Cloud Application Management for Platforms Version 1.1

Committee Specification Draft ~~03 /~~04 / Public Review Draft ~~01~~02

~~31 July 2013~~

12 February 2014

Specification URIs

This version:

~~http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd01/camp-spec-v1.1-csprd01.pdf (Authoritative)~~
~~http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd01/camp-spec-v1.1-csprd01.html~~
~~http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd01/camp-spec-v1.1-csprd01.doc~~
http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd02/camp-spec-v1.1-csprd02.pdf (Authoritative)
http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd02/camp-spec-v1.1-csprd02.html
http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd02/camp-spec-v1.1-csprd02.doc

Previous version:

~~http://docs.oasis-open.org/camp/camp-spec/v1.1/csd02/camp-spec-v1.1-csd02.pdf Authoritative)~~
~~http://docs.oasis-open.org/camp/camp-spec/v1.1/csd02/camp-spec-v1.1-csd02.html~~
~~http://docs.oasis-open.org/camp/camp-spec/v1.1/csd02/camp-spec-v1.1-csd02.doc~~

http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd01/camp-spec-v1.1-csprd01.pdf (Authoritative)
http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd01/camp-spec-v1.1-csprd01.html
http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd01/camp-spec-v1.1-csprd01.doc

Latest version:

~~http://docs.oasis-open.org/camp/camp-spec/v1.1/camp-spec-v1.1.pdf (Authoritative)~~
~~http://docs.oasis-open.org/camp/camp-spec/v1.1/camp-spec-v1.1.html~~
~~http://docs.oasis-open.org/camp/camp-spec/v1.1/camp-spec-v1.1.doc~~

http://docs.oasis-open.org/camp/camp-spec/v1.1/camp-spec-v1.1.pdf (Authoritative)
http://docs.oasis-open.org/camp/camp-spec/v1.1/camp-spec-v1.1.html
http://docs.oasis-open.org/camp/camp-spec/v1.1/camp-spec-v1.1.doc

Technical Committee:

OASIS Cloud Application Management for Platforms (CAMP) TC)

Chair:

Martin Chapman (martin.chapman@oracle.com), Oracle

Editors:

Jacques Durand (jdurand@us.fujitsu.com), Fujitsu Limited
Adrian Otto (adrian.otto@rackspace.com), Rackspace Hosting, Inc.
Gilbert Pilz (gilbert.pilz@oracle.com), Oracle
Tom Rutt (trutt@us.fujitsu.com), Fujitsu Limited

Additional artifacts:

This prose specification is one component of a Work Product that also includes:

- CAMP Type Definitions: http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd02/type-definitions/

Related work:

This specification is related to:

- *Cloud Application Management for Platforms Test Assertions Version 1.1*. Latest version. http://docs.oasis-open.org/camp/camp-ta/v1.1/camp-ta-v1.1.html

Abstract:

This document defines the artifacts and APIs that need to be offered by a Platform as a Service (PaaS) cloud to manage the building, running, administration, monitoring and patching of applications in the cloud. Its purpose is to enable interoperability among self-service interfaces to PaaS clouds by defining artifacts and formats that can be used with any conforming cloud and enable independent vendors to create tools and services that interact with any conforming cloud using the defined interfaces. Cloud vendors can use these interfaces to develop new PaaS offerings that will interact with independently developed tools and components.

Status:

This document was last revised or approved by the OASIS Cloud Application Management for Platforms (CAMP) TC on the above date. The level of approval is also listed above. Check the “Latest version” location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee’s email list. Others should send comments to the Technical Committee by using the “Send A Comment” button on the Technical Committee’s web page at http://www.oasis-open.org/committees/camp/.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (http://www.oasis-open.org/committees/camp/ipr.php).

Citation format:

When referencing this specification the following citation format should be used:

[CAMP-v1.1]

Cloud Application Management for Platforms Version 1.1. ~~31 July 2013. OASIS Committee Specification Draft 03 / Public Review Draft 01. http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd01/camp-spec-v1.1-csprd01.html~~Edited by Jacques Durand, Adrian Otto, Gilbert Pilz, and Tom Rutt. 12 February 2014. OASIS Committee Specification Draft 04 / Public Review Draft 02. http://docs.oasis-open.org/camp/camp-spec/v1.1/csprd02/camp-spec-v1.1-csprd02.html. Latest version: http://docs.oasis-open.org/camp/camp-spec/v1.1/camp-spec-v1.1.html.

Notices

Table of Contents

1 Introduction

1.1 Overview

Platform as a Service (PaaS) is a term that refers to a type of cloud computing in which the service provider offers customers/consumers access to one or more instances of a running application computing platform or application service stack. NIST defines PaaS [~~SP800-145~~SP800-145] as a “service model” with the following characteristics:

> The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations.

There are multiple commercial PaaS offerings in existence using languages such as Java, Python and Ruby and frameworks such as Spring, Django and Rails. Although these offerings differ in such aspects as programming languages, application frameworks, etc., there are inherent similarities in the way they manage the lifecycle of the applications that are targeted for, and deployed upon them. *The core proposition of this specification is that these similarities can be leveraged to produce a generic application and platform management API that is language, framework, and platform neutral.*

For PaaS consumers this management API would have the following benefits:

- “Portability between clouds” is emerging as one of the primary concerns of cloud computing. By standardizing the management API for the use cases around deploying, stopping, starting, and updating applications, this specification increases consumers’ ability to port their applications between PaaS offerings.
- It is likely that implementations of this specification will appear as plugins for application development environments (ADEs) and application management systems. Past experience has shown that, over time, such generic implementations are likely to receive more attention and be of higher quality than the implementations written for solitary, proprietary application management interfaces.

For PaaS providers this management API would have the following benefits:

- Because the strength and features of a PaaS offering’s application management API are unlikely to be perceived as key differentiators from other PaaS offerings, the existence of a consensus management API allows providers to leverage the experience and insight of the specification’s contributors and invest their design resources in other, more valuable areas.
- By increasing the portability of applications between PaaS offerings, this management API helps “grow the pie” of the PaaS marketplace by addressing one of the key pain points for PaaS consumers.

1.2 Purpose

This document defines the artifacts and APIs that need to be offered by a Platform as a Service (PaaS) cloud to manage the building, running, administration, monitoring and patching of applications in the cloud. Its purpose is to enable interoperability among self-service interfaces to PaaS clouds by defining artifacts and formats that can be used with any conforming cloud and enable independent vendors to create tools and services that interact with any conforming cloud using the defined interfaces. Cloud vendors can use these interfaces to develop new PaaS offerings that will interact with independently developed tools and components.

The following is a non-exhaustive list of the use cases which are supported by this specification.

- Building and packaging an application in a local Application Development Environment (ADE)
- Building an application in an ADE running in the cloud

- Importing a Platform Deployment Package into the cloud
- Uploading application artifacts into the cloud
- Run, stop, suspend, snapshot, and patch an application

1.3 Example (non-normative)

This example illustrates a scenario in which the application administrator wants to run and monitor an application. It assumes that the application package was previously made available to the platform, either because it was uploaded to the platform or developed directly on the platform.

The administrator ~~starts~~does this by deploying the application package to the platform. This is done by sending an HTTP POST request to the ~~platform entry point~~ URL of the *assemblies resource* as shown below, where "/~~myPaaS~~my_paas/assemblies" is ~~the entry point~~this URL and "/~~myPaas~~my_paas/pkgs/1" is the location of the application package.

```
POST /~~myPaaS~~my_paas/assemblies HTTP/1.1
Host: example.org
Content-Type: application/json
Content-Length: ~~...~~

~~{~~
~~  "pdpUri": "/myPaaS~~…

{
  "pdp_uri": "/my_paas/pkgs/1"
}
```

On receiving such a request the platform ~~deploys~~unpacks the application package, parses and ~~creates a new resource "/myPaas/templates/1" that represents~~validates the ~~deployed application. The response from~~deployment plan, resolves the ~~platform is show below.~~

```
~~HTTP/1.1 201 Created~~
~~Location: http://example.org/myPaaS/templates/1~~
~~Content-Type: ...~~
~~Content-Length: ...~~

~~...~~
```

~~Once the application is deployed, the administrator~~service dependencies described by that plan and starts the application ~~by sending an HTTP POST request to the resource that represents the deployed application, which was obtained in the previous step ("/myPaaS/templates/1").~~

```
~~POST /myPaaS/templates/1 HTTP/1.1~~
~~Host: example.org~~
```

. On successful start the platform creates a new resource representing the running application and provides the URL of that resource "/~~myPaaS~~my_paas/apps/1" in the response as shown below.

```
HTTP/1.1 201 Created
Location: http://example.org/~~myPaaS~~my_paas/apps/1
Content-Type: ~~...~~…
Content-Length: ~~...~~

~~...~~…

…
```

The administrator can now monitor the running application by sending an HTTP GET request to the resource that represents the running application, which was obtained in the previous step ("/~~myPaas~~my_paas/apps/1").

```
GET /~~myPaaS~~my_paas/apps/1 HTTP/1.1
Host: example.org
```

The response contains the JSON representation of the running application as shown below.

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: ~~....~~…

{
  "uri": "http://example.org/~~myPaaS~~my_paas/apps/1",
  "name": "Hello Cloud App",

