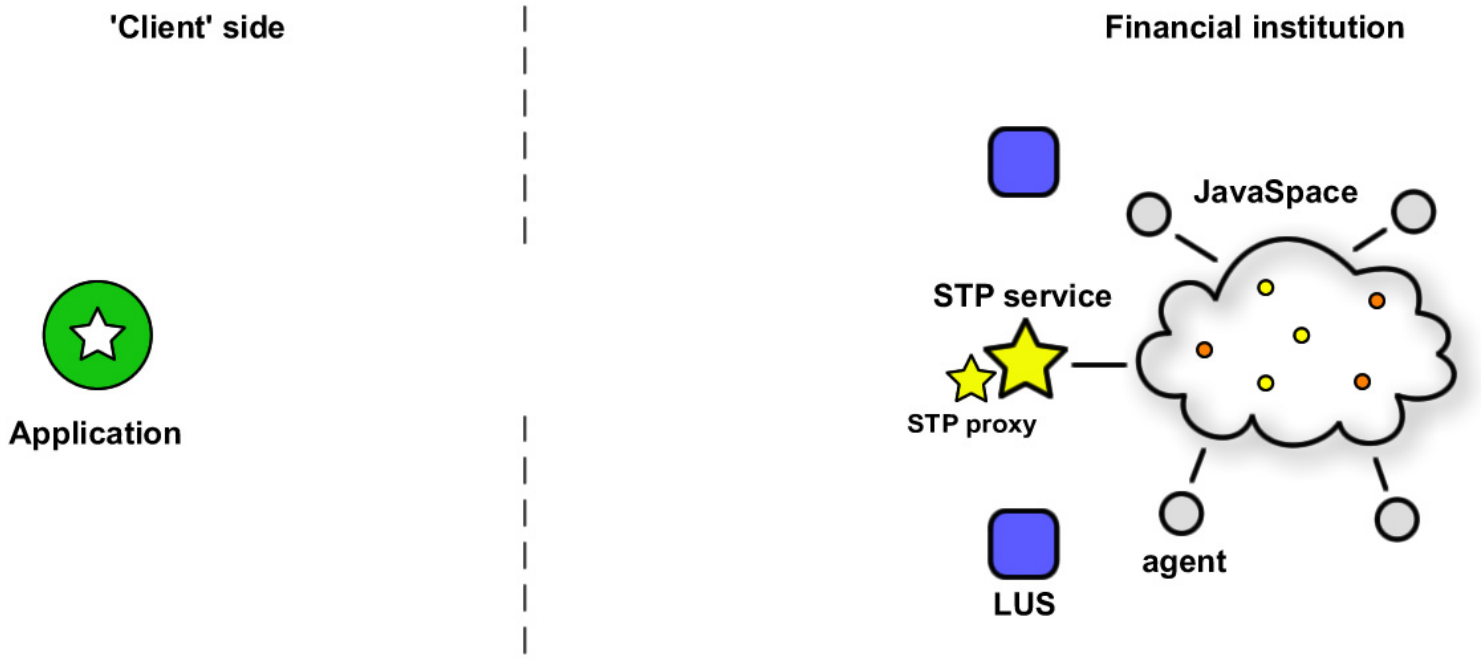
Cheiron

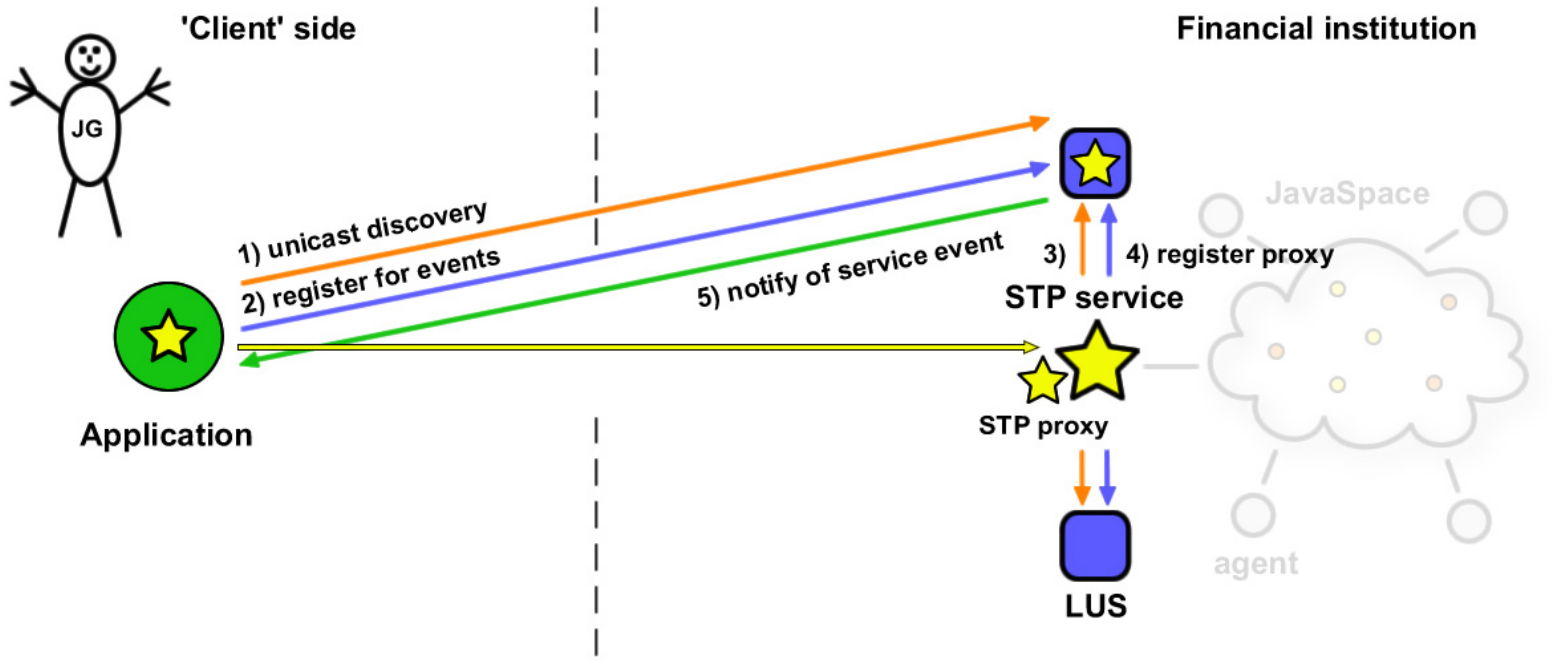
across the firewall

Mark Brouwer, JCM10 Brussels

Common Jini deployment



Local deployment

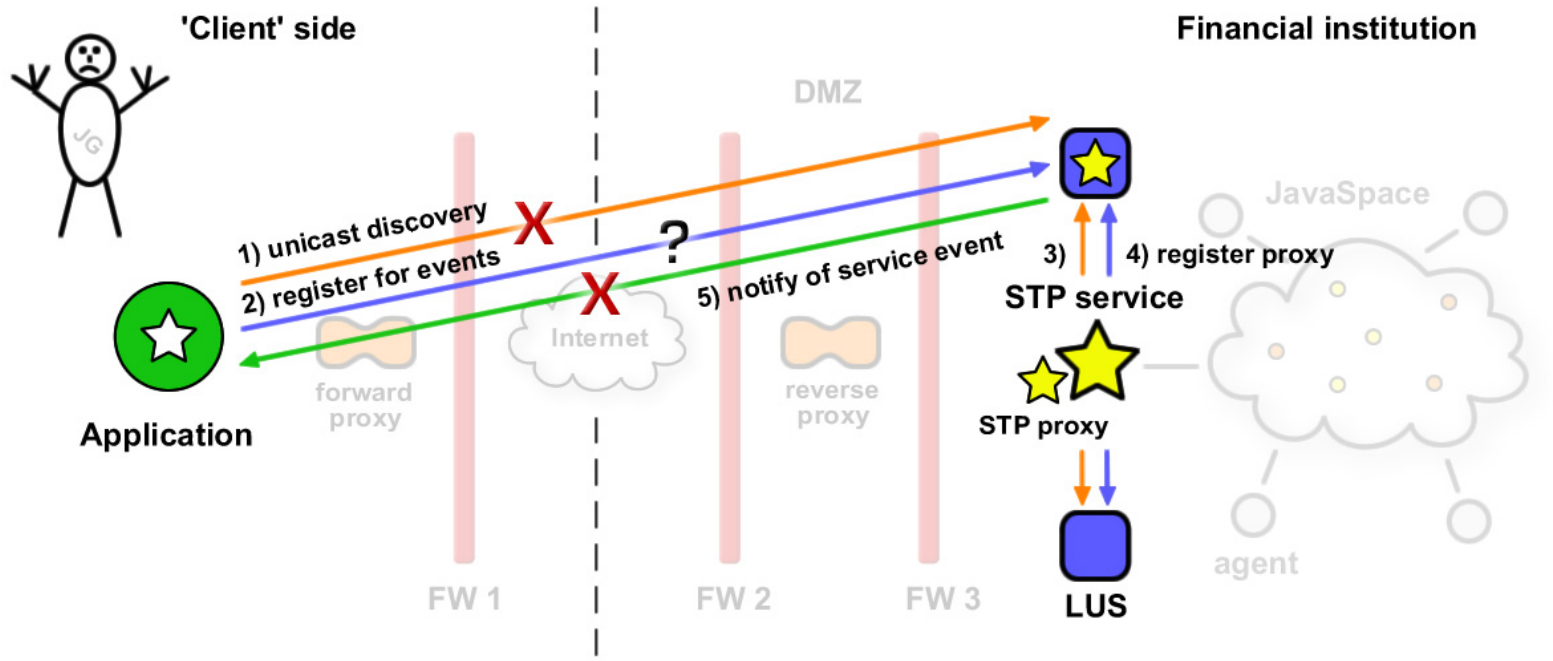


Cheiron

Highlander Principle of the Internet

- There can only be one
  - protocol: HTTP(S)
  - inbound port: 80 (443)
- We have firewalls to enforce the principle
  - ignoring the principle is asking for troubles
  - resulted in the development of 'HTTP proxying techniques' to get us into most places that are otherwise not routable

Internet Jini deployment



Cheiron

Problems Jini faces with Internet deployment

- Unicast discovery protocol
  - does not support HTTP(S)
  - does not allow for routing information
- Jini ERI `Http(s)Endpoint` no support for routing information
- Use of Jini Distributed Events likely not possible
  - inbound request to client
