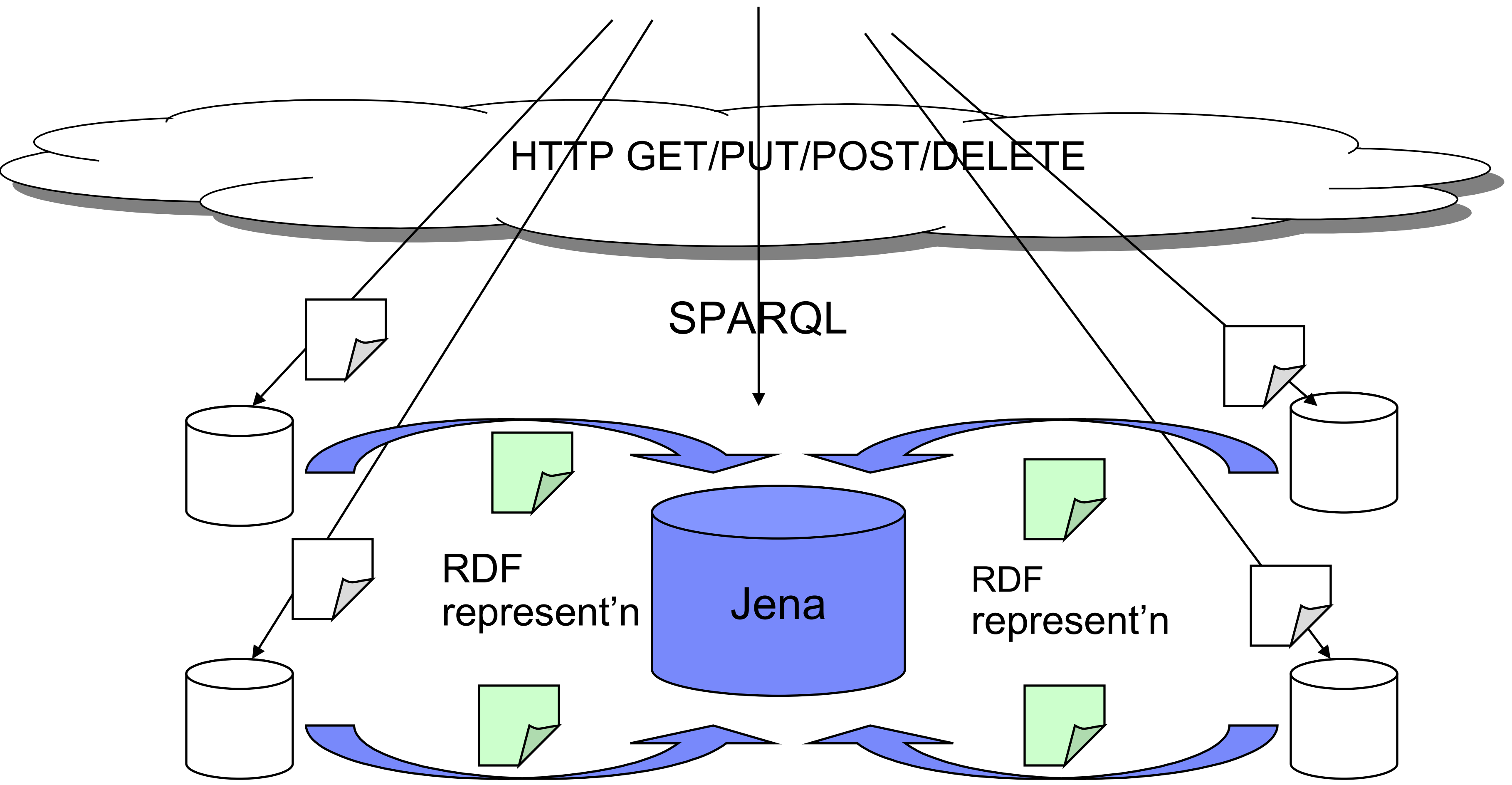
Shapes Example

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix bp: <http://open-services.org/basicProfile#/>.
@prefix : <http://example.org/test/>.

:teatCaseShape = {
   :testCaseShape
      a bp:Shape;
      bp:describedClass :TestCase;
      pb:propertyConstraint :testsRequirementConstraint }.

:testsRequirementConstraint = {
   :testsRequirementConstraint
      a bp:propertyConstraint;
      bp:describedProperty :testsRequirement;
      bp:rangeShape :requirementShape }.
```

Cross-server query

- Query server independent of storage, broader scope
- Schema-less query
- Exploit RDF
  - XML and XQuery were tried, did not work as well

Implementing query on data not in a repository



Security

- Requirements
  - Allow each product to have its own security administration and authentication
  - Provide optional central place to configure users, integrate with LDAP etc
  - Allow each product to manage it's own access control lists, pre-conditions and post-actions
  - Provide optional central federation of ACL administration
- Solutions
  - For products that implement their own authentication, federate with OAuth
  - For products that want to manage ACLs, provide federation
  - Provide optional building blocks for new-construction that take over admin, authentication and ACLs