  "type": "assembly",
  "description": "Hello Cloud Application Running in a PaaS Env",
  "~~applicationComponents~~components": [
    {
      "href": "/~~myPaaS~~my_paas/apps/1/acs/1", "~~targetName~~target_name":
"appComp1"
    },
    {
      "href": "~~myPaaS~~my_paas/apps/1/acs/2", "~~targetName~~target_name::
"appComp2"
    ~~}~~
~~  ],~~
~~  "platformComponents": [~~},
    {
      "href": "/~~myPaaS~~my_paas/pcs/1", "~~targetName~~target_name": "dbPlatComp"
    },
    {
      "href": "~~myPaaS~~my_paas/pcs/2", "~~targetName~~target_name": "msgBusPlatComp"
    }
  ~~],~~
~~  "assemblyTemplate": "/myPaaS/templates/1",~~]
}
```

1.4 Non-Goals

The interfaces exposed by the components and services in a PaaS system can be broadly split into two categories; functional interfaces and management interfaces. Functional interfaces are those that involve the specific utility provided by that component. For example, the interface used to submit a message to a message queuing service is as a functional interface. Management interfaces are those that deal with the administration of components. For example, the interface used to deploy and start an application on the platform is a management interface.

The specification of functional interfaces ~~specific to services provided by individual components (see Application Components and Platform Components, below) is~~ is out of scope for this document. ~~This is because such interfaces may be quite diverse and differ significantly from platform to platform.~~

~~1.5 PaaS Roles~~

1.5 Actors

There are many ~~roles that can be defined~~actors for a PaaS environment. For the purposes of this specification we identify the following ~~roles~~actors:

Application Developer: The person that builds and tests an application and presents the developed artifacts for deployment.

Application Administrator: The person that deploys applications and manages the application throughout its life-cycle.

Together these two ~~roles~~actors make up the consumers of the management API described in this specification. This specification is intended mainly for Application Administrators, though it does constraint the artifacts that an Application Developer presents for deployment.

Platform Administrator: The person that manages the platform. This specification describes some of the functions of a Platform Administrator, though most of the functions of this ~~role~~actor are outside its scope.

Application End-User: A user of an application deployed on the platform. The interactions of the Application end-user and the application are outside the scope of this specification.

Extension Developer: The person who creates new Extensions for Platforms.

1.6 Terminology

1.6.1 Term Definitions

1.6.1.1 CAMP Provider (Provider)

A CAMP Provider (Provider) is an implementation of the service aspects of this specification.

1.6.1.2 CAMP Consumer (Consumer)

A CAMP Consumer (Consumer) is an implementation of the client aspects of this specification.

1.6.2 Keywords, Conventions, and Normative Text

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119 [RFC2119].

Upper case versions of the RFC 2119 keywords are used in this document for emphasis. All text except examples, unless otherwise labeled, is normative. ~~All examples are non-~~Normative statements that use these keywords have been highlighted as per this sentence. Each such statement has been given a unique tag in the following manner: **[EX-00]**. For convenience these statements have been tabulated and cross-indexed by their tags and appear in Appendix C. All examples, figures and notes in this document are informative. Unless marked otherwise text in this specification is normative.

~~Three conformance targets are defined in this specification:~~

0. ~~A CAMP Provider, or “Provider” for short, is an implementation of the server aspects of this specification.~~
0. ~~A CAMP Consumer, or “Consumer” for short, is an implementation of the client aspects of this specification.~~
0. ~~Platform Deployment Package (PDP) defines the format of a deployment plan of an application and associated artifacts and meta-data.~~

See Section 8, “Conformance”, for details on Conformance to this specification.

1.7 Notational Conventions

The JSON and YAML descriptions that depict the representation of resources and the structure of Plans use a pseudo-schema notation with the following conventions:

- Characters are appended to items to indicate cardinality:
  - “?” (0 or 1)
  - “*” (0 or more)
  - “+” (1 or more)

  Absent any indication of cardinality, the default cardinality of an element is “exactly 1”. The scope of these operators is the entire line on which they appear.
- Vertical bars, “|”, denote choice. For example, “a | b” means a choice between “a” and “b”.
- Parentheses, “(“ and “)”, are used to indicate the scope of the “|” operator.

- An expression in italics indicates a value whose type is indicated by the italicized expression. For example, “foo: *String*” indicates that the value of “foo” will be a String.
- Square brackets, “[]”, indicate an array of the type indicated by the expression preceding it. For example, “foo: *String[]*” indicates that the value of “foo” will be an array of Strings.
- An expression surrounded in angle brackets, “<” and “>”, indicates a value whose type is indicated either by some of field in the object or by other context information. For example, “foo: <*sensor_type*>” indicates that the type of the value of “foo” is provided by the value of the “sensor_type” attribute.

Note that the information presented in pseudo-schema is intended as a condensed guide and is subordinate to the textual descriptions of the nodes and objects that appear in those descriptions. In the event of a conflict (due to a typo or other editorial error) the text takes precedence over the pseudo-schema.

~~1.7~~1.8 Specification Version

Each version of a CAMP specification is identified by a unique string termed the “Specification Version String”. The Specification Version String for this specification is “CAMP 1.1”.