  - outbound request from server
  - code mobility

Solutions with available technology

- 'Bringing Jini to the web' by Mahmoud Abaza, Warren Strange
  - transport based on JXTA endpoints, allows for callbacks
- 'Customizing the Jini ERI Invocation and Transport Layers' by Michael Warres & Daniel Jiang
  - using socket relay service

Why it didn't take off

- Both are partial solutions
- Large performance penalty in case of JXTA
- Relay service must be reachable by both client and server
- Some guesses
  - configuration burden
  - Jini Community only eats what is in the Jini Starter Kit or resembles a JavaSpace

Cheiron

Jini Platform must adapt

- Unicast discovery over HTTP(S) that allows for routing information
- `Http(s)Endpoint` that provides support for routing information
- Event model that does not require callbacks

Cheiron

Unicast discovery over HTTP(S)

- Wrap current 'raw' unicast protocol in HTTP(S) protocol
  - HTTP GET request with **`Request-URI`** of form: **`path-absolute [ "?" query ]`**
    - has routing information as part of **`path-absolute`**
    - request data encoded as content type **`application/x-www-form-urlencoded`** in query part of request, supports both version 1 and 2 of discovery protocol
  - message body of HTTP response contains the unicast discovery contents that includes the marshalled lookup service proxy

Unicast discovery over HTTP(S)

- Expand what is allowed for the Jini Technology URL
  - absolute path for routing purposes
  - query part to differentiate between the type of protocol, such as `raw`, `http` and `https`
- Some examples
  - jini://www.cheiron.org = jini://www.cheiron.org:4160/?raw
  - jini://www.cheiron.org:80/unicast?http
  - jini://www.cheiron.org:443/unicast?https

