

Extensible SPARQL Functions with Embedded Javascript

Gregory Todd Williams

University of Maryland

SPARQL

- Standardized query interface
- Allows interoperability
- Extensibility through “extension functions”

Extension Functions

```
SELECT *  
WHERE {  
  ?subj ?pred ?obj .  
  FILTER( ex:func( ?subj ) ) .  
}
```

- Allow custom filtering of query results
- Can use any constant or variable as function arguments

Extension Functions

- Support for extension functions still limited
- Use requires programming and configuration *on server and before query time*

Motivating Example

- Want all images taken near a specific point
- Could retrieve *all* images, and filter locally, but...
 - Wastes bandwidth
 - Client may be too slow to do this efficiently
 - Client may not have enough memory to hold all results (mobile clients, ...)

Motivating Example (cont)

- Better to filter on the server, but...
- Server needs to be configured to support distance function

Just-in-time Extension Functions

- Implement functions in a scripting language (Javascript)
- Use dereferencable HTTP URIs to define functions
- Dereference URI for function metadata and pointer to source code
- Allow endpoints to retrieve and execute function implementations on the fly

Implementation

- Working implementation in RDF::Query perl engine
- Uses SpiderMonkey (Mozilla JavaScript engine)
- Minimal changes required to RDF::Query code

Javascript API

- Uses a subset of AJAR RDF API for Javascript
- Extension function arguments are constants or RDF terms
- Extension function return value is a constant or RDF term



Client



SPARQL Endpoint



Webservice

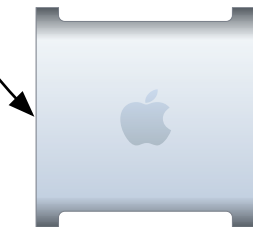
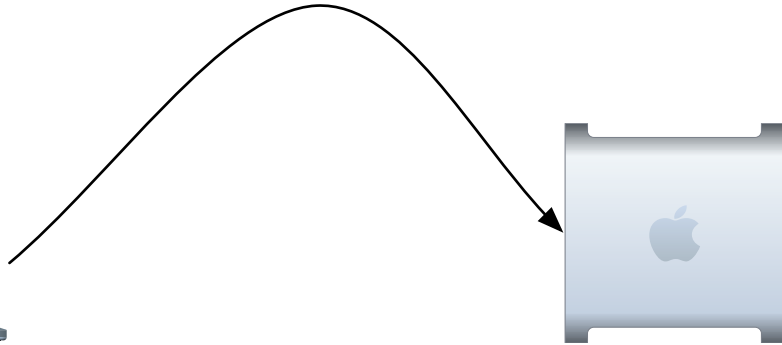


Webservice

```
PREFIX ex: <http://example.com/functions.rdf#>
SELECT ?place ?image
WHERE {
  ?image dcterms:spatial [
    a geo:Point;
    foaf:name ?place;
    geo:lat ?lat ; geo:long ?long
  ].
  FILTER( ex:distance( ?lat, ?long, 38.9, -76.9 ) < 15 ) .
}
```



Client



SPARQL Endpoint



Webserver



Webserver

```
PREFIX ex: <http://example.com/functions.rdf#>
SELECT ?place ?image
WHERE {
  ?image dcterms:spatial [
    a geo:Point;
    foaf:name ?place;
    geo:lat ?lat ; geo:long ?long
  ].
  FILTER( ex:distance( ?lat, ?long, 38.9, -76.9 ) < 15 ) .
}
```