~~1.7.1~~1.8.1 Backwards Compatibility

This version of the CAMP specification is not backwards compatible with any previous version of the CAMP specification.

~~1.8~~1.9 Normative References

[ISO 8601:2004] International Organization for Standardization, Geneva, Switzerland, “Data elements and interchange formats -- Information interchange - - Representation of dates and times”, March 2008. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40874

[OVF] DMTF DSP0243, “Open Virtualization Format Specification 2.0.1”, http://www.dmtf.org/sites/default/files/standards/documents/DSP0243_2.0.1.pdf

[RFC1952] Deutsch, P., "GZIP file format specification version 4.3", RFC 1952, May 1996. http://www.ietf.org/rfc/rfc1952.txt

[RFC2119] Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels”, BCP 14, RFC 2119, March 1997. http://www.ietf.org/rfc/rfc2119.txt.

[RFC2616] ~~R.~~ Fielding ~~et al, “~~, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer ~~Protcol –~~ Protocol -- HTTP/1.1~~”, IETF~~", RFC 2616, June 1999. ~~http://www.w3.org/Protocols/rfc2616/rfc2616.txt~~http://www.ietf.org/rfc/rfc2616.txt

~~**[RFC2617]** J. Franks et al, “HTTP Authentication: Basic and Digest Access Authentication”, IETF RFC 2617, June 1999. http://www.ietf.org/rfc/rfc2617.txt~~

~~[~~**[RFC2818]** E. Rescorla, “HTTP Over TLS”, RFC 2818, May 2000. http://www.ietf.org/rfc/rfc2818.txt

[RFC2388] Masinter, L., "Returning Values from Forms: multipart/form-data", RFC ~~**3986]**~~ ~~T.~~2388, August 1998. http://www.ietf.org/rfc/rfc2388.txt

[RFC3986] Berners-Lee ~~et al, “~~, T., Fielding, R., and L. Masinter, "Uniform Resource ~~Identifiers~~Identifier (URI): Generic Syntax~~”, IETF~~", STD 66, RFC 3986, ~~August 1998~~January 2005. http://www.ietf.org/rfc/rfc3986.txt

[RFC4346] ~~T.~~ Dierks, T. and E. Rescorla, ~~“~~"The Transport Layer Security (TLS) Protocol Version 1.~~2”, IETF~~1", RFC 4346, April 2006. http://www.ietf.org/rfc/rfc4346.txt

[RFC4627] ~~D.~~ Crockford, ~~“~~D., "The application/json Media Type for JavaScript Object Notation (JSON~~)”, IETF~~)", RFC 4627, July 2006. http://www.ietf.org/rfc/rfc4627.txt

[RFC5246] ~~T.~~ Dierks, T. and E. Rescorla, ~~“~~"The Transport Layer Security (TLS) Protocol Version 1.2~~”, IETF~~", RFC 5246, August 2008. http://www.ietf.org/rfc/rfc5246.txt

[RFC5789] ~~L.~~ Dusseault, ~~Linden Lab,~~L. and J. Snell, ~~“~~"PATCH Method for HTTP~~”, IETF~~", RFC 5789, March 2010. ~~http://www.ietf.org/rfc/rfc5789.txt~~http://www.ietf.org/rfc/rfc5789.txt

[RFC6902] Bryan, P., Ed., and M. Nottingham, Ed., "JavaScript Object Notation (JSON) Patch", RFC 6902, April 2013. http://www.ietf.org/rfc/rfc6902.txt

[SHA256] FIPS PUB 180-4, Federal Information Processing Standards Publication, “Secure Hash Standard (SHS) (FIPS PUB 180-4)”, 6.2, SHA-256. http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf

~~**[JSON Patch]** P. Bryan, M. Nottingham, “JSON Patch”, IETF Draft, January 2013. http://tools.ietf.org/html/draft-ietf-appsawg-json-patch-10~~

~~**[ISO 8601:2004]** International Organization for Standardization, Geneva, Switzerland, “Data elements and interchange formats -- Information interchange - - Representation of dates and times”, March 2008. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=40874~~

[TAR] IEEE Std 1003.1, 2004 Edition, Standard for Information Technology - Portable Operating System Interface (POSIX) http://ieeexplore.ieee.org/servlet/opac?punumber=9158

[YAML 1.1] Oren Ben-Kiki, Clark Evans, Brian Ingerson, “YAML Ain’t Markup Language (YAML) Version 1.1~~”, January~~, 2005~~.~~-01-18”. http://yaml.org/spec/1.1/. Also archived at http://xml.coverpages.org/yaml-spec-v1.1-archive-copy.html.

~~**[OVF]** Distributed Management Task Force, “Open Virtualization Format Specification”, DSP0243, 13 December 2012. http://www.dmtf.org/sites/default/files/standards/documents/DSP0243_2.0.0.pdf~~

~~**[TAR]** "IEEE Std 1003.1-2001, IEEE Standard for Information Technology - Portable Operating System Interface (POSIX)"~~

~~**[GZIP]** RFC-1952 "GZIP file format specification version 4.3," http://tools.ietf.org/html/rfc1952~~

[ZIP] ZIP File Format Specification, http://www.pkware.com/documents/APPNOTE/APPNOTE-6.3.0.TXT

1.10 ~~[SHA256] FIPS PUB 180-4, Federal Information Processing Standards Publication, “Secure Hash Standard (SHS),”~~Non-Normative References

[Git] The Software Freedom Conservancy, “Git, the fast version control system”, March 2012. http://git-scm.com/

[Keystone] OpenStack Foundation, “OpenStack Identity Service API v2.0 Reference”, July 2013. http://docs.openstack.org/api/openstack-identity-service/2.0/content/

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999. http://www.ietf.org/rfc/rfc2617.txt

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012. http://www.ietf.org/rfc/rfc6749.txt

~~http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf~~

~~1.81.1 Non-Normative References~~

[SP800-145] Peter Mell, Timothy Grance, “The NIST Definition of Cloud Computing”, Special Publication 800-145, September 2011. http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

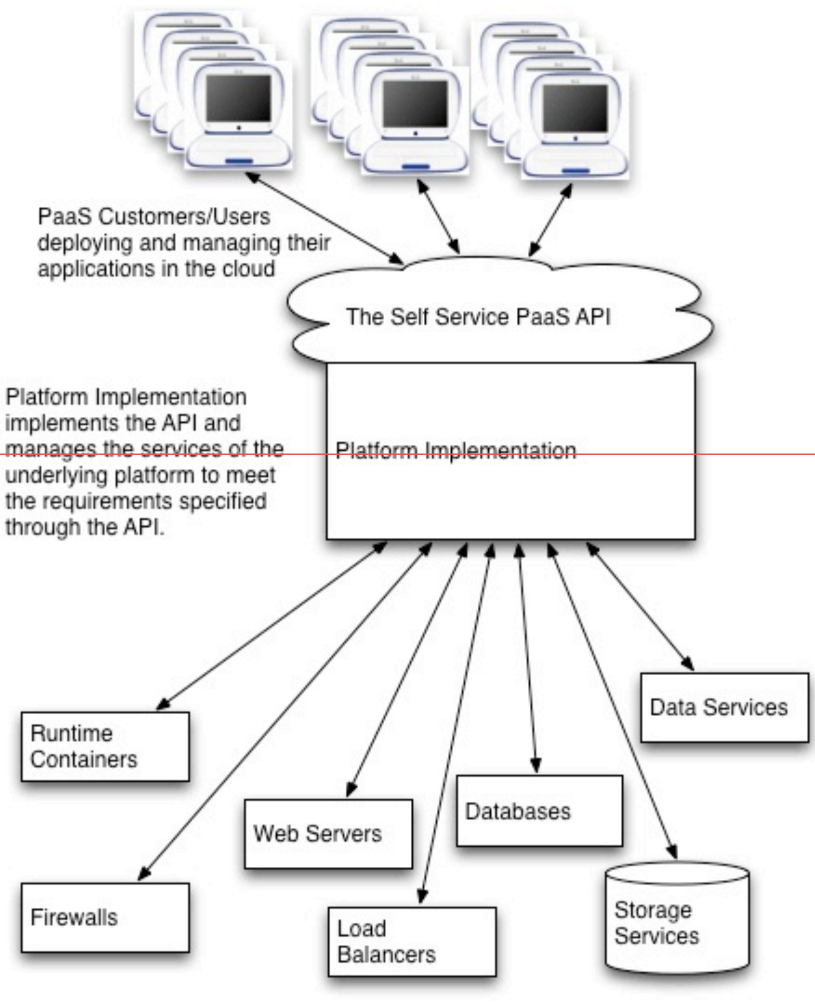
~~**[Git]** The Software Freedom Conservancy, “Git, the fast version control system”, March 2012. http://git-scm.com/~~

~~3~~2 Concepts and Types

This ~~document specifies~~section is informative.

2.1 Introduction

This specification defines the self-service management API that a Platform as a Service offering presents to the consumer of the platform. The API is ~~typically~~ the interface into a platform implementation layer that controls the deployment of applications and their use of the platform.



PaaS Customers/Users deploying and managing their applications in the cloud
The Self Service PaaS API
Platform Implementation implements the API and manages the services of the underlying platform to meet the requirements specified through the API.
Platform Implementation
Runtime Containers
Firewalls
Web Servers
Load Balancers
Databases
Storage Services
Data Services