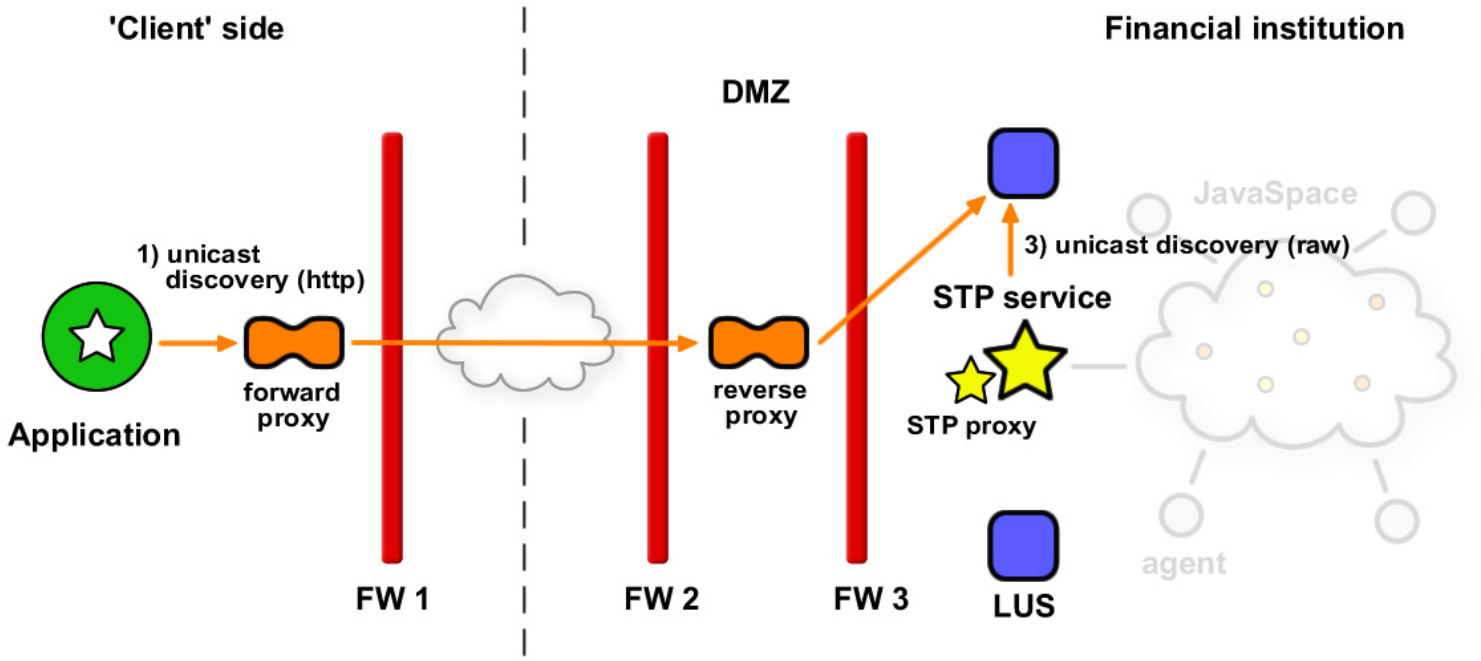
Impact

- New `org.cheiron.discovery.UniversalLookupLocator`
  - takes only a URL as argument
  - supports `raw` and `http` unicast discovery
- Modified `net.jini.discovery.LookupLocatorDiscovery`
- Modified `net.jini.core.discovery.LookupLocator`

```
public UnicastResponse getUnicastResponse()
        throws IOException, ClassNotFoundException {...}
public UnicastResponse getUnicastResponse(int timeout)
        throws IOException, ClassNotFoundException {...}
```

