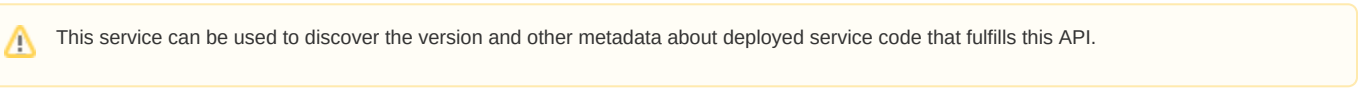
Service Catalog Service Contract Description - v0.9-alpha

Brief description

The Service Catalog service provides information about the services offered by the Bamboo Service Platform. Based on this information, service consumers can see what functionality a service provides and how, select which service provider best meets their needs, and invoke the service.

Version

This page describes the Service Catalog service API, v. 0.9



Overview and Definitions

The Service Catalog service provides a service consumer (e.g. end user, application developer) with access to information about available services developed for use in, and deployed through, the Bamboo Service Platform (BSP). This information includes both a reference to description of the service's RESTful API (i.e. a contract that specifies how the service will perform internet-based HTTP GET, POST, PUT, and DELETE calls); as well as a profile of the service, listing all service providers and any supporting services that are potentially involved in implementing the functionality specified by the service. Using the Service Catalog service, a service consumer interested in some set of functionality can:

- Obtain information about what services are currently available
- What functionality a service provides and how
- How to invoke a service over the internet
- Select from among multiple candidate service providers for a service; service providers are differentiated by the organization that developed the service provider, and by the version of the service a service provider supports.

A **Service Profile** describes the functionality provided by a single service, as well as information about the service providers that it calls in the course of delivering that functionality.

A **Service Profiles list** is an XML document that, for each available service, includes a link to the service's Service Description, and a summary version of its Service Profile.

Data available via the Service Catalog service is supplied by service developers. *For a fuller understanding of the information in a Service Profile, please refer to the Service Catalog Service Description and Assumptions - v0.9 page of this wiki (a child to this page).*

Assumptions

For assumptions implicit in this service's functionality, visit the Service Catalog Service Description and Assumptions - v0.9 page of this wiki (a child to this page).

References

${**REPOSITORY_ROOT**} is http://svn.code.sf.net/p/projectbamboo/code/

- *Service Catalog Service Description and Assumptions - v0.9*
- Codebase: ${REPOSITORY_ROOT}/platform-services/bsp/trunk/bsp/core-services/service-catalog-service/

Known Issues

None

ROA Layer API

RESTful service methods performing Read and List functions over Service Profiles.

Base URLs

It is assumed in this documentation that no centrally-run instances of the Bamboo Services Platform will be running after the project ends on 31 March 2013. Therefore, base URLs are assumed to be on a developer's machine, localhost. The base URL with a port number assumes that the BSP is running unsecured; the URL without a port number assumes that security is enforced and Apache Web Server is intercepting and redirecting service calls. Please see the page *Identity and Access Management - Authentication and Authorization* for context, as well as links to installation and configuration instructions for secured instances of BSP.

Note that *ONLY services at v0.9 or greater* will run properly in a secured instance of the BSP.

Currently available base URLs:

- http://localhost:8181
- http://localhost/services

Client Responsibilities in Requests to Secured BSP Web Services

This section of the Service API documentation describes a client application's responsibilities when making requests to secured Web Services hosted on the Bamboo Services Platform (including this service).

A secured instance of the Bamboo Services Platform (BSP) implies a significant set of installation and configuration tasks for which the operator of the BSP is responsible. These are described in overview on the wiki page *Identity and Access Management - Authentication and Authorization* , and in detail on pages linked from that one.

(1) A client must be configured as a Trusted Application if requests are to be treated other than as *Anonymous*

A client application – whether a web app or a simple testing client such as Firefox Poster or curl – may make requests anonymously or as a "Trusted Application." Only a Trusted Application may assert the identity of a user on behalf of whom the request is made, and scoped roles to be assigned to that user; Bamboo Services trust such clients to assert the identity and assigned-roles *only* of users who have authenticated in the current session of application activity. (A special-case type of client application, termed *Innovation Licensed* applications, are trusted to assert the identity of and roles assigned to a fixed set of special-case users without those users having to authenticate in the current session.)

Configuration of client applications are described in detail in this wiki page: *Configure Apache Web Server for Client Auth*. It is assumed in #2, below, that this configuration has been performed.