Client



SPARQL Endpoint

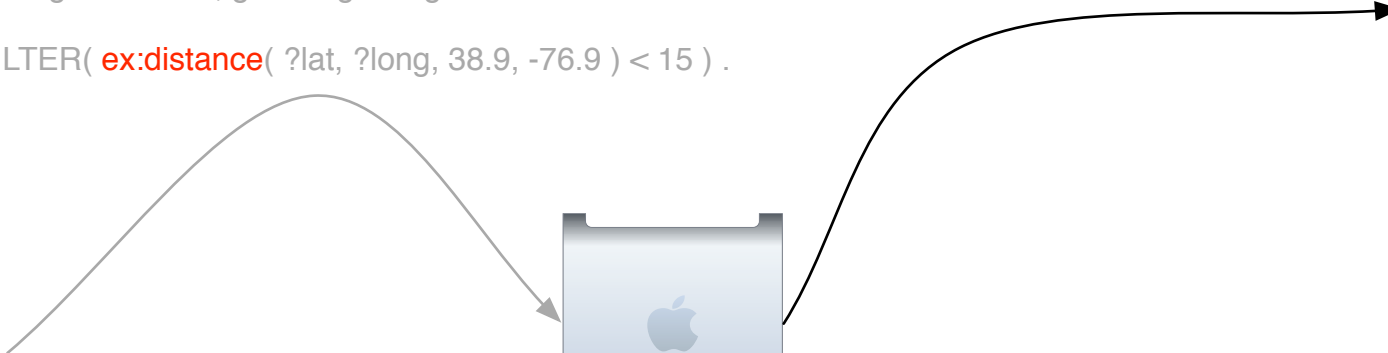


Webserver



Webserver

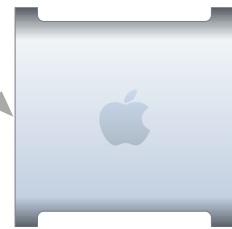
GET <http://example.com/functions.rdf>



```
PREFIX ex: <http://example.com/functions.rdf#>
SELECT ?place ?image
WHERE {
  ?image dcterms:spatial [
    a geo:Point;
    foaf:name ?place;
    geo:lat ?lat ; geo:long ?long
  ].
  FILTER( ex:distance( ?lat, ?long, 38.9, -76.9 ) < 15 ) .
}
```



Client



SPARQL Endpoint



Webserver



Webserver

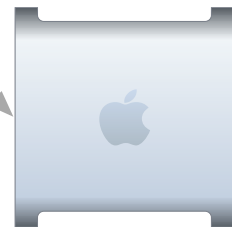
GET <http://example.com/functions.rdf>

```
:distance a ex:Function ;
ex:source <http://example2.com/distance.js> ;
ex:function "distance" .
```

```
PREFIX ex: <http://example.com/functions.rdf#>
SELECT ?place ?image
WHERE {
  ?image dcterms:spatial [
    a geo:Point;
    foaf:name ?place;
    geo:lat ?lat ; geo:long ?long
  ].
  FILTER( ex:distance( ?lat, ?long, 38.9, -76.9 ) < 15 ) .
}
```