- Modified 'Reggie' supports `raw` and `http` unicast discovery

Unicast discovery revisited



Cheiron

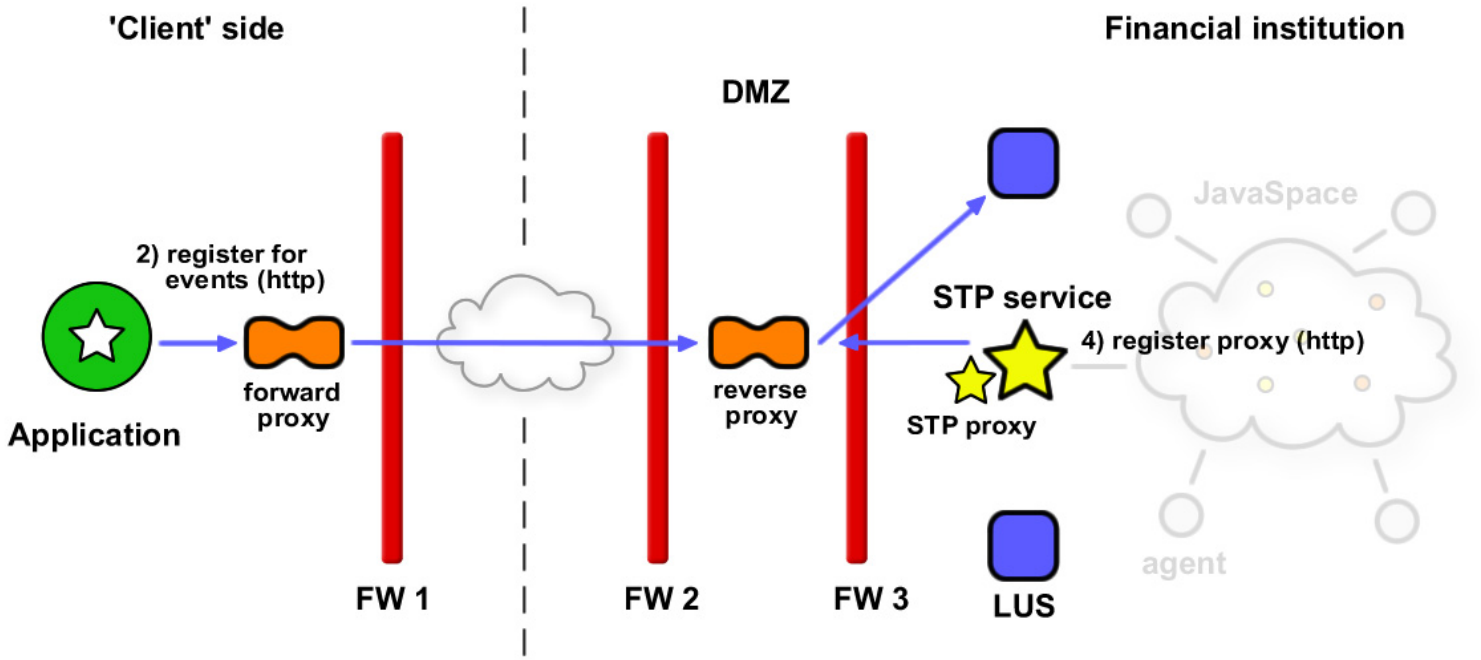
Routable Jini ERI

`Http(s)Endpoint`**`S`**

- Created **`Http(s)RequestUriEndpoint`** with support for routing information
  - allows one to provide an additional **`Request-URI`** of form: **`path-absolute [ "?" query ]`**
- Created **`net.jini.jeri.AsymmetricServerEndpoint`**
  - pairs an arbitrary **`ServerEndpoint`** and **`Endpoint`** for protocol conversion
  - pass in to **`net.jini.jeri.BasicJeriExporter`**

Cheiron

Jini ERI revisited



Cheiron

JSC Service Event model to prevent from callbacks

- Requirements
  - event registration leased, usage of `RemoteEvent`
  - no need for client to export itself as a service
  - must allow for implementation based on polling and must be able to leverage Jini ERI
- Part of Jini Service Container Specification, nothing prevents others to implement the model seperate from a JSC container

Cheiron

JSC Service Event model applied to Lookup Service

```
public interface ServiceRegistrarX extends ServiceRegistrar {

    EventRegistration notify(ServiceTemplate tmpl,
                             int transitions,
                             ServiceEventListener listener,
                             MarshalledObject handback,
                             long leaseDuration) throws
                                                  RemoteException;
}

public interface ServiceEventListener extends EventListener {

    void notify(RemoteEvent event) throws UnknownEventException;
}
```