(2) A Trusted client is expected to pass HTTP Request Headers to identify itself and an authenticated user

A client application that is *Trusted* in the security context of the Bamboo Trust Federation (cf. *Identity and Access Management - Authentication and Authorization*) must augment each request to a service hosted by a secured instance of the Bamboo Services Platform (BSP) with a set of HTTP headers, as follows:

- **X-Bamboo-AppID**: A UUID that identifies the client research environment, application, tool, or service; this UUID is issued as part of the process of *registering* a trusted client in the Bamboo Trust Federation as described in overview on the page *Identity and Access Management - Authentication and Authorization*; and in detail with respect to physical establishment of trust on the page *Configure Apache Web Server for Client Auth*. The value of this header is linked to the X.509 certificate by which the application establishes an SSL connection to the BSP host in the registration process, and a match between this *Application ID* and the linked X.509 certificate is checked by the BSP on receipt of *every* request.
- **X-Bamboo-BPID**: A UUID that identifies the logged-in user on whose behalf the request is being sent; this value, a Bamboo Person Identifier, or BPId, is obtained via a call to the *Person Service* that occurs in time between user login and any other service request. See the *Read a BambooPersonId* method of the *Person Service API* for details. [†]
- **X-Bamboo-Roles**: A pipe-delimited (|) set of scoped roles asserted by the trusted client to belong to the logged-in user, of the form role@domain, which are to be evaluated as factors in the determination of whether the request satisfies policies (access restrictions) that apply to the requested resource. If a user is *authenticated*, the client is expected to include the role *undefined@domain* where *domain* identifies the organization that authenticated the user (example: *undefined@google.com* is a client app's assertion that the user authenticated to Google). This header is otherwise optional (depending on policies governing the requested resource that may require one or more scoped roles for access to be permitted). Example of multiple roles asserted in this header: *roleA@domainOne.xxx|roleB@domainTwo* [‡]

[†] The value of X-Bamboo-BPID is set to the identifier for the application itself (X-Bamboo-AppID) when a client application calls the Person Service to *crea te* a new Bamboo Person Identifier; or to *retrieve* the BPId for a user based on the identifier of the IdP with which she has logged in and an SHA-256 hash of that IdP's user identifier for the logged-in person.

[‡] Policies and policy evaluation are described on the page *Authorization and Policy*. Also see *Conventions for representing Identity Providers in the Bamboo Trust Federation*.

Schema

${**REPOSITORY_ROOT**} is http://svn.code.sf.net/p/projectbamboo/code/

- ${REPOSITORY_ROOT}/platform-services/bsp/trunk/bsp/core-services/service-catalog-service/domain/src/main/resources/ServiceProfile.xsd

Service Profile

A **Service Profile** describes the functionality provided by a single service, as well as information about the service providers that it calls in the course of delivering that functionality.

Read a Service Profile

A client requests a service's profile by submitting the service's unique identifier (i.e. spid).

Calling Method and Arguments

Invoked as an HTTP GET method. Send an HTTP request of the form:

```
GET /bsp/serviceprofiles/spid HTTP/1.1
```

Parameter	Meaning
spid	The unique identifier for a Service Profile
Accept Header	application/xml

Response

On success, a response with a "200 OK" HTTP status code will be returned.