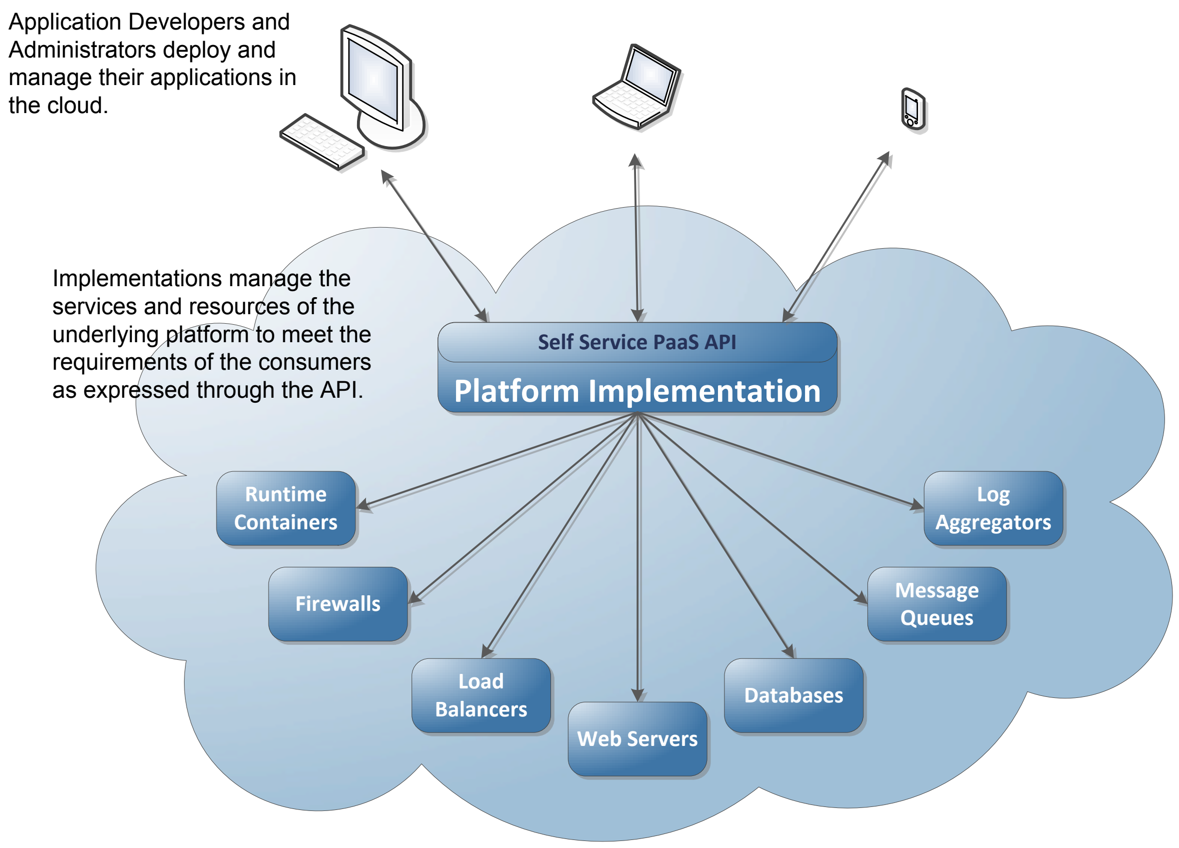

Application Developers and Administrators deploy and manage their applications in the cloud.
Implementations manage the services and resources of the underlying platform to meet the requirements of the consumers as expressed through the API.
Self Service PaaS API
Platform Implementation
Runtime Containers
Log Aggregators
Firewalls
Message Queues
Load Balancers
Databases
Web Servers

Figure 2-1: Typical PaaS Architecture

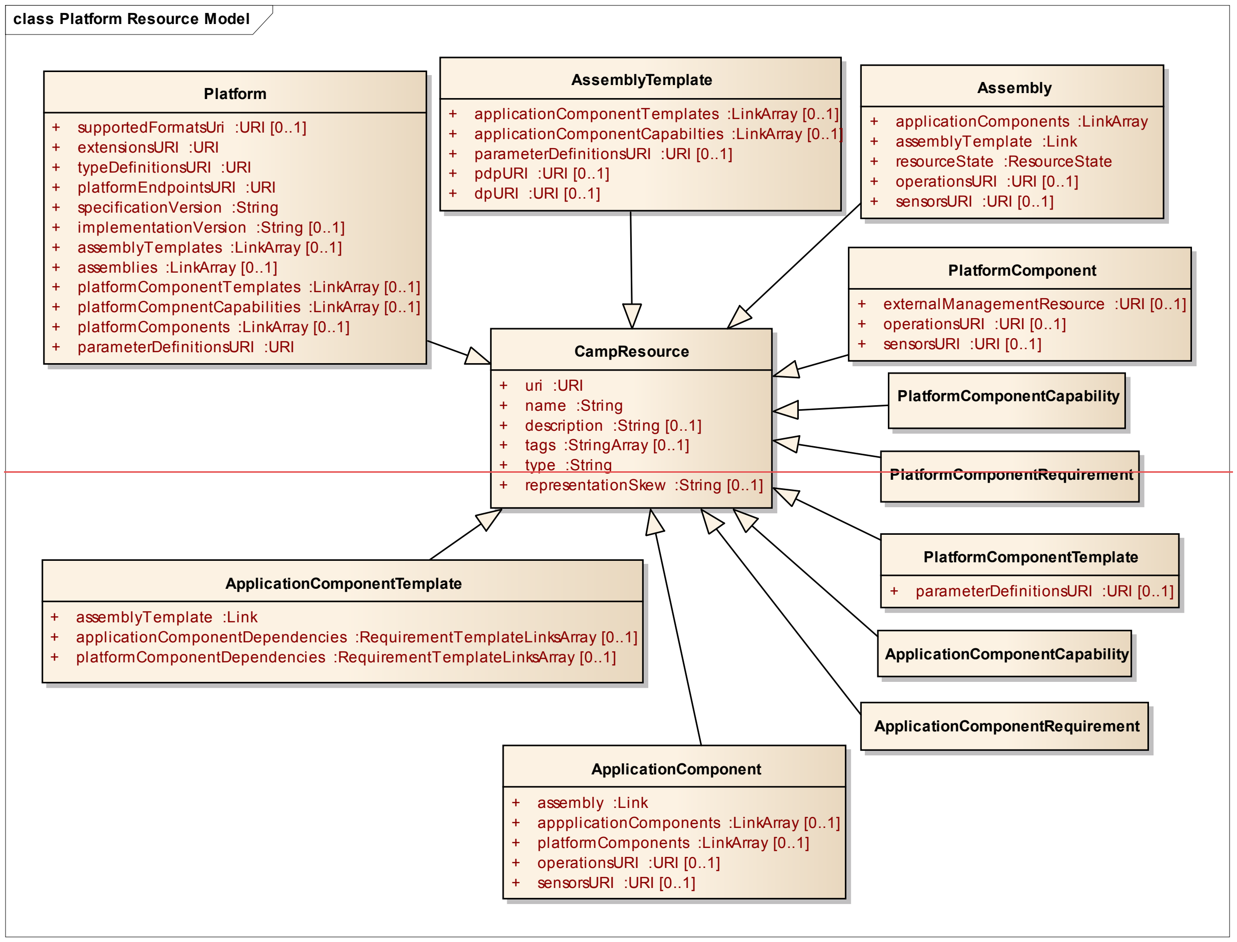
The figure above shows a typical architecture of a Platform as a Service cloud. The platform implementation is a management client of the underlying resources that transforms (through policies) the application requirements expressed by the Application Administrator into provisioning and other operations on those resources. The Platform Administrator manages the underlying hardware, storage, networks, and software services that make up the platform through existing administrative interfaces. Thus the Application Administrator is able to concentrate on their application and its deployment environment rather than having to be a systems administrator, database administrator and middleware administrator as well (as is the case with IaaS).

The goal of the management interface is to provide the PaaS consumer with a model that is as simple as possible, and yet still provides the essential elements that give them control over the deployment, execution, administration and metering of their application and its deployment environment.

~~3.1~~ Resources~~Application Assemblies~~

2.2

The ~~model~~CAMP API is made up of resources ~~manipulated through~~in a REST protocol. The resources represent elements of the ~~interface, in combination~~underlying system. The protocol enables interaction with the ~~protocol used to remotely accomplish this, constitutes~~resources. The following are the ~~self-service PaaS management API. The model contains resource types corresponding to~~main resources in the ~~artifacts discussed earlier.~~API:


class Platform Resource Model
Platform
+ supportedFormatsUri :URI [0..1]
+ extensionsURI :URI
+ typeDefinitionsURI :URI
+ platformEndpointsURI :URI
+ specificationVersion :String
+ implementationVersion :String [0..1]
+ assemblyTemplates :LinkArray [0..1]
+ assemblies :LinkArray [0..1]
+ platformComponentTemplates :LinkArray [0..1]
+ platformCompnentCapabilities :LinkArray [0..1]
+ platformComponents :LinkArray [0..1]
+ parameterDefinitionsURI :URI
AssemblyTemplate
+ applicationComponentTemplates :LinkArray [0..1]
+ applicationComponentCapabilties :LinkArray [0..1]
+ parameterDefinitionsURI :URI [0..1]
+ pdpURI :URI [0..1]
+ dpURI :URI [0..1]
Assembly
+ applicationComponents :LinkArray
+ assemblyTemplate :Link
+ resourceState :ResourceState
+ operationsURI :URI [0..1]
+ sensorsURI :URI [0..1]
PlatformComponent
+ externalManagementResource :URI [0..1]
+ operationsURI :URI [0..1]
+ sensorsURI :URI [0..1]
CampResource
+ uri :URI
+ name :String
+ description :String [0..1]
+ tags :StringArray [0..1]
+ type :String
+ representationSkew :String [0..1]
PlatformComponentCapability
PlatformComponentRequirement
PlatformComponentTemplate
+ parameterDefinitionsURI :URI [0..1]
ApplicationComponentTemplate
+ assemblyTemplate :Link
+ applicationComponentDependencies :RequirementTemplateLinksArray [0..1]
+ platformComponentDependencies :RequirementTemplateLinksArray [0..1]
ApplicationComponentCapability
ApplicationComponentRequirement
ApplicationComponent
+ assembly :Link
+ appplicationComponents :LinkArray [0..1]
+ platformComponents :LinkArray [0..1]
+ operationsURI :URI [0..1]
+ sensorsURI :URI [0..1]

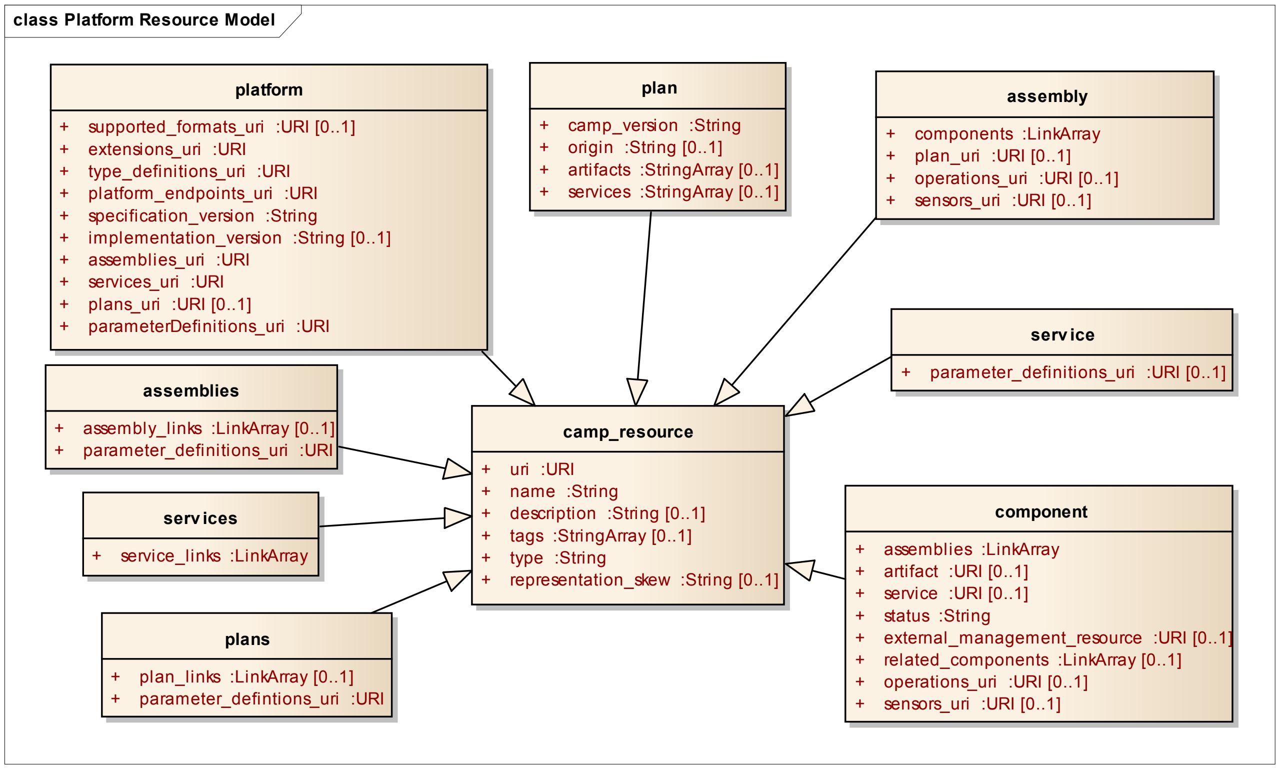

class Platform Resource Model
platform
+ supported_formats_uri :URI [0..1]
+ extensions_uri :URI
+ type_definitions_uri :URI
+ platform_endpoints_uri :URI
+ specification_version :String
+ implementation_version :String [0..1]
+ assemblies_uri :URI
+ services_uri :URI
+ plans_uri :URI [0..1]
+ parameterDefinitions_uri :URI
plan
+ camp_version :String
+ origin :String [0..1]
+ artifacts :StringArray [0..1]
+ services :StringArray [0..1]
assembly
+ components :LinkArray
+ plan_uri :URI [0..1]
+ operations_uri :URI [0..1]
+ sensors_uri :URI [0..1]
service
+ parameter_definitions_uri :URI [0..1]
assemblies
+ assembly_links :LinkArray [0..1]
+ parameter_definitions_uri :URI
camp_resource
+ uri :URI
+ name :String
+ description :String [0..1]
+ tags :StringArray [0..1]
+ type :String
+ representation_skew :String [0..1]
services
+ service_links :LinkArray
component
+ assemblies :LinkArray
+ artifact :URI [0..1]
+ service :URI [0..1]
+ status :String
+ external_management_resource :URI [0..1]
+ related_components :LinkArray [0..1]
+ operations_uri :URI [0..1]
+ sensors_uri :URI [0..1]
plans
+ plan_links :LinkArray [0..1]
+ parameter_defintions_uri :URI

Figure 2-2: CAMP Resources as UML Classes

Figure 2-2 is a UML Class Diagram showing the CAMP resources as UML classes. All CAMP resources share a set of common attributes ~~shown here as a CampResource~~which they inherit from the *camp_resource* parent class.

Each attribute shown in these UML class diagrams has a CAMP common attribute type. The '+' symbol before each attribute name in the boxes indicates that the attribute access is public (i.e. available thru the API). Non-mandatory resource attributes are indicated using the [0..1] UML multiplicity tag.

~~3.1.1~~2.2.1 Platform

The ~~Platform~~*platform resource* ~~resource~~ is the primary view of the platform and what is running on it. The ~~Platform~~*platform resource* references ~~templates for creating~~collections of resources that represent the services provided by the platform (as Services), the applications ~~(as Assembly Templates)~~running on this platform (as assembly Resources), as well as ~~running applications (as Assemblies) and enables discovery of the PaaS offering in terms~~collections of ~~Platform Components and their capabilities.~~metadata resources that describe the resources supported by the platform as well as any extensions that the Provider has implemented. The ~~Platform~~*platform resource* also determines the scope of access for sharing amongst multiple applications.