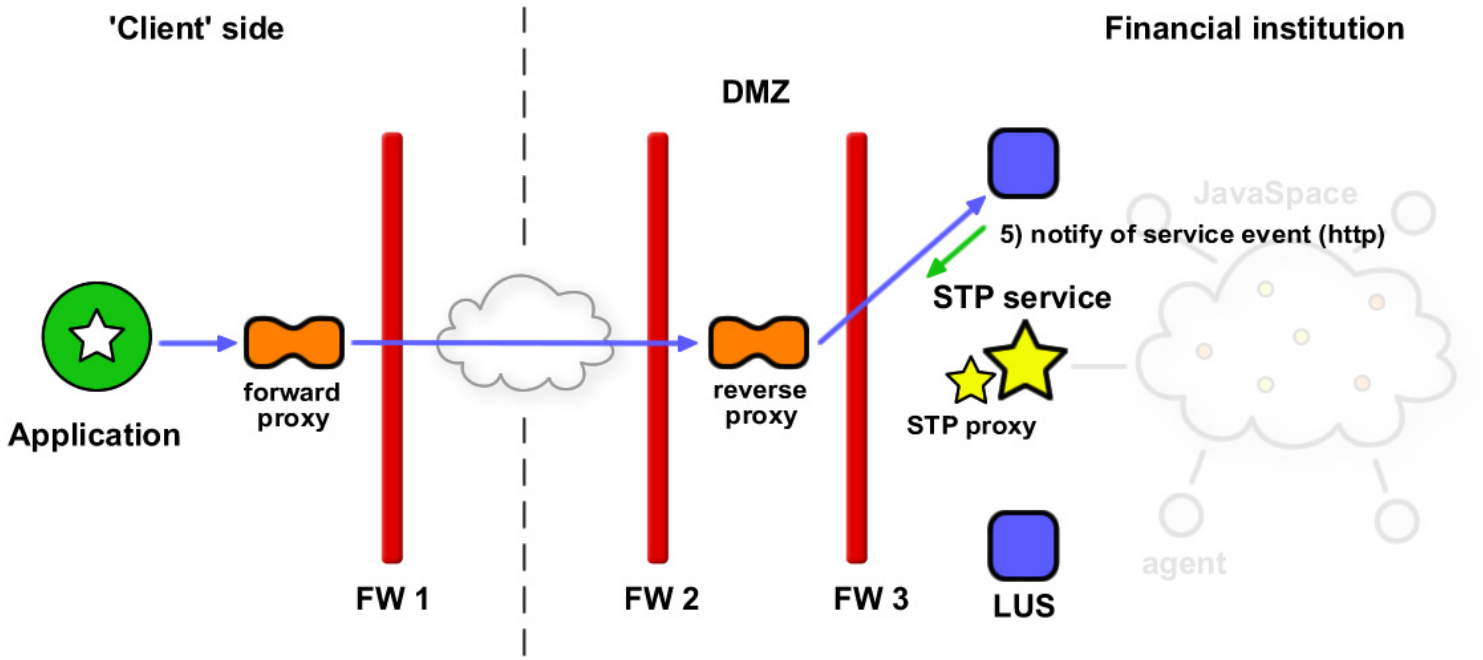
Cheiron

Modifications to existing code

- 'Reggie' fitted into Seven
  - stripped event handling code and replaced by event framework of Seven (supports both event models)
  - supports raw unicast discovery and unicast discovery over HTTP
- Added support for `ServiceRegistarX` to `ServiceDiscoveryManager` through `Configuration`