Parameter	Meaning
HTTP Body	An instance of a ServiceProfile XML document

Example ServiceProfile XML document:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<sc:serviceProfile
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dcterms="http://purl.org/dc/terms/"
    xmlns:foaf="http://xmlns.com/foaf/0.1/"
    xmlns:bsp="http://projectbamboo.org/bsp/resource"
    xmlns:sc="http://projectbamboo.org/bsp/services/core/servicecatalog/domain/serviceprofile"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    rdf:about="http://localhost/bsp/serviceprofiles/urn:uuid:813f76f4-cc4b-44d8-ad36-81810e0356d7">
    <sc:servicePId>urn:uuid:813f76f4-cc4b-44d8-ad36-81810e0356d7</sc:servicePId>
    <sc:name>Service Catalog Service</sc:name>
    <sc:version>1.0.0</sc:version>
    <sc:objectClass>org.projectbamboo.bsp.services.core.servicecatalog.resources.ServiceProfiles</sc:
objectClass>
    <sc:serviceDescription>Provides information about the services offered by the Bamboo Service Platform</sc:
serviceDescription>
    <sc:serviceDescriptionLocation>https://wiki.projectbamboo.org/display/BTECH/Service+Catalog+Service+Home<
/sc:serviceDescriptionLocation>
    <sc:serviceVendor>UC Berkeley</sc:serviceVendor>
    <sc:serviceRestfulAddress>http://localhost/bsp/serviceprofiles</sc:serviceRestfulAddress>
    <sc:serviceProviders>
        <sc:name>Service Profile Document Handler Service</sc:name>
        <sc:defaultServiceProvider>true</sc:defaultServiceProvider>
        <sc:serviceProviderSupportedVersionsRange>[1.0.0,2.0.0)</sc:serviceProviderSupportedVersionsRange>
        <sc:serviceProviderDetails>
            <sc:serviceRanking>1</sc:serviceRanking>
            <sc:serviceProviderType>functional</sc:serviceProviderType>
            <sc:serviceProviderContact>falvarez@berkeley.edu</sc:serviceProviderContact>
            <sc:metadata sc:key="defaultServiceProvider">true</sc:metadata>
            <sc:metadata sc:key="org.springframework.osgi.bean.name">serviceProfileDocumentHandler</sc:metadata>
            <sc:metadata sc:key="service.id">449</sc:metadata>
            <sc:metadata sc:key="Bundle-SymbolicName">org.projectbamboo.bsp.servicecatalog.service</sc:metadata>
            <sc:metadata sc:key="Bundle-Version">0.8.0.-SNAPSHOT</sc:metadata>
        </sc:serviceProviderDetails>
    </sc:serviceProviders>
    <sc:servicesUsed>
        <sc:servicePId>urn:uuid:813f76f4-bc4b-44d8-ad36-81810f0356d7</sc:servicePId>
        <sc:name>Service Catalog</sc:name>
        <sc:version>1.0.0</sc:version>
        <sc:objectClass>org.projectbamboo.bsp.services.core.servicecatalog.service.ServiceCatalogService</sc:
objectClass>
        <sc:serviceDescription>Provides access to service providers</sc:serviceDescription>
        <sc:serviceDescriptionLocation>https://wiki.projectbamboo.org/display/BTECH
/Service+Catalog+Service+Description+and+Assumptions</sc:serviceDescriptionLocation>
        <sc:serviceVendor>UC Berkeley</sc:serviceVendor>
    </sc:servicesUsed>
    <sc:servicesUsed>
        <sc:servicePId>urn:uuid:813f76f4-bc4b-44d8-ad36-81810e0356d9</sc:servicePId>
        <sc:name>Service Profile Document Handler Service</sc:name>
        <sc:version>1.0.0</sc:version>
        <sc:objectClass>org.projectbamboo.bsp.services.core.servicecatalog.service.
ServiceProfileDocumentHandlerService</sc:objectClass>
        <sc:serviceDescription>Document marshaling and unmarshaling for the Service Profile resource</sc:
serviceDescription>
        <sc:serviceDescriptionLocation>https://wiki.projectbamboo.org/display/BTECH
/Document+Handler+Service+Home</sc:serviceDescriptionLocation>
        <sc:serviceVendor>UC Berkeley</sc:serviceVendor>
    </sc:servicesUsed>
</sc:serviceProfile>
```

Exceptions

If an error occurred, some non-2xx code will be returned. Check the HTTP Status Code that is returned in the response's HTTP headers for the specific error. The following errors may be returned in response to the Get request:

Error (Status Code)	Meaning	Returned When
401	Unauthorized	The resource could not be created because the client submitting the request either has not provided authentication credentials, or authentication failed, or authorization has been refused for those credentials
404	Not Found	The resource requested for reading does not exist
500	Internal Server Error	A service error prevented the resource from being returned

List Service Profiles

A **Service Profiles list** is an XML document that, for each available service, includes a link to the service's Service Description, and a summary version of its Service Profile.

Calling Method and Arguments

Invoked as an HTTP GET method. Send an HTTP request of the form:

```
GET [http://services.projectbamboo.org/bsp/serviceprofiles HTTP/1.1
```

Parameter	Meaning
Accept Header	application/xml

Response

On success, a response with a "200 OK" HTTP status code will be returned.

Parameter	Meaning
HTTP Body	An instance of a serviceProfileList XML document

Example serviceProfileList XML document:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<sc:serviceProfileList
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dcterms="http://purl.org/dc/terms/"
    xmlns:foaf="http://xmlns.com/foaf/0.1/"
    xmlns:bsp="http://projectbamboo.org/bsp/resource"
    xmlns:sc="http://projectbamboo.org/bsp/services/core/servicecatalog/domain/serviceprofile"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <bsp:header>
        <dcterms:created xsi:type="dcterms:W3CDTF">2012-04-27T11:27:20.645-07:00</dcterms:created>
        <dcterms:creator xsi:type="foaf:Agent" rdf:about="serviceprofiles"/>
        <bsp:orderedBy>ascending</bsp:orderedBy>
        <bsp:listLength>1</bsp:listLength>
        <bsp:pageNumber>1</bsp:pageNumber>
        <bsp:pageLength>20</bsp:pageLength>
    </bsp:header>
    <sc:serviceProfile rdf:about="http://localhost/bsp/serviceprofiles/urn:uuid:813f76f4-cc4b-44d8-ad36-
81810e0356d7">
        <sc:servicePId>urn:uuid:813f76f4-cc4b-44d8-ad36-81810e0356d7</sc:servicePId>
        <sc:name>Service Catalog Service</sc:name>
        <sc:version>1.0.0</sc:version>
        <sc:serviceDescription>Provides information about the services offered by the Bamboo Service Platform<
/sc:serviceDescription>
        <sc:serviceDescriptionLocation>https://wiki.projectbamboo.org/display/BTECH
/Service+Catalog+Service+Home</sc:serviceDescriptionLocation>
        <sc:serviceVendor>UC Berkeley</sc:serviceVendor>
        <sc:serviceRestfulAddress>http://localhost/bsp/serviceprofiles</sc:serviceRestfulAddress>
        <sc:serviceProviders>
            <sc:name>Service Profile Document Handler Service</sc:name>
            <sc:defaultServiceProvider>false</sc:defaultServiceProvider>
            <sc:serviceProviderSupportedVersionsRange>[1.0.0,2.0.0)</sc:serviceProviderSupportedVersionsRange>
        </sc:serviceProviders>
    </sc:serviceProfile>
</sc:serviceProfileList>
```

Exceptions

If an error occurred, some non-2xx code will be returned. Check the HTTP Status Code that is returned in the response's HTTP headers for the specific error. The following errors may be returned in response to the Get request:

Error (Status Code)	Meaning	Returned When
401	Unauthorized	The resource could not be created because the client submitting the request either has not provided authentication credentials, or authentication failed, or authorization has been refused for those credentials
500	Internal Server Error	A service error prevented the resource from being returned

SOA Layer API

See generated Javadoc for this service. Packages are org.projectbamboo.bsp.services.core.servicecatalog..*

ServiceCatalogService API for obtaining service providers.

Return a specific service provider

```
/**
 * Returns a specific service provider.
 *
 * @param service - the service's interface
 * @param serviceProviderId - identifies a service provider
 * @return instance of a service provider for the service
 * @throws FileNotFoundException if the service provider does not exist
 */
<T> T getServiceProvider(Class<T> service, URI serviceProviderId) throws FileNotFoundException;
```

Return a service provider

```
/**
 * Returns a service provider.
 *
 * @param service - the service's interface
 * @return instance of a service provider for the service
 * @throws FileNotFoundException if the service does not have a service provider
 */
<T> T getServiceProvider(Class<T> service) throws FileNotFoundException;
```

Return all service providers

```
/**
 * Returns all service providers.
 *
 * @param service - the service's interface
 * @return list of instances of a service provider for the service
 * @throws FileNotFoundException if the service does not have a service provider
 */
 <T> List<T> getAllServiceProvider(Class<T> service) throws FileNotFoundException;
```

Return a service provider for a specific version of the service

```
/**
 * Returns a service provider for a specific version of the service.
 *
 * @param service - the service's interface
 * @param version - the version of the service
 * @return instance of a service provider for the service
 * @throws IllegalArgumentException if version is invalid
 * @throws FileNotFoundException if the service does not have a service provider
 */
<T> T getServiceProvider(Class<T> service, String version) throws FileNotFoundException;
```

Return a service provider that satisfies a set of predicates

```
/**
 * Returns a service provider that satisfies a set of predicates.
 *
 * @param service - the service's interface
 * @param filter - conforms to RFC 1960 grammar
 * @return instance of a service provider for the service
 * @throws FileNotFoundException if the service does not have a service provider
 */
<T> T getServiceProvider(Class<T> service, RFC1960SearchFilter filter)  throws FileNotFoundException;
```

Returns a service provider that satisfies a simple predicate

```
/**
 * Return a service provider that satisfies a simple predicate.
 *
 * @param service - the service's interface
 * @param version - the version of the service; can be null
 * @param predicate - in the form key=value
 * @return instance of a service provider for the service
 * @throws IllegalArgumentException if the predicate is invalid
 * @throws FileNotFoundException if the service does not have a service provider
 */
<T> T getServiceProvider(Class<T> service, String version, String predicate) throws FileNotFoundException;
```

Return a service's RESTful endpoint

```
/**
 * Returns a service's RESTful endpoint.
 *
 * @param service - the service's interface
 * @param version - the version of the service
 * @return URL
 * @throws IllegalArgumentException if version is invalid
 * @throws FileNotFoundException if the service does not have a service provider
 */
URL getServiceRestfulEndPoint(Class<?> service, String version) throws FileNotFoundException;
```