~~3.1.2~~2.2.2 Assemblies

An ~~Assembly~~*assembly* ~~resources represent~~ *resource* represents running applications. Operations on an ~~Assembly~~*assembly resource* affect the components and elements of that application. ~~Assembly Template resources are templates for the creation of Assemblies.~~

~~3.1.3~~2.2.3 Components

~~There are two kinds of components, application components and platform components, each of which can exist in either template or instantiated forms.~~

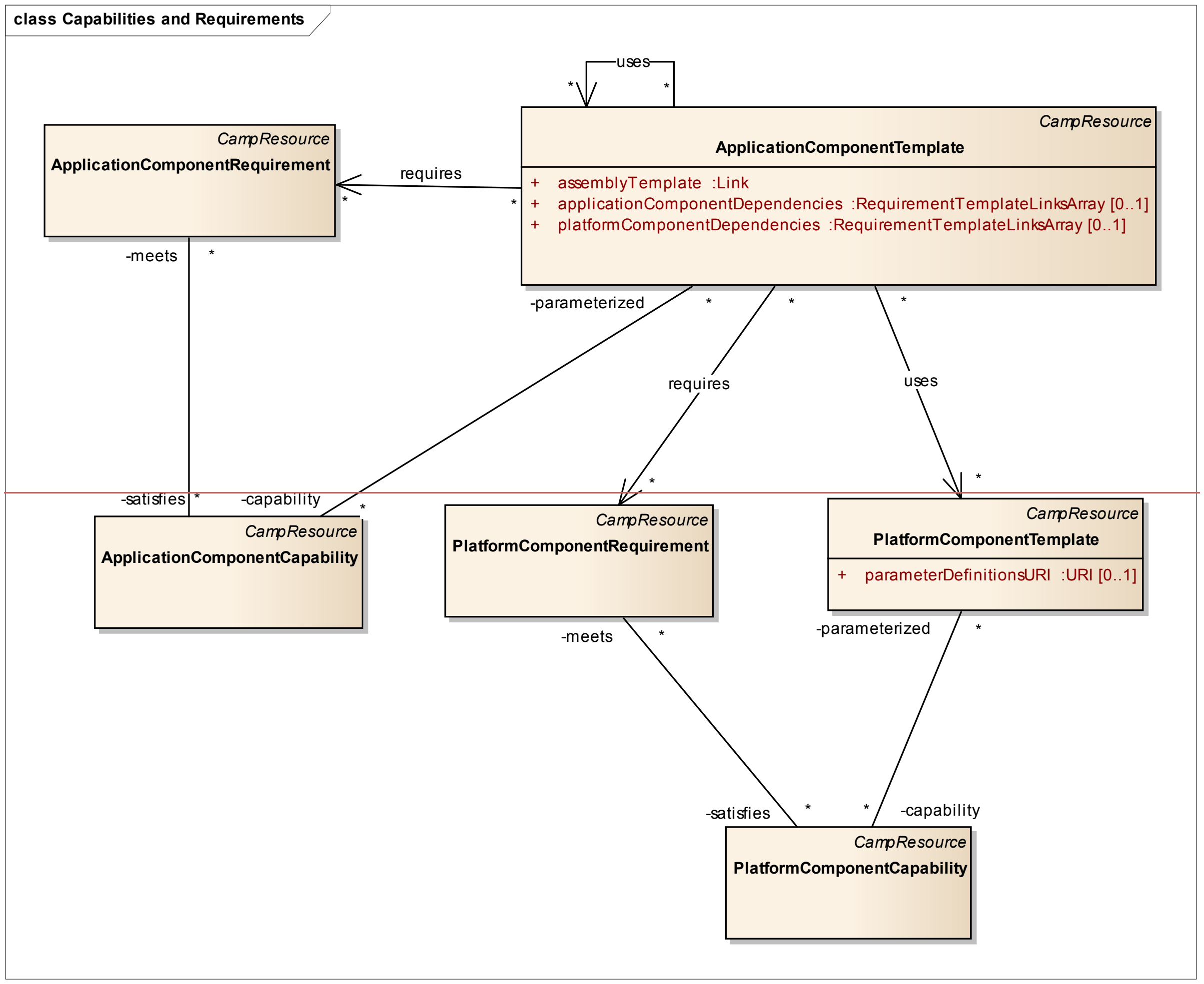
~~The Application Component Template resource represents a discrete configuration of a part of an Assembly Template supplied by the Application Administrator. The attributes of this~~An *assembly resource* ~~represent the configuration values that will be applied to the~~ is comprised of one or more *component* ~~upon instantiation.~~

~~The Application Component~~*resources*. A *component resource* represents ~~an instantiated instance~~a discrete and, in most cases, dynamic element of an ~~Application Component Template,~~application such as such as a deployed ~~WAR file.~~

~~The Platform Component Template resource represents a discrete configuration of a part of an Assembly Template supplied by the Platform. The attributes of this resource represent the configuration values that will be applied to the component upon instantiation.~~

~~The Platform Component resource represents an instantiated instance of a Platform Component Template in use by an application.~~

~~3.1.3 Capabilities and Requirements~~


class Capabilities and Requirements
CampResource
ApplicationComponentRequirement
CampResource
ApplicationComponentTemplate
+ assemblyTemplate :Link
+ applicationComponentDependencies :RequirementTemplateLinksArray [0..1]
+ platformComponentDependencies :RequirementTemplateLinksArray [0..1]
uses
requires
-meets
-parameterized
requires
uses
-satisfies
-capability
CampResource
ApplicationComponentCapability
CampResource
PlatformComponentRequirement
CampResource
PlatformComponentTemplate
+ parameterDefinitionsURI :URI [0..1]
-meets
-parameterized
-satisfies
-capability
CampResource
PlatformComponentCapability

~~*Figure 2-3: Capabilities, Requirements, and Template Relationships*~~

~~Figure 2-3 shows the relationships between capabilities, requirements, and template resources. Associations which are visible thru pointer attributes in resources (i.e. URI, Link~~Ruby gem, a database instance, or ~~LinksArray attribute types) are show using UML named associations with navigation arrows. Associations which model implementation specific relationships, not visible thru the API, are represented using the UML association end notation, without navigation arrows. The association end names were chosen to represent the role that resource plays in the relationship. The '–' symbol before the association end names expresses that they have private access (i.e. navigation across resource links is not available thru the API).~~

~~Like Templates, Capability resources represent the configuration of instantiable Components (Application Components or Platform Components). Unlike Templates, which delineate discrete configurations, Capabilities specify ranges of configuration values.~~

~~Requirement resources are created by the Application Developer or Application Administrator to express an application's dependency on a~~ a set of entries in a LDAP directory. A *component* ~~that is capable of satisfying a certain set of requirements. For example, an application~~ *resource* can be related to other *component* ~~might depend upon a messaging service that supports a certain version of an AMQP API, can accept messages of up to 2MB in size, and which provides a persistent message store~~*resources* through producer/consumer or other kinds of relationships.

~~The process of matching Requirements with Capabilities is referred to as "requirement resolution".~~

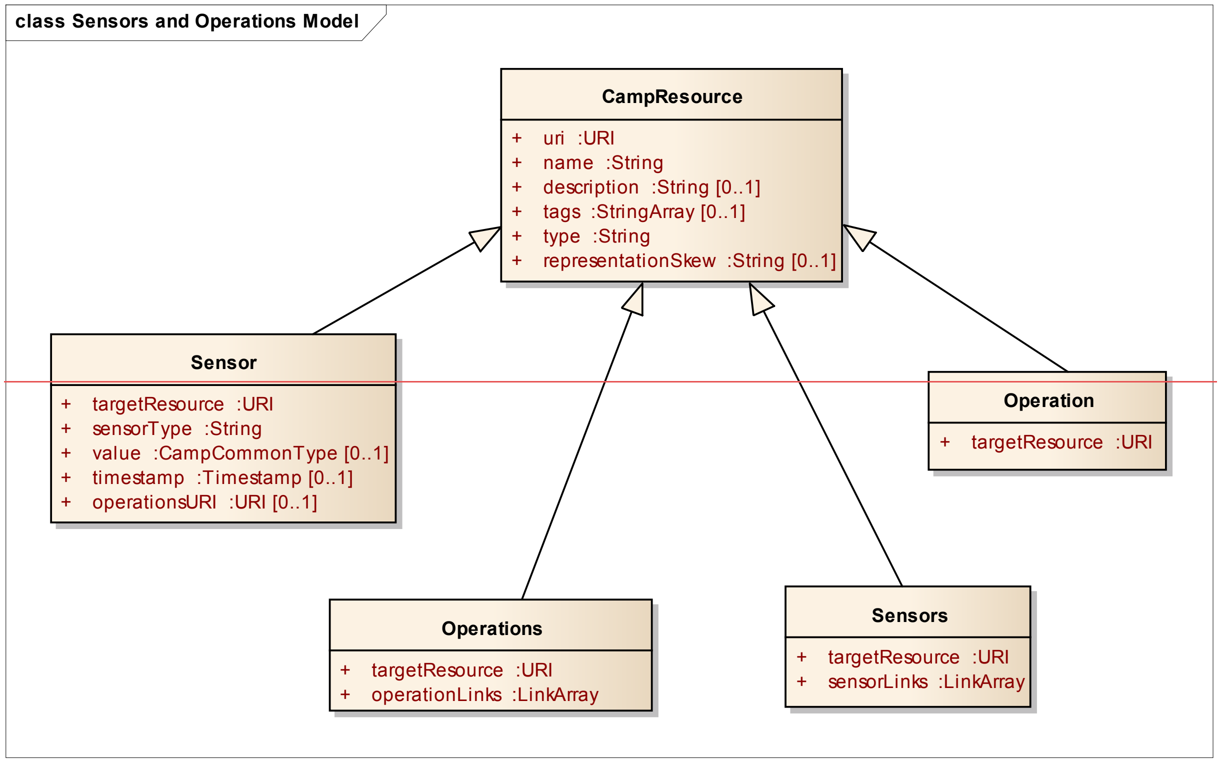
2.2.4 Plans

A Plan is meta-data that provides a description of the artifacts that make up an application, the services that are required to execute or utilize those artifacts, and the relationship of the artifacts to those services. Plans can be expressed in two forms; either as a YAML file or, optionally, as a CAMP resource. The Artifacts described in a Plan represent discrete, static elements of an application such as a Ruby gem file, an SQL script, or a PKCS #12 file.

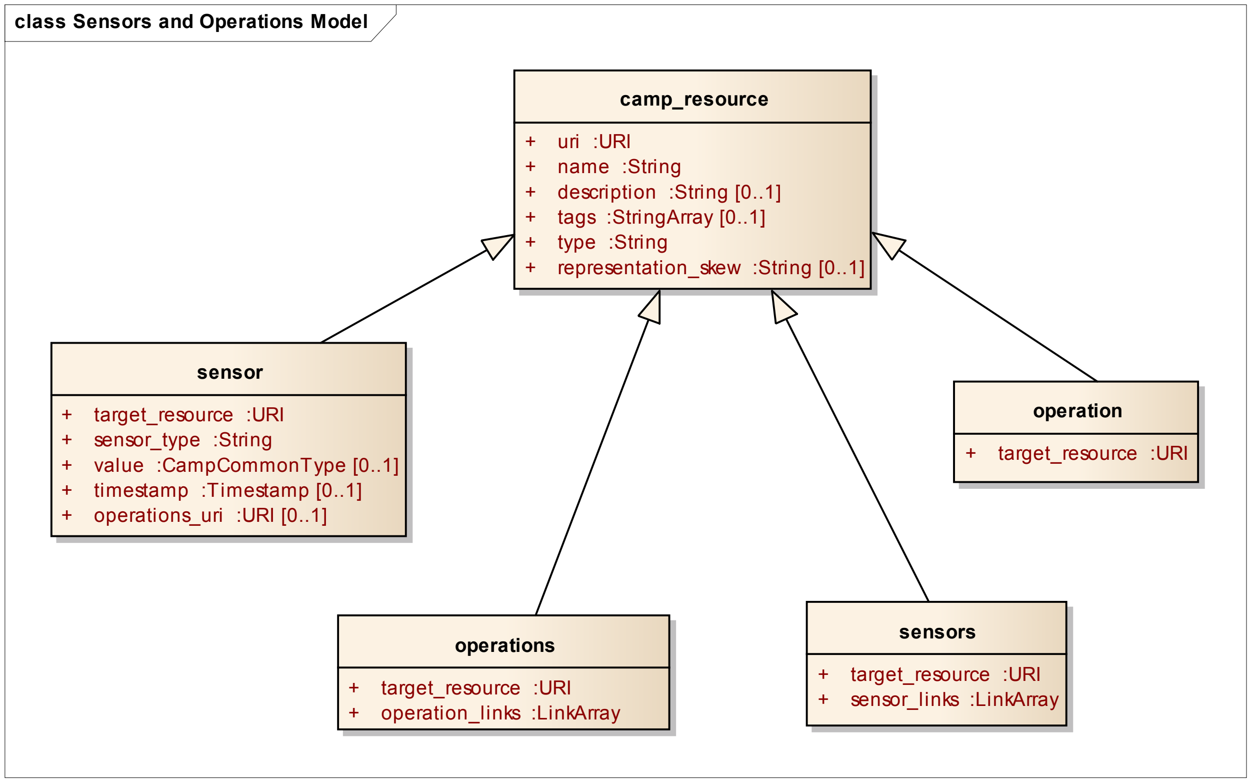
2.2.5 Services

A *service resource* represents a blueprint for creating *component resources* that utilize or embody a platform-provided service in some way. For example, a Service may represent the platform's ability to create a message queue for use by an application.

~~3.1.5~~2.2.6 Operations and Sensors



class Sensors and Operations Model
CampResource
+ uri :URI
+ name :String
+ description :String [0..1]
+ tags :StringArray [0..1]
+ type :String
+ representationSkew :String [0..1]
Sensor
+ targetResource :URI
+ sensorType :String
+ value :CampCommonType [0..1]
+ timestamp :Timestamp [0..1]
+ operationsURI :URI [0..1]
Operation
+ targetResource :URI
Operations
+ targetResource :URI
+ operationLinks :LinkArray
Sensors
+ targetResource :URI
+ sensorLinks :LinkArray



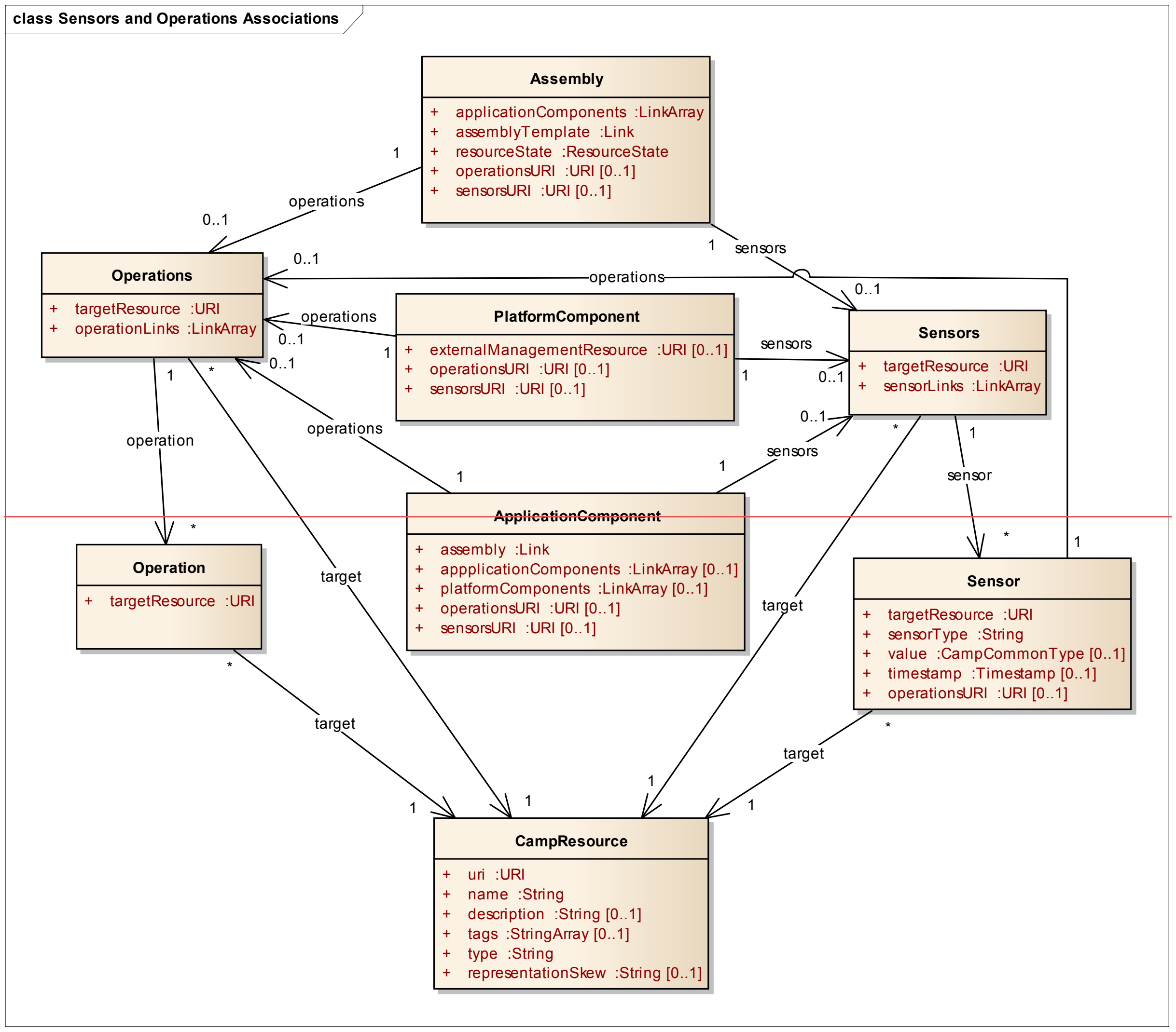
class Sensors and Operations Model
camp_resource
+ uri :URI
+ name :String
+ description :String [0..1]
+ tags :StringArray [0..1]
+ type :String
+ representation_skew :String [0..1]
sensor
+ target_resource :URI
+ sensor_type :String
+ value :CampCommonType [0..1]
+ timestamp :Timestamp [0..1]
+ operations_uri :URI [0..1]
operation
+ target_resource :URI
operations
+ target_resource :URI
+ operation_links :LinkArray
sensors
+ target_resource :URI
+ sensor_links :LinkArray

Figure 2-3: Operations and Sensors

Figure 2-3 is a UML class diagram showing the attributes of the ~~Operations~~*operation resources* and ~~Sensors~~*sensor resources*.