Client



SPARQL Endpoint



Webserver

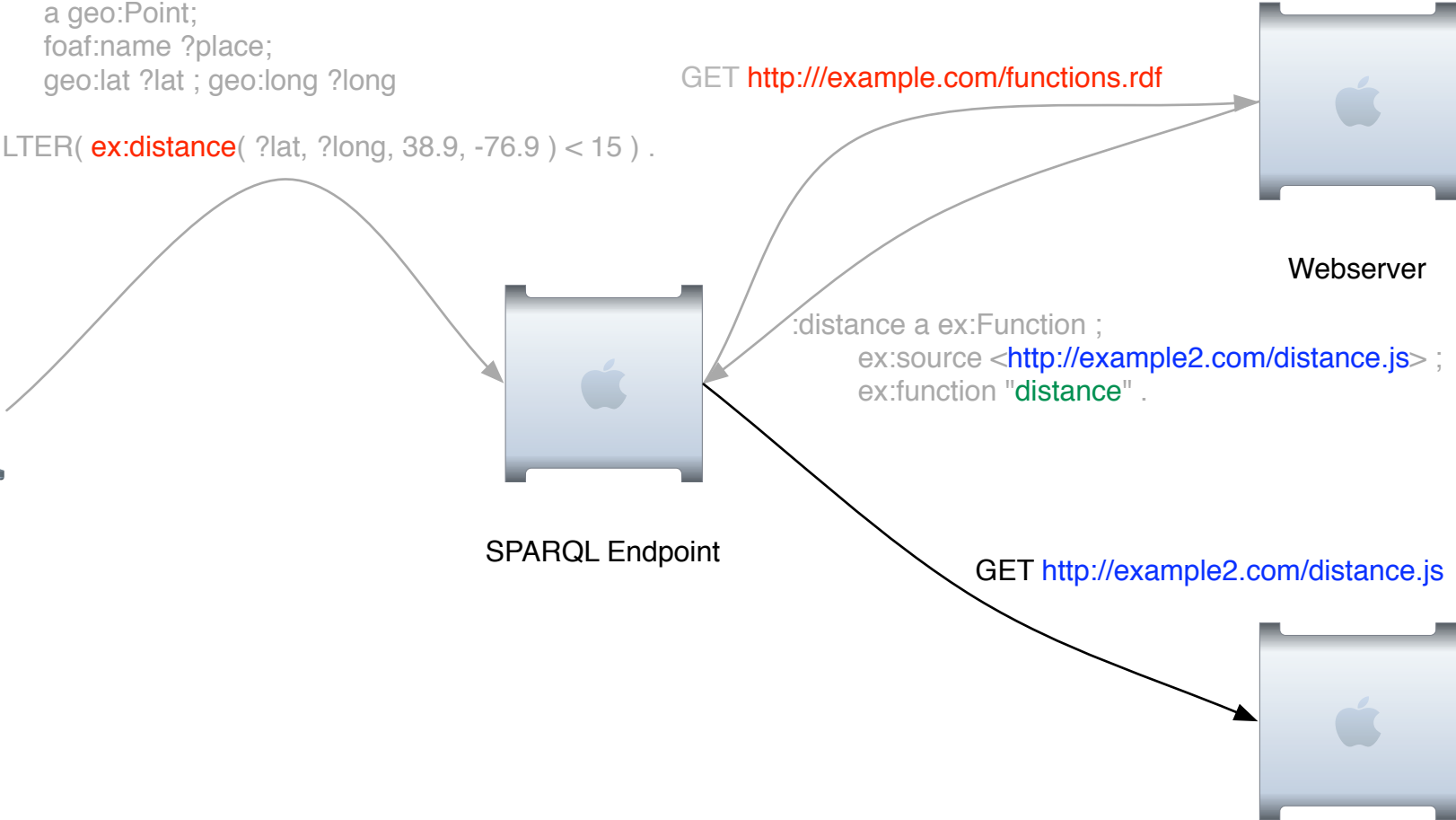


Webserver

GET <http://example.com/functions.rdf>

```
:distance a ex:Function ;
ex:source <http://example2.com/distance.js> ;
ex:function "distance" .
```

GET <http://example2.com/distance.js>



```

PREFIX ex: <http://example.com/functions.rdf#>
SELECT ?place ?image
WHERE {
  ?image dcterms:spatial [
    a geo:Point;
    foaf:name ?place;
    geo:lat ?lat ; geo:long ?long
  ].
  FILTER( ex:distance( ?lat, ?long, 38.9, -76.9 ) < 15 ) .
}

```



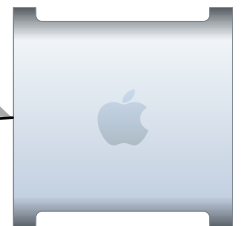
Client



SPARQL Endpoint



Webserver



Webserver

GET <http://example.com/functions.rdf>

```

:distance a ex:Function ;
ex:source <http://example2.com/distance.js> ;
ex:function "distance" .

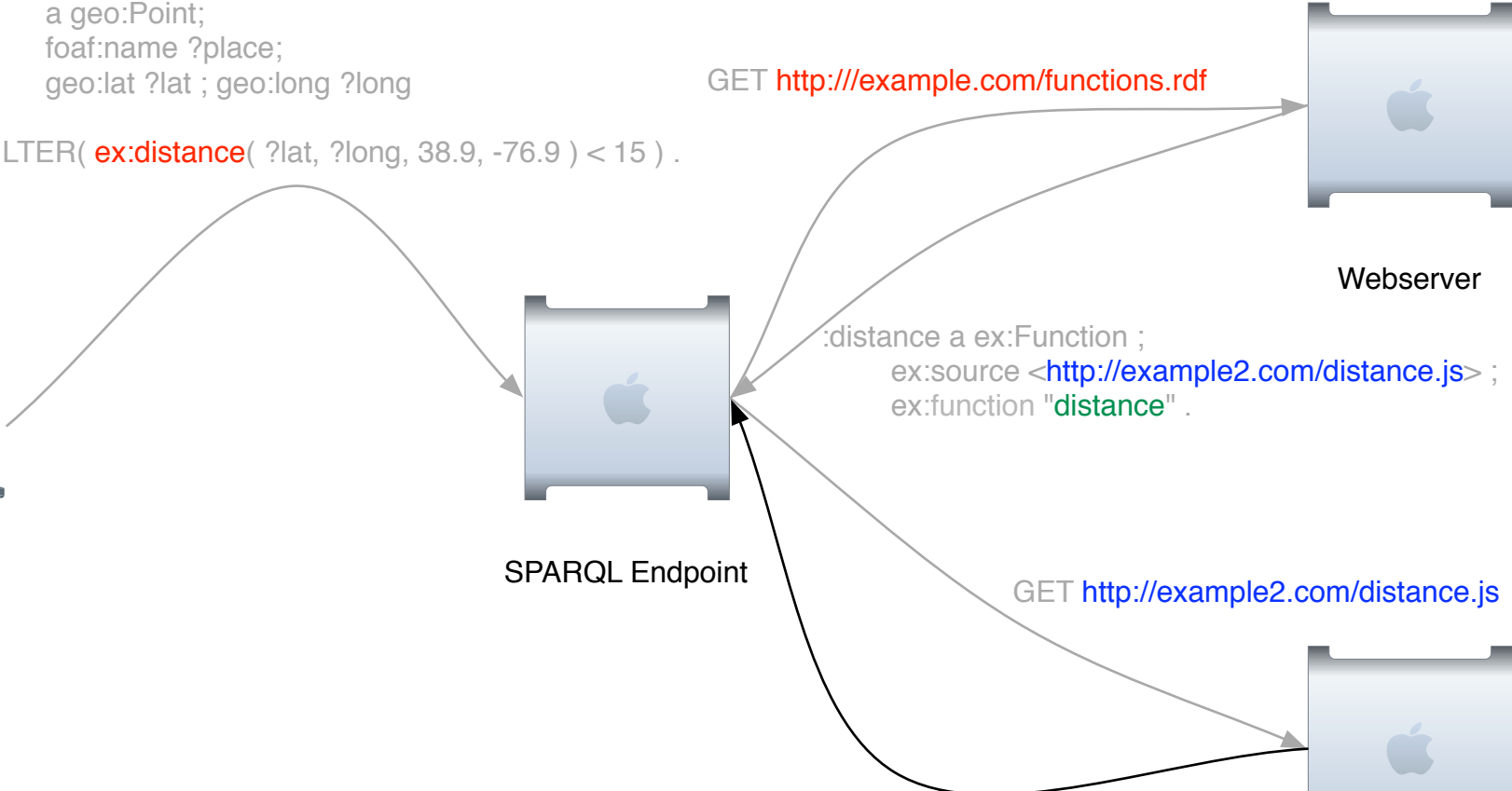
```

