

# SCA Java binding.jsonrpc

Unable to render  
{include} me  
Included page could  
not be found.

Unable to render  
{include} me  
Included page could  
not be found.

## <binding.jsonrpc>

Tuscany supports [JSON-RPC](#) as a protocol for use with SCA services by using the <binding.jsonrpc> SCDL extension. This enables remote web browser clients to easily make RPC style calls to server-side SCA components.

The complete timeline of available and future plans is given in the [Tuscany Web 2.0 Roadmap](#)

## Using the Tuscany JSON-RPC binding

You could use this binding without any configuration, or by providing a specific service URI.  
To include it on a SCA service choose one of the SCDL examples below :

```
<binding.jsonrpc/>
```

```
<binding.jsonrpc uri="http://localhost:8080/store/catalog"/>
```

## Consuming JSON-RPC services on the client application

Any JSON-RPC client may be used to access SCA services which use <binding.jsonrpc>, below we are going to describe different ways you could consume the Tuscany JSON-RPC services in your client application.

### Utilizing Tuscany Implementation.widget

When your web client application is defined as an SCA component utilizing [Tuscany Widgets](#), a JavaScript is generated which may be included within an HTML document to properly inject services references to SCA references defined on the same HTML document.

This script is used by simply including the following tag within the HTML page :

```
<script type="text/javascript" src="html-page-name.js" />
```

This initializes the proxys for the SCA services which can then be injected into SCA references to make requests to the server-side components. For example, if there was a service named "myService" which had operations "aOnewayRequest" and "anRpcRequest" the scripts in the HTML page could now invoke these operations with the following:

```
//@Reference  
var myService = new Reference("myService");  
myService.aOnewayRequest(args);
```

or

```
//@Reference  
var myService = new JSONRpcClient("myService");  
myService.anRpcRequest(args, responseFunction);
```

Also see [Tuscany Widgets](#) for more details.

## Utilizing a JavaScript Client Proxy

To simplify the task for web browsers developers Tuscany provides a 'binding-jsonrpc.js' JavaScript client proxy code that can be included in your application.

After copying the script to your application, it can be used by simply including the following tag within the HTML page :

```
<script type="text/javascript" src="binding-jsonrpc.js" />
```

This initializes the proxies for the SCA services which can then be used to make requests to the server-side components. Based on the scenario described above, below are the ways to invoke these operations.

```
var myService = new JSONRpcClient("Catalog").service;
myService.aOnewayRequest(args);
```

or

```
var myService = new JSONRpcClient("Catalog").service;
myService.anRpcRequest(args, responseFunction);
```

## Handling JSON-RPC Response with callbacks

In that example 'responseFunction' is the name of a function which is called to process the response and which gets called asynchronously on another thread when the response is available. RPC requests are done this way instead of the simpler "answer = myService.anRpcRequest(args)" to avoid hanging the browser while the (potentially slow) request is being processed. An example of the responseFunction for the previous example is:

```
function responseFunction(answer){
    // do something with answer
}
```

## Handling errors

```
//initialization code
try{
    myService.anRpcRequest(args, responseFunction);
} catch(e) {
    //handle error
    alert(e);
}

function responseFunction(answer, exception){
    //handle exception information
    if(exception){
        alert(exception.message);
        return;
    }
    // do something with answer
}
```

## Using SCA JSON-RPC services with Dojo

Apache Tuscany JSON-RPC services provide built-in support for [Dojo RPC](#). The [Dojo](#) toolkit is a popular framework for writing Ajax /Web 2.0 style browser client applications. Tuscany SCA services which use <binding.jsonrpc> will by default support the [Simple Method Description \(SMD\)](#) protocol. SMD is similar to ?wsdl for Web services, entering a service endpoint appended with ?smd will return a SMD descriptor for the service.

Using Tuscany SCA services with Dojo can therefore be as simple as the following:

```
var myService = new dojo.rpc.JsonService("myService?smd");
```

## Supported data types


The JSON-RPC binding utilize the Databinding Framework to provide support for the following data transformations :

- Primitive Type <==> JSON <==> Primitive Type
- Array of Primitive Type <==> JSON <==> Array of Primitive Type
- Java bean <==> JSON <==> Java bean
- List <==> JSON <==> List
- Map <==> JSON <==> Map
- Set <==> JSON <==> Set

## Some examples:

There are two samples showing using <binding.jsonrpc>, one which uses the Dojo Toolkit on the client, and another which uses the Tuscany scaDomain.js script. The samples are [helloworld-dojo](#) and [helloworld-jsonrpc](#).