Operations and Sensors provide a way of interacting with an application through the CAMP API. ~~Operation resources represent~~An *operation resource* represents actions that can be taken on a resource, and ~~Sensor~~*sensor resources* represent dynamic data about resources, such as metrics or state. ~~Sensor resources are~~A *sensor resource* is useful for exposing data that changes rapidly, or that might need to be fetched from a secondary system. ~~Sensor Resources~~A *sensor resource* can also offer Operations to allow resetting metrics, or adjusting frequency collection settings.

~~Operation and Sensor~~Multiple *operation resources* ~~are~~and *sensor resources* can be exposed both on ~~Assembly resources, Application Component~~*assembly resources* and ~~Platform Component~~*component resources*. Operations are also known as effectors. The combination of ~~sensors~~Operations and ~~operations~~Sensors enables ongoing management. This can include automation techniques such as using policies, event-condition-action paradigms, or autonomic control. A Consumer can use the REST API to perform such management. A Provider can also use them. For example, a ~~Platform Component~~*component resource* could be offered that allows for “autoscaling” capacity based on the volume of work an application performs.


class Sensors and Operations Associations
Assembly
+ applicationComponents :LinkArray
+ assemblyTemplate :Link
+ resourceState :ResourceState
+ operationsURI :URI [0..1]
+ sensorsURI :URI [0..1]
Operations
+ targetResource :URI
+ operationLinks :LinkArray
PlatformComponent
+ externalManagementResource :URI [0..1]
+ operationsURI :URI [0..1]
+ sensorsURI :URI [0..1]
Sensors
+ targetResource :URI
+ sensorLinks :LinkArray
ApplicationComponent
+ assembly :Link
+ appplicationComponents :LinkArray [0..1]
+ platformComponents :LinkArray [0..1]
+ operationsURI :URI [0..1]
+ sensorsURI :URI [0..1]
Operation
+ targetResource :URI
Sensor
+ targetResource :URI
+ sensorType :String
+ value :CampCommonType [0..1]
+ timestamp :Timestamp [0..1]
+ operationsURI :URI [0..1]
CampResource
+ uri :URI
+ name :String
+ description :String [0..1]
+ tags :StringArray [0..1]
+ type :String
+ representationSkew :String [0..1]
operations
sensors
operation
sensor
target
1
0..1
*

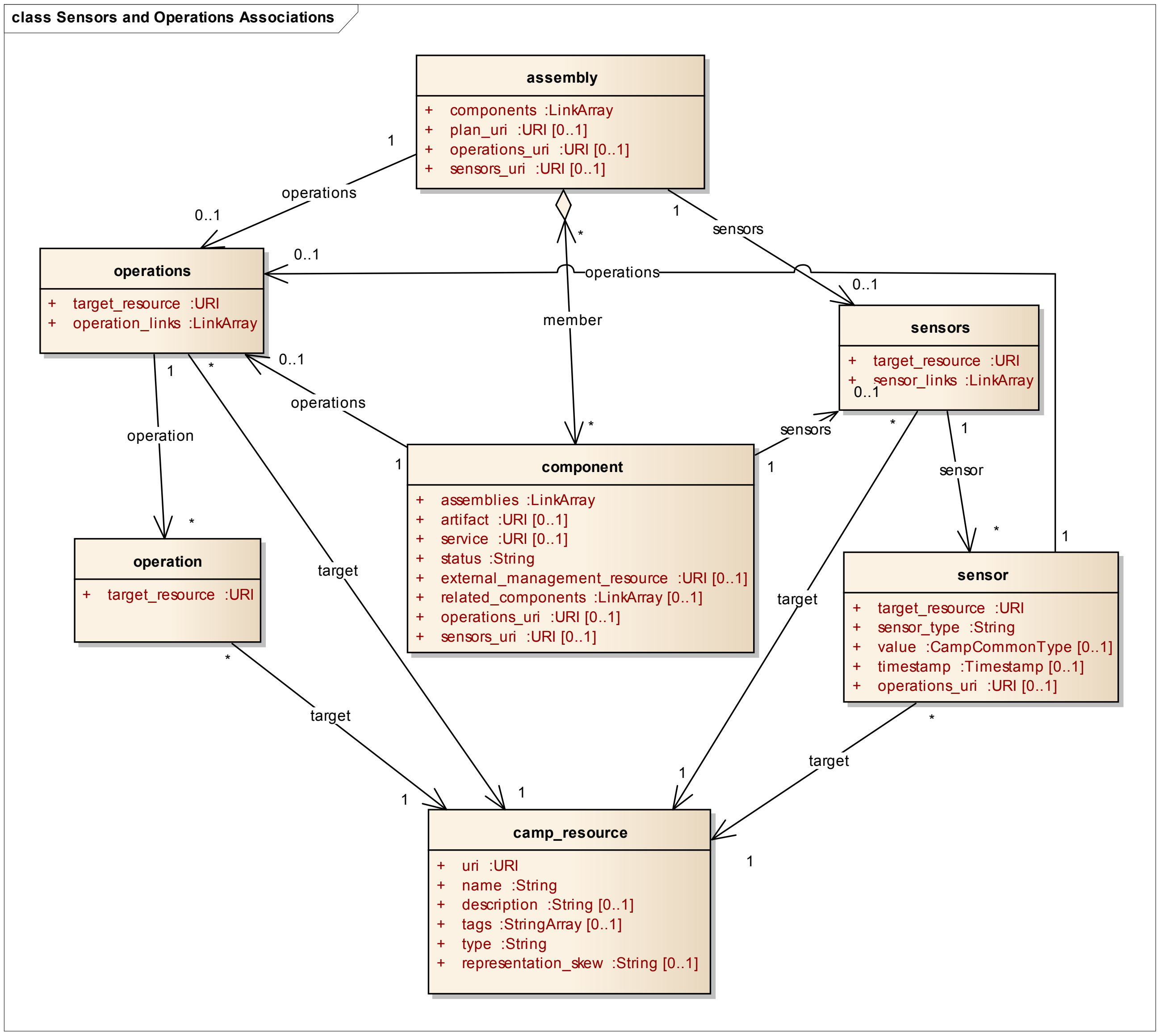
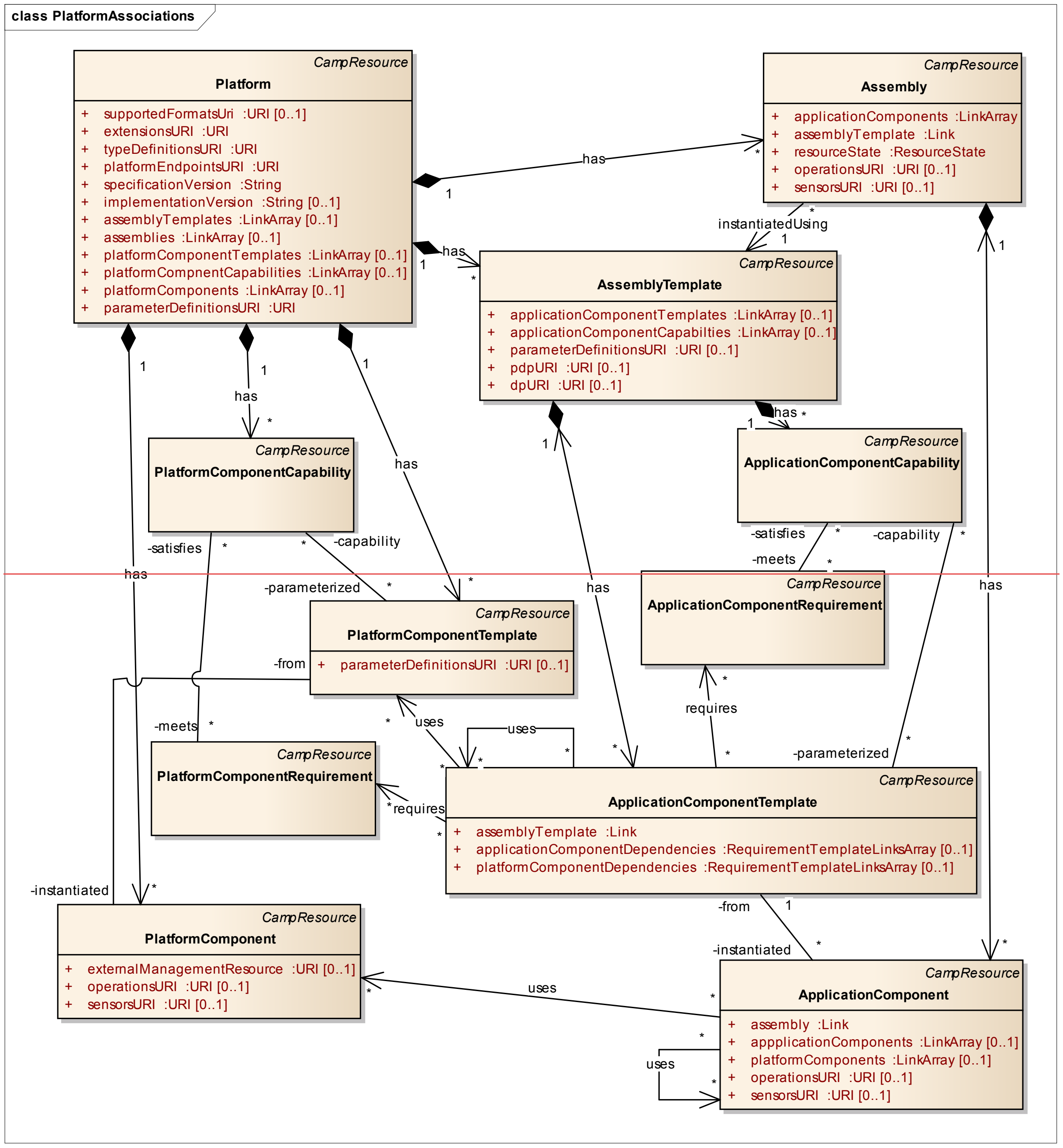

class Sensors and Operations Associations
assembly
+ components :LinkArray
+ plan_uri :URI [0..1]
+ operations_uri :URI [0..1]
+ sensors_uri :URI [0..1]
operations
+ target_resource :URI
+ operation_links :LinkArray
sensors
+ target_resource :URI
+ sensor_links :LinkArray
component
+ assemblies :LinkArray
+ artifact :URI [0..1]
+ service :URI [0..1]
+ status :String
+ external_management_resource :URI [0..1]
+ related_components :LinkArray [0..1]
+ operations_uri :URI [0..1]
+ sensors_uri :URI [0..1]
operation
+ target_resource :URI
sensor
+ target_resource :URI
+ sensor_type :String
+ value :CampCommonType [0..1]
+ timestamp :Timestamp [0..1]
+ operations_uri :URI [0..1]
camp_resource
+ uri :URI
+ name :String
+ description :String [0..1]
+ tags :StringArray [0..1]
+ type :String
+ representation_skew :String [0..1]
operations
sensors
member
operation
sensor
target

Figure 2-4: Operations and Sensors Associations

Figure 2-4 is a UML class diagram showing ~~Operations~~*operation resources* and ~~Sensors~~*sensors resources*, and the other CAMP resources that they are associated with.

~~3.1.6~~2.2.7 Resource Relationships


class PlatformAssociations
CampResource
Platform
+ supportedFormatsUri :URI [0..1]
+ extensionsURI :URI
+ typeDefinitionsURI :URI
+ platformEndpointsURI :URI
+ specificationVersion :String
+ implementationVersion :String [0..1]
+ assemblyTemplates :LinkArray [0..1]
+ assemblies :LinkArray [0..1]
+ platformComponentTemplates :LinkArray [0..1]
+ platformCompnentCapabilities :LinkArray [0..1]
+ platformComponents :LinkArray [0..1]
+ parameterDefinitionsURI :URI
CampResource
Assembly
+ applicationComponents :LinkArray
+ assemblyTemplate :Link
+ resourceState :ResourceState
+ operationsURI :URI [0..1]
+ sensorsURI :URI [0..1]
has
instantiatedUsing
CampResource
AssemblyTemplate
+ applicationComponentTemplates :LinkArray [0..1]
+ applicationComponentCapabilties :LinkArray [0..1]
+ parameterDefinitionsURI :URI [0..1]
+ pdpURI :URI [0..1]
+ dpURI :URI [0..1]
CampResource
PlatformComponentCapability
CampResource
ApplicationComponentCapability
-satisfies
-capability
-meets
CampResource
ApplicationComponentRequirement
-parameterized
CampResource
PlatformComponentTemplate
+ parameterDefinitionsURI :URI [0..1]
-from
uses
requires
CampResource
PlatformComponentRequirement
CampResource
ApplicationComponentTemplate
+ assemblyTemplate :Link
+ applicationComponentDependencies :RequirementTemplateLinksArray [0..1]
+ platformComponentDependencies :RequirementTemplateLinksArray [0..1]
-instantiated
CampResource
PlatformComponent
+ externalManagementResource :URI [0..1]
+ operationsURI :URI [0..1]
+ sensorsURI :URI [0..1]
CampResource
ApplicationComponent
+ assembly :Link
+ appplicationComponents :LinkArray [0..1]
+ platformComponents :LinkArray [0..1]
+ operationsURI :URI [0..1]
+ sensorsURI :URI [0..1]

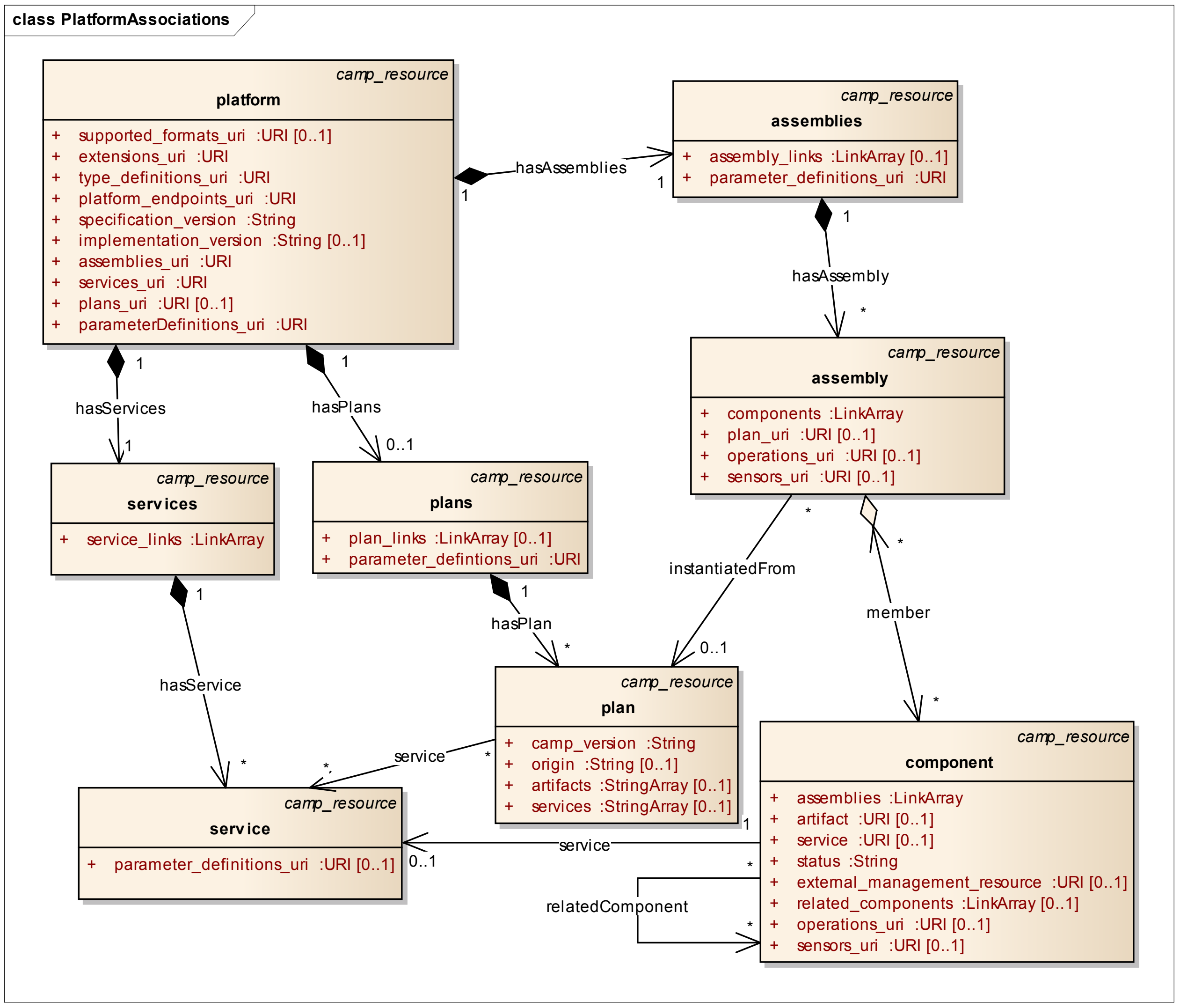

class PlatformAssociations
camp_resource
platform
+ supported_formats_uri :URI [0..1]
+ extensions_uri :URI
+ type_definitions_uri :URI
+ platform_endpoints_uri :URI
+ specification_version :String
+ implementation_version :String [0..1]
+ assemblies_uri :URI
+ services_uri :URI
+ plans_uri :URI [0..1]
+ parameterDefinitions_uri :URI
camp_resource
assemblies
+ assembly_links :LinkArray [0..1]
+ parameter_definitions_uri :URI
hasAssemblies
hasAssembly
camp_resource
assembly
+ components :LinkArray
+ plan_uri :URI [0..1]
+ operations_uri :URI [0..1]
+ sensors_uri :URI [0..1]
hasServices
hasPlans
camp_resource
services
+ service_links :LinkArray
camp_resource
plans
+ plan_links :LinkArray [0..1]
+ parameter_defintions_uri :URI
instantiatedFrom