GET <http://example2.com/distance.js>

```

function distance (lat1, lon1, lat2, lon2) {
  ...
}

```



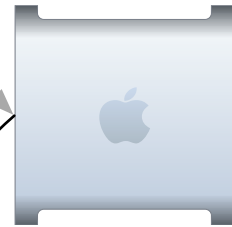
```

PREFIX ex: <http://example.com/functions.rdf#>
SELECT ?place ?image
WHERE {
  ?image dcterms:spatial [
    a geo:Point;
    foaf:name ?place;
    geo:lat ?lat ; geo:long ?long
  ].
  FILTER( ex:distance( ?lat, ?long, 38.9, -76.9 ) < 15 ) .
}

```



Client



SPARQL Endpoint



Webserver



Webserver

GET <http://example.com/functions.rdf>

```

:distance a ex:Function ;
ex:source <http://example2.com/distance.js> ;
ex:function "distance" .

```

GET <http://example2.com/distance.js>

```

function distance (lat1, lon1, lat2, lon2) {
  ...
}

```

?place	?image
University of Maryland	http://www.mindswap.org/~rreck/mindlab.jpg
Washington, DC	http://farm1.static.flickr.com/201/448918094_fe793cb8a7_b.jpg

Example: distance

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .  
@prefix dc: <http://purl.org/dc/elements/1.1/> .  
@prefix ex: <http://www.mindswap.org/~gtw/sparql#> .
```

```
<>
```

```
  a ex:Namespace;  
  ex:hasFunction <#distance> .
```

```
<#distance>
```

```
  a ex:Function;  
  dc:description "Great Circle Distance";  
  ex:source <http://localhost/distance.js>;  
  ex:function "distance" .
```

Example: distance

```
function distance( lat1, lon1, lat2, lon2 ) {
    lat1 = deg2rad( makeTerm(lat1).toString() );
    lat2 = deg2rad( makeTerm(lat2).toString() );
    lon1 = deg2rad( makeTerm(lon1).toString() );
    lon2 = deg2rad( makeTerm(lon2).toString() );
    var londiff = Math.abs(lon1 - lon2);
    var s1 = square(Math.sin((lat2 - lat1) / 2));
    var s2 = square(Math.sin( londiff / 2 ));
    var sq = Math.sqrt(
        s1
        + Math.cos(lat1)
        * Math.cos(lat2)
        * s2
    );
    var adist = 2 * Math.asin( sq );
    var r = 6372.795;
    return r * adist;
}
function square (x) { return x * x; }
function deg2rad(d) { return Math.PI*d/180 }
```

Concerns

- Protect against malicious code
 - Local attacks
 - Remote DOS attacks
- Maintain system performance by allowing native implementation

Cryptographic Signing

- Implementations may be signed using GPG
- SPARQL endpoint may run only code signed by trusted signatures
- Expect organizations or interest groups to sign domain specific functions

Demo

Performance

- To address performance problems, endpoints may:
 - Override slower Javascript implementations
 - Define maximum runtime or resource consumption

Future work

- Implement more of AJAR interface
- Support for other programming languages
- Support for other signing mechanisms

Questions?