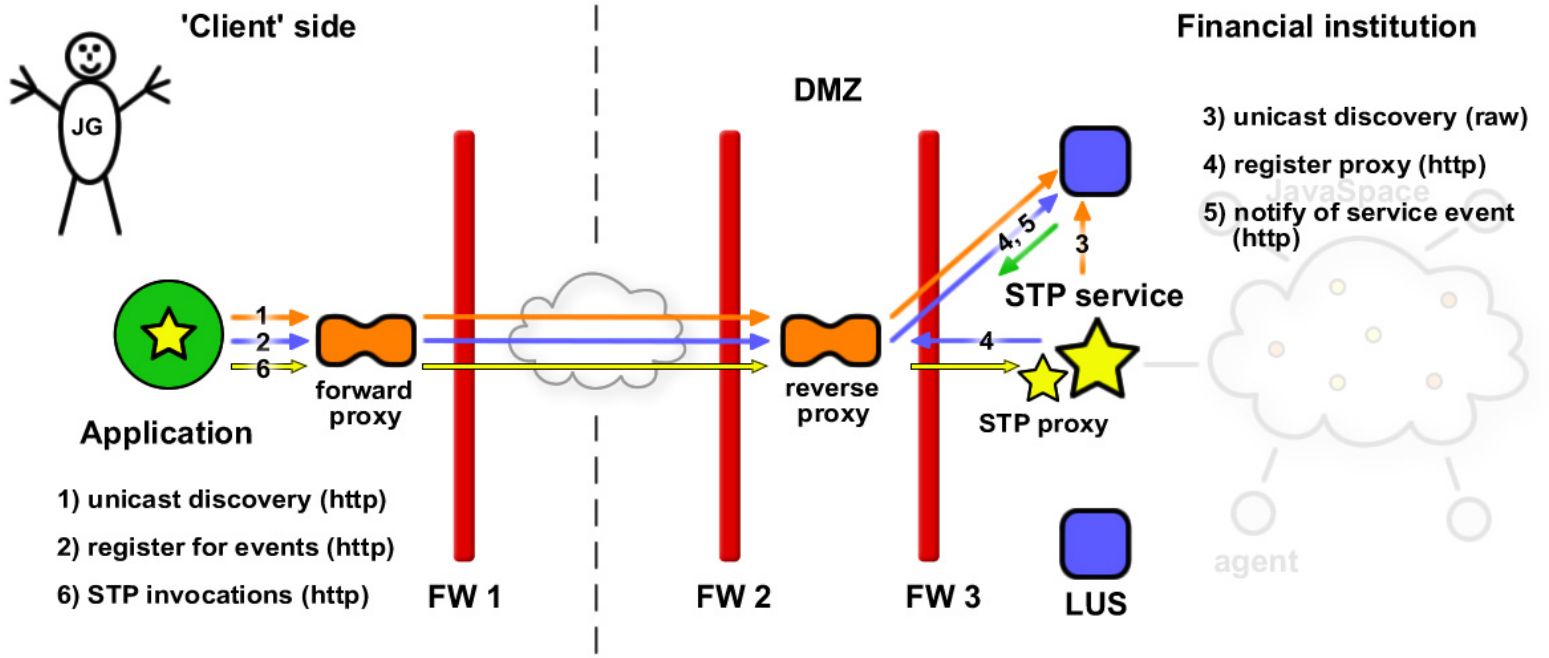
Cheiron

Events revisited



Cheiron

Internet Jini deployment revisited



Cheiron

Example Apache configuration

```
<VirtualHost *:80>
  ServerName jini-portal.cheiron.org
  SetEnvIf User-Agent ".*com\.sun.*HttpClientConnection" downgrade-1.0

  ProxyPass /codebase http://polaris:15555/
  ProxyPassReverse /codebase http://polaris:15555/

  ProxyPass /unicast http://polaris:4160/
  ProxyPassReverse /unicast http://polaris:4160/

  ProxyPass /jeri/lus http://polaris:5050/
  ProxyPassReverse /jeri/lus http://polaris:5050/

  ProxyPass /jeri/stp http://polaris:5050/
  ProxyPassReverse /jeri/stp http://polaris:5050/
</VirtualHost>
```

Future work

- Get the aforementioned work in a future ASF project
  - unicast discovery over HTTP(S)
    - implement version 2 of unicast discovery over HTTP and `ConstrainableUniversalLookupLocator`
    - implement unicast discovery over HTTPS
  - `Http(s)UriRequestEndpoint` and `AsymmetricServerEndpoint`
- Develop AJP13 based `ServerEndpoint` to pair with `HttpsRequestUriEndpoint`

Cheiron

Future work

NEEDS YOU!

Cheiron

Q & A