member
hasPlan
hasService
camp_resource
plan
+ camp_version :String
+ origin :String [0..1]
+ artifacts :StringArray [0..1]
+ services :StringArray [0..1]
camp_resource
component
+ assemblies :LinkArray
+ artifact :URI [0..1]
+ service :URI [0..1]
+ status :String
+ external_management_resource :URI [0..1]
+ related_components :LinkArray [0..1]
+ operations_uri :URI [0..1]
+ sensors_uri :URI [0..1]
service
camp_resource
service
+ parameter_definitions_uri :URI [0..1]
service
relatedComponent

Figure 2-5: Platform Resource Relationships

Figure 2-5 shows the relationships between Platform Resources using a UML class diagram.

Associations which are visible thru pointer attributes in resources (i.e. URI, Link, or LinkArray attribute types) are shown using UML named associations with navigation arrows.

Associations which model implementation specific relationships, not visible thru the API, are represented using the UML association end notation, without navigation arrows. The '–' symbol on these association ends expresses that access is private (i.e. navigation using resource links is not available thru the API).

Strict aggregation (i.e. "has" relationship) is indicated using a solid diamond on the association end attached to the owning resource. This implies that the owned resource cannot exist independent of its owner.

~~A Platform provides a set of Platform Components that can be used by the invoking applications. Examples of Platform Components include servlet containers, web servers, LDAP stores, and database instances. Platforms can also provide higher-level business components such as a business rules manager to gain competitive advantage and developer loyalty. The Platform Administrator manages and operates the implementation of Platform Components.~~

~~An application is composed of a set of Application Components that depend on one or more Platform Components. Examples of Application Components include Ruby gems, Java libraries, and PHP modules. Application Components can also include non-code artifacts such as datasets and collections of identity information.~~

~~Application Components can also interact with other Application Components. Thus, an Application Component has two different sets of dependencies. It depends on the Components provided by the platform, and depends on services provided by other Application Components. Such Application Components can be on the same platform or can reside at some other location. The Assembly resource is used to aggregate the management of these components as shown in Figure 2-6.~~

~~Platform Components have a set of Platform Component Capabilities that an application can choose from to meet its requirements. Applications can tailor Platform Component Requirements (a refinement or narrowing of the configuration ranges in the Capabilities) to meet their needs based on the range of parameters expressed in the Platform Component Capability.~~

~~The relationships of an Assembly Template to Application Component Templates and Platform Component Templates, or their Requirements are shown in Figure 2-6.~~

~~An Application Component can express the exact configuration of its dependency on other Components using one of the Component Template resources (either an Application Component Template or Platform Component Template). Alternatively, it can express a range of configuration values that are acceptable for that dependency by using one of the Component Requirement resources (either an Application Component Requirement or Platform Component Requirement). This might be done, for example, in an ADE when the existing Component Templates are not known. During the deployment, these Requirements are matched with Capabilities that have attribute values that fall within the ranges specified by the Requirements.~~

~~An Application Component Template cannot be instantiated unless all of its dependencies are satisfied. An Application Component Template SHALL be referenced by a single Assembly Template.~~ ~~**[CO-01]**~~ ~~An Assembly Template SHALL NOT be instantiated until all of its Application Component Templates are successfully instantiated.~~ ~~**[CO-02]**~~

~~3.9~~2.3 Deployment

~~A Deployment Plan (DP) is packaging management meta-data that provides a description of the artifacts that make up an application, the services that are required to execute or utilize those artifacts, and the relationship of the artifacts to those services.~~

A Platform Deployment Package (PDP) is an archive containing ~~the DP~~a Plan file together with application content files such as web archives, database schemas, scripts, source code, localization bundles, and icons; and metadata files such as manifests, checksums, signatures, and certificates. It can be used to move an Application and its Components from Platform to Platform, or between an Application Development Environment and a Platform.

In the simplest case (an example of which is provided Section 1.3, “Example”), a PDP or a Plan file can be used to create an *assembly resource* by transmitting an HTTP POST request containing either the PDP or the Plan file to the *assemblies resource*.

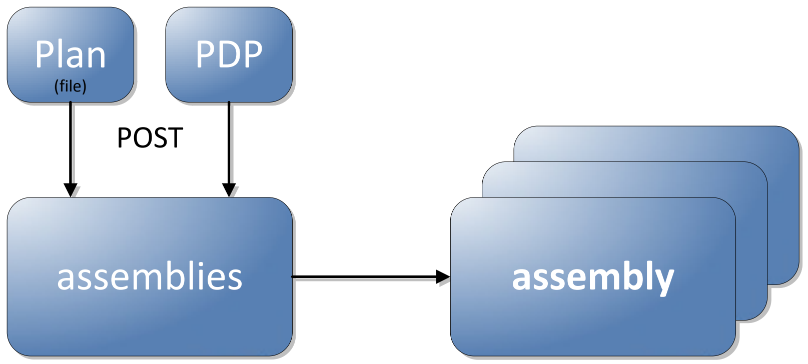
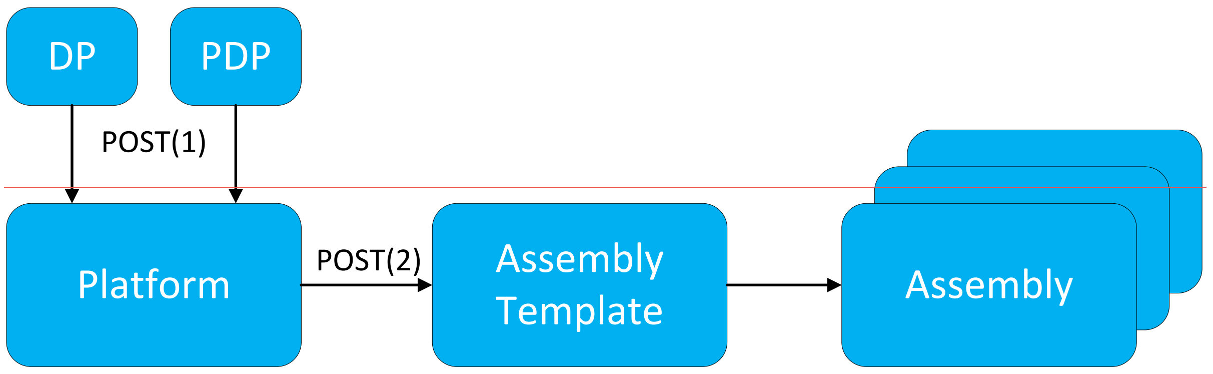


Figure 2-6: Deploying an application

On platforms that choose to support Plans, ~~A~~a CAMP Consumer can create a ~~new Assembly Template~~*plan resource* by uploading either a PDP or a ~~DP~~Plan file to the ~~Platform~~*plans resource* URI using an HTTP POST request. ~~Assembly resources are created from the Assembly Templates.~~ An ~~Assembly resource represents a running instance of an application. An Assembly~~*assembly resource* can then be created ~~using~~from the *plan resource* by including a reference to the *plan resource* in an HTTP POST request to the ~~URI of an Assembly Template~~*assemblies resource*.



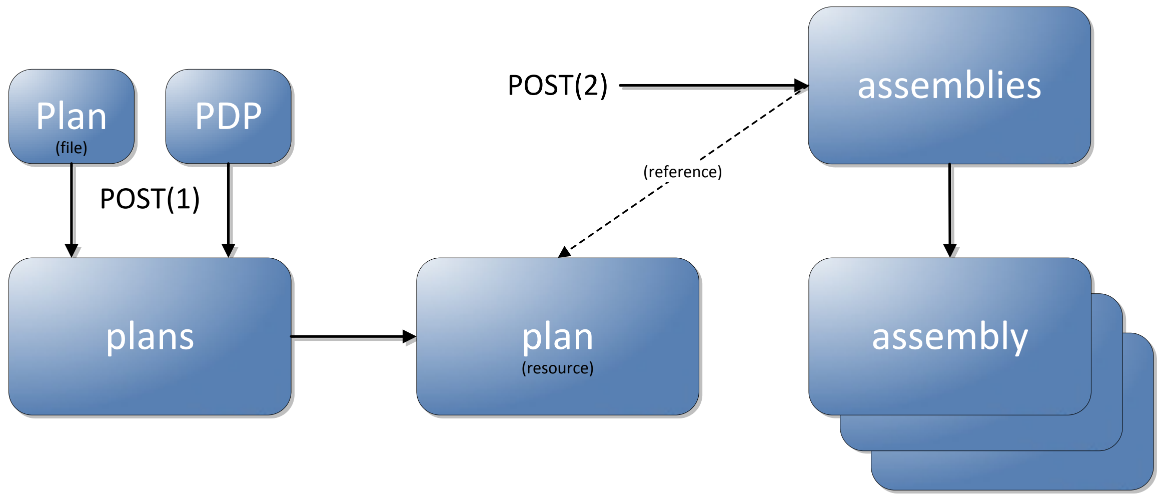
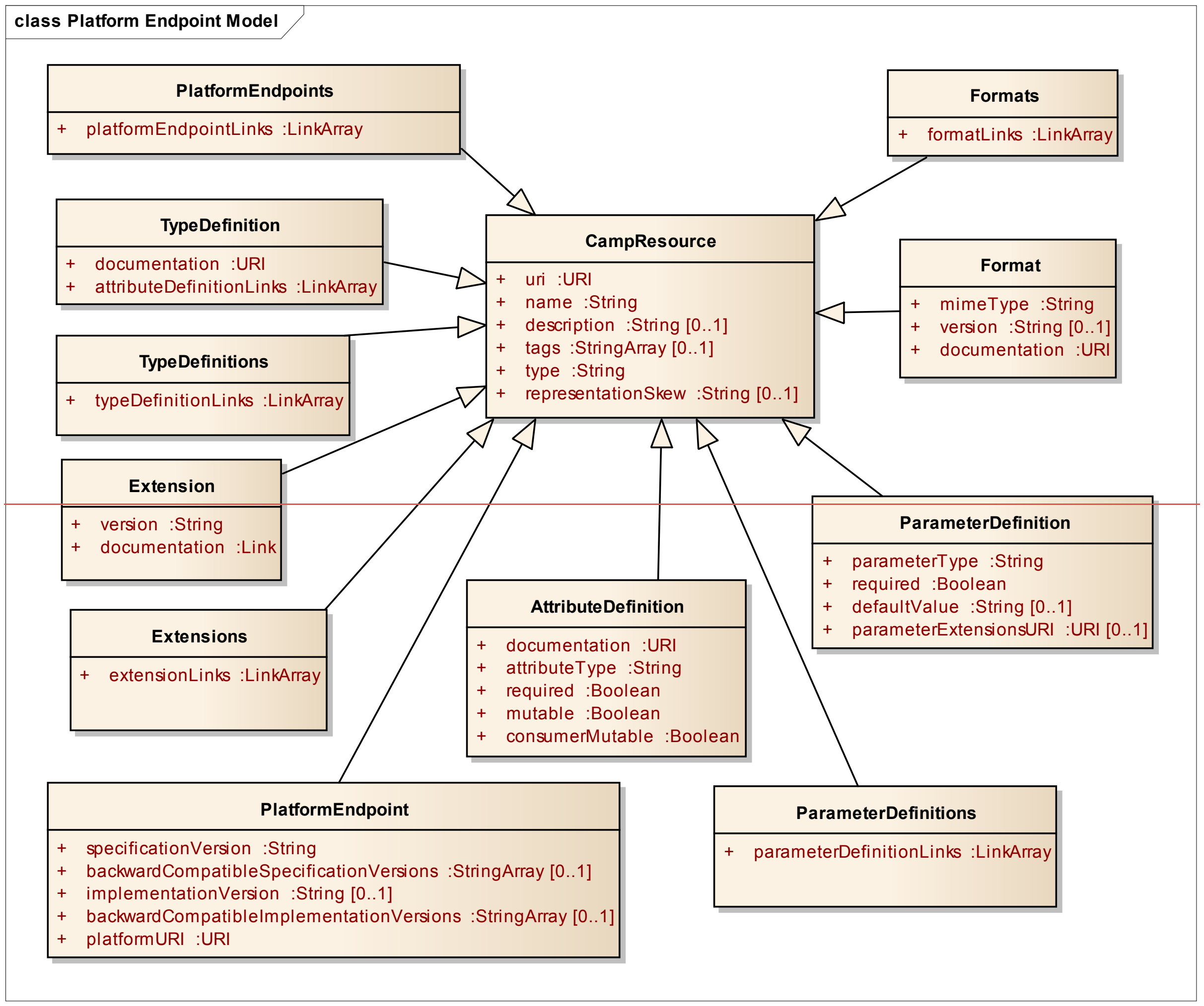


Figure 2-7: ~~Creation of an Assembly~~Instantiating an application from a plan resource