## Security Policy support in HTTP and Web 2.0 Bindings

 work in progress

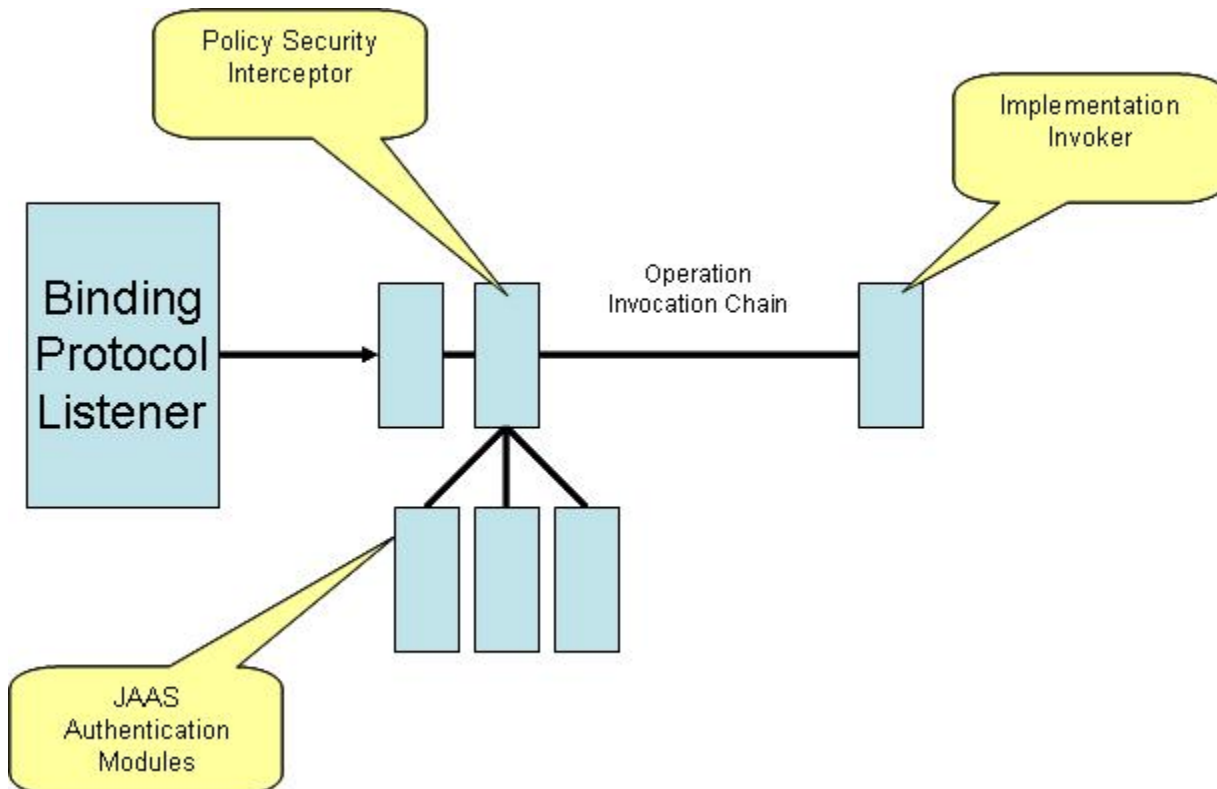
## Scenarios

- A Web 2.0 application requires that a user get authenticated before it can access the application.
- A Web 2.0 application requires that all communication between client/server be done using SSL.
- A given service, exposed using a web 2.0 binding requires user authentication.
- A given operation, exposed using a web 2.0 binding requires user authentication.

## Policy Interceptor

The design approach that is being considered is to inject policy security interceptors, that would properly validate and enforce the security intents.

The authentication will be done using JAAS modules for authentication, and initially we would support authenticating to a list of username/password supplied by the application or using an LDAP.





#### Differences between `<binding.jsonrpc>` and `<binding.dwr>`

The current Tuscany SCA runtime supports `<binding.jsonrpc>` and `<binding.dwr>` which provide similar functionality. The differences are:

- `<binding.jsonrpc>` supports the SMD protocol enabling easy use with Dojo, `<binding.ajax>` does not support SMD
- `<binding.ajax>` supports SCA references and using [COMET](#) style asynchronous operation, `<binding.jsonrpc>` does not
- `<binding.jsonrpc>` uses the standard [JSON-RPC](#) protocol, `<binding.dwr>` uses a proprietry protocol using [DWR](#)

These differences should be resolved by the Tuscany SCA 1.0 release.



#### Changes since 0.90 release

The Tuscany JSON-RPC and Ajax binding's have had significant functional and useability changes since the 0.90 release. It is recommended that if possible the latest code is used (the description on this page is based on the latest code).