In Figure 2-7 the POST(1) request creates ~~an Assembly Template~~a *plan resource* by uploading either a PDP or a ~~DP~~Plan file to the ~~Platform resource's URI~~*plans resource*. The POST(2) request ~~on~~to the ~~Assembly Template's URI~~*assemblies resource* creates an ~~Assembly~~*assembly resource*. Multiple ~~Assembly~~*assembly resources* can be created from ~~an Assembly Template~~a single *plan resource* by submitting multiple HTTP POST requests.

~~3.10~~2.4 Versions and Extensions

This specification supports multiple endpoints and versions, and extensions. All of these are represented in the resource model so they can be discovered by CAMP Consumers. The resources enabling discovery are shown in Figure 2-8, and their relationships are shown in Figure 2-9.


class Platform Endpoint Model
PlatformEndpoints
+ platformEndpointLinks :LinkArray
Formats
+ formatLinks :LinkArray
TypeDefinition
+ documentation :URI
+ attributeDefinitionLinks :LinkArray
CampResource
+ uri :URI
+ name :String
+ description :String [0..1]
+ tags :StringArray [0..1]
+ type :String
+ representationSkew :String [0..1]
Format
+ mimeType :String
+ version :String [0..1]
+ documentation :URI
TypeDefinitions
+ typeDefinitionLinks :LinkArray
Extension
+ version :String
+ documentation :Link
ParameterDefinition
+ parameterType :String
+ required :Boolean
+ defaultValue :String [0..1]
+ parameterExtensionsURI :URI [0..1]
Extensions
+ extensionLinks :LinkArray
AttributeDefinition
+ documentation :URI
+ attributeType :String
+ required :Boolean
+ mutable :Boolean
+ consumerMutable :Boolean
PlatformEndpoint
+ specificationVersion :String
+ backwardCompatibleSpecificationVersions :StringArray [0..1]
+ implementationVersion :String [0..1]
+ backwardCompatibleImplementationVersions :StringArray [0..1]
+ platformURI :URI
ParameterDefinitions
+ parameterDefinitionLinks :LinkArray

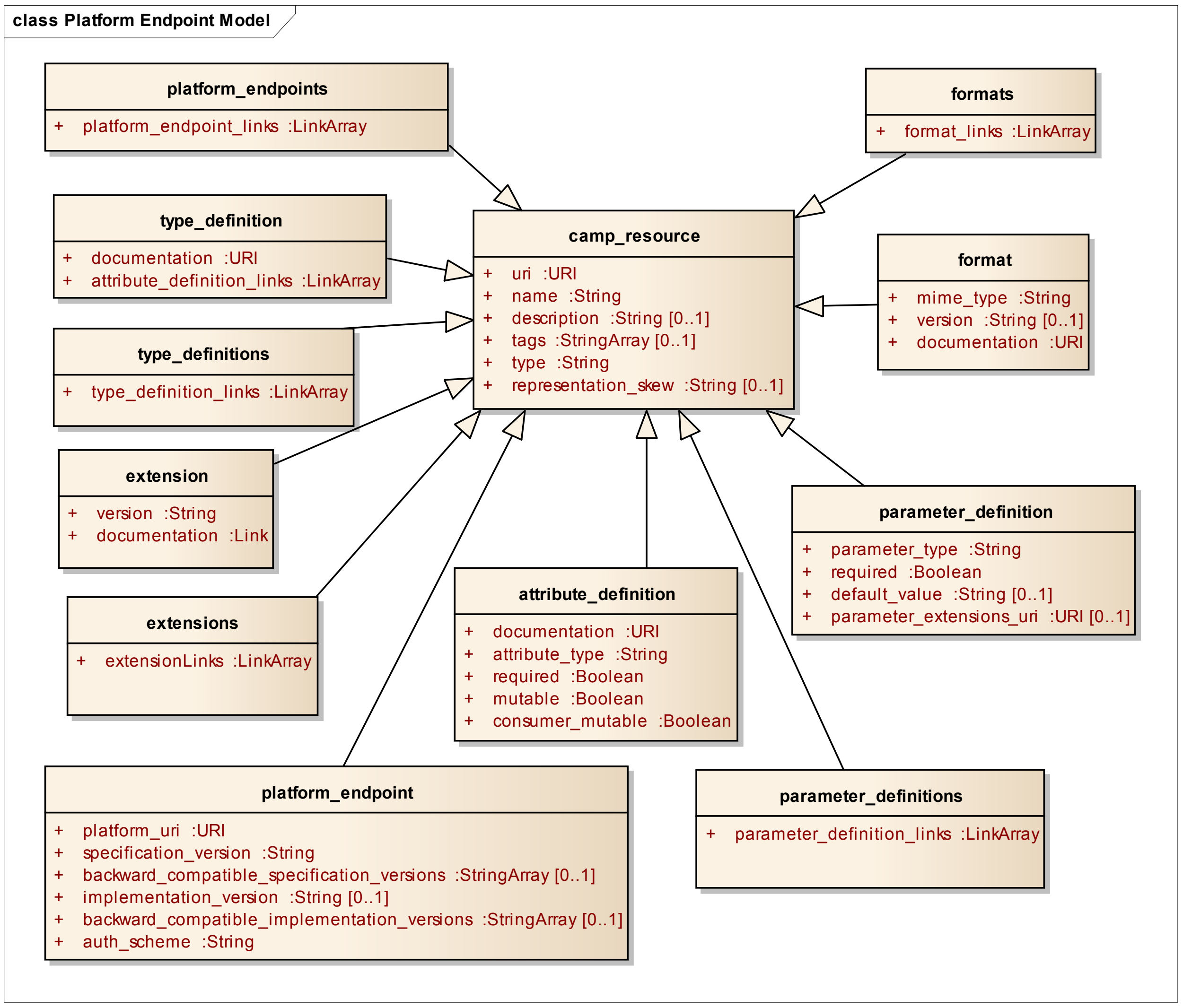
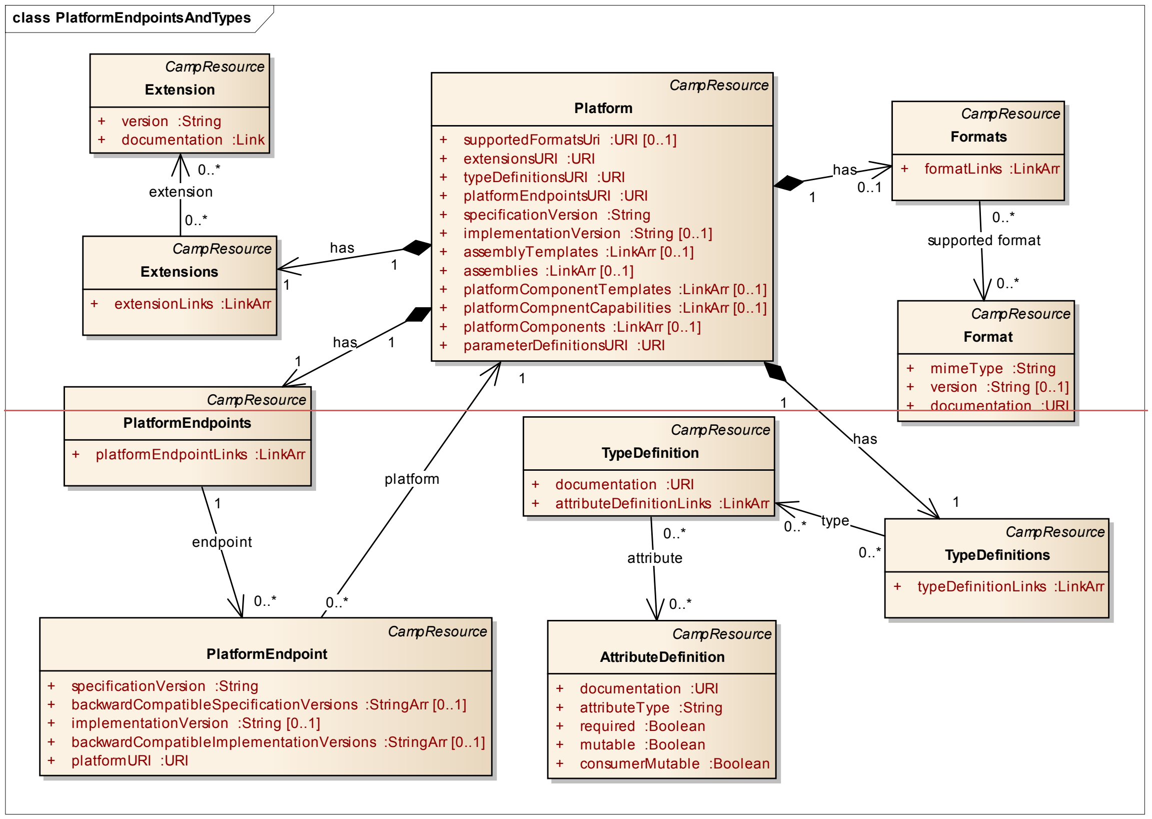

class Platform Endpoint Model
platform_endpoints
+ platform_endpoint_links :LinkArray
formats
+ format_links :LinkArray
type_definition
+ documentation :URI
+ attribute_definition_links :LinkArray
camp_resource
+ uri :URI
+ name :String
+ description :String [0..1]
+ tags :StringArray [0..1]
+ type :String
+ representation_skew :String [0..1]
format
+ mime_type :String
+ version :String [0..1]
+ documentation :URI
type_definitions
+ type_definition_links :LinkArray
extension
+ version :String
+ documentation :Link
parameter_definition
+ parameter_type :String
+ required :Boolean
+ default_value :String [0..1]
+ parameter_extensions_uri :URI [0..1]
extensions
+ extensionLinks :LinkArray
attribute_definition
+ documentation :URI
+ attribute_type :String
+ required :Boolean
+ mutable :Boolean
+ consumer_mutable :Boolean
platform_endpoint
+ platform_uri :URI
+ specification_version :String
+ backward_compatible_specification_versions :StringArray [0..1]
+ implementation_version :String [0..1]
+ backward_compatible_implementation_versions :StringArray [0..1]
+ auth_scheme :String
parameter_definitions
+ parameter_definition_links :LinkArray

Figure 2-8: Platform Endpoint and Meta Data Resources


class PlatformEndpointsAndTypes
CampResource
Extension
+ version :String
+ documentation :Link
0..*
extension
0..*
CampResource
Extensions
+ extensionLinks :LinkArr
has
1
1
CampResource
Platform
+ supportedFormatsUri :URI [0..1]
+ extensionsURI :URI
+ typeDefinitionsURI :URI
+ platformEndpointsURI :URI
+ specificationVersion :String
+ implementationVersion :String [0..1]
+ assemblyTemplates :LinkArr [0..1]
+ assemblies :LinkArr [0..1]
+ platformComponentTemplates :LinkArr [0..1]
+ platformCompnentCapabilities :LinkArr [0..1]
+ platformComponents :LinkArr [0..1]
+ parameterDefinitionsURI :URI
has
1
0..1
CampResource
Formats
+ formatLinks :LinkArr
0..*
supported format
0..*
CampResource
Format
+ mimeType :String
+ version :String [0..1]
+ documentation :URI
has
1
1
CampResource
PlatformEndpoints
+ platformEndpointLinks :LinkArr
1
endpoint
0..*
platform
1
0..*
CampResource
PlatformEndpoint
+ specificationVersion :String
+ backwardCompatibleSpecificationVersions :StringArr [0..1]
+ implementationVersion :String [0..1]
+ backwardCompatibleImplementationVersions :StringArr [0..1]
+ platformURI :URI
CampResource
TypeDefinition
+ documentation :URI
+ attributeDefinitionLinks :LinkArr
0..*
attribute
0..*
CampResource
AttributeDefinition
+ documentation :URI
+ attributeType :String
+ required :Boolean
+ mutable :Boolean
+ consumerMutable :Boolean
0..*
type
0..*
1
has
1
CampResource
TypeDefinitions
+ typeDefinitionLinks :LinkArr

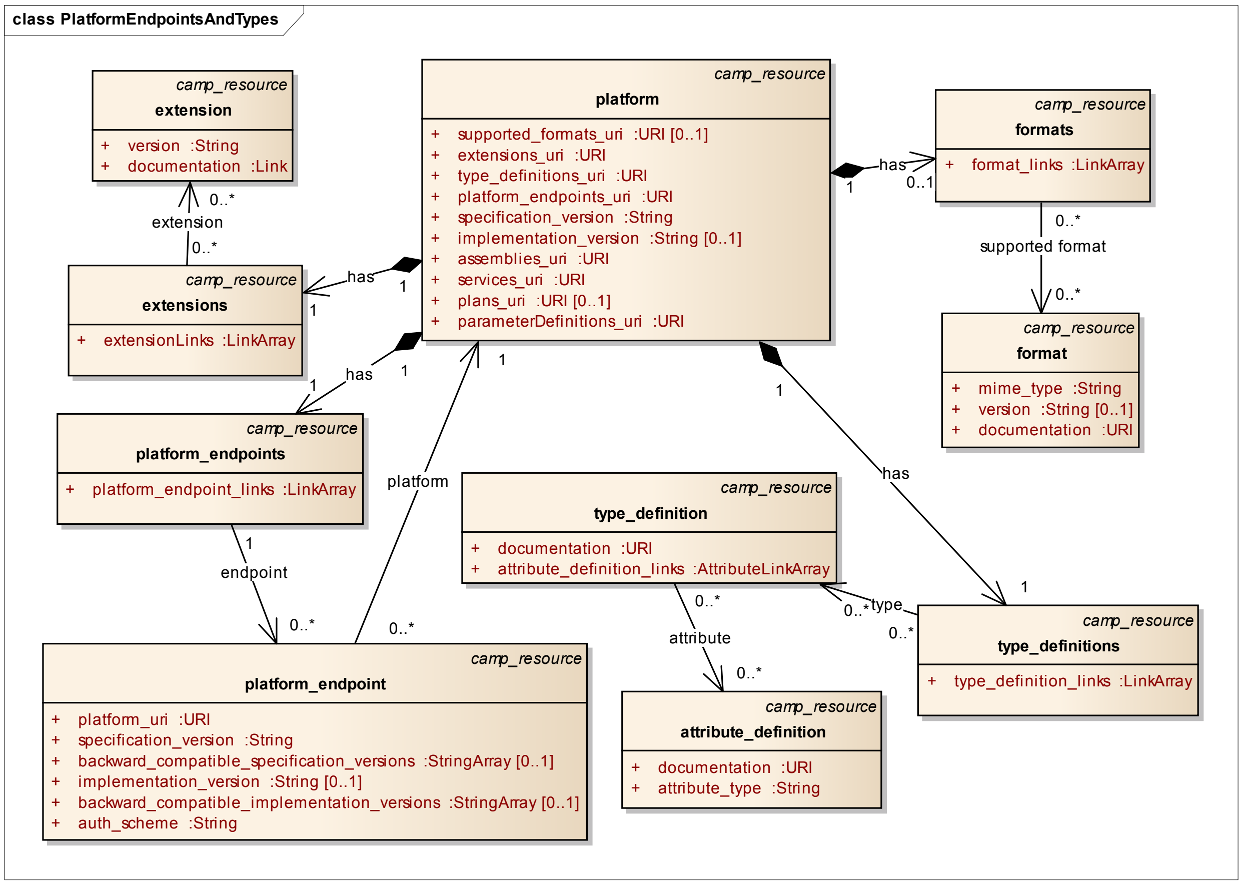
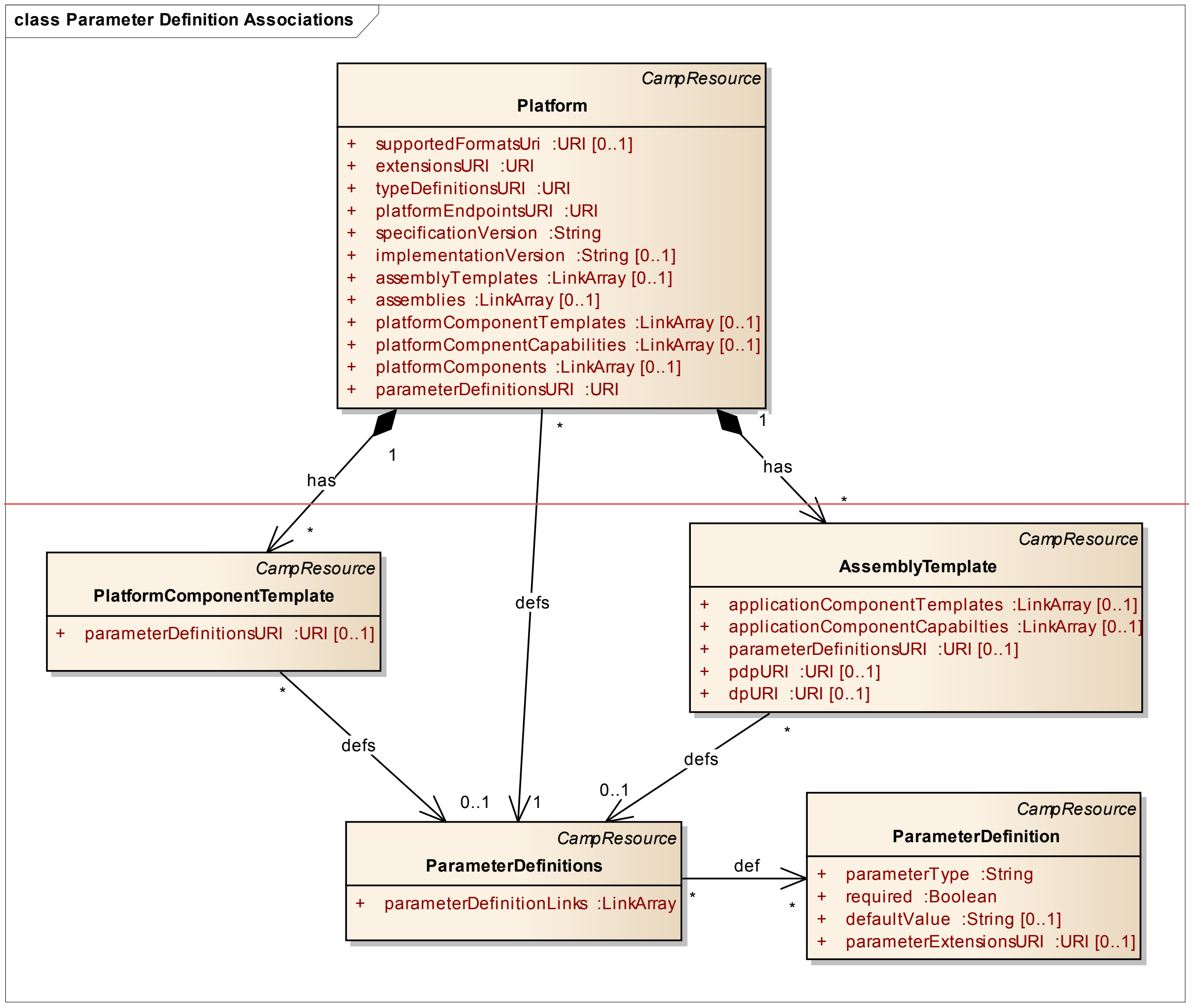


Figure 2-9: Platform Endpoint and Extension Resource Relationships

~~3.11~~2.5 Parameters

Parameters can be defined on the ~~Platform, Assembly Template~~*assemblies resource, services resource*, and ~~Platform Component Template resources.~~, if supported, *plans resource*. Parameters affect the resources that are ~~instantiated~~generated from these resources. Figure 2-10 illustrates the relationships between these resources and the resources used to represent Parameters.


class Parameter Definition Associations
CampResource
Platform
+ supportedFormatsUri :URI [0..1]
+ extensionsURI :URI
+ typeDefinitionsURI :URI
+ platformEndpointsURI :URI
+ specificationVersion :String
+ implementationVersion :String [0..1]
+ assemblyTemplates :LinkArray [0..1]
+ assemblies :LinkArray [0..1]
+ platformComponentTemplates :LinkArray [0..1]
+ platformCompnentCapabilities :LinkArray [0..1]
+ platformComponents :LinkArray [0..1]
+ parameterDefinitionsURI :URI
has
1
*
has
1
*
defs
*
1
CampResource
PlatformComponentTemplate
+ parameterDefinitionsURI :URI [0..1]
CampResource
AssemblyTemplate
+ applicationComponentTemplates :LinkArray [0..1]
+ applicationComponentCapabilties :LinkArray [0..1]
+ parameterDefinitionsURI :URI [0..1]
+ pdpURI :URI [0..1]
+ dpURI :URI [0..1]
defs
*
0..1
defs
*
0..1
CampResource
ParameterDefinitions
+ parameterDefinitionLinks :LinkArray
def
*
*
CampResource
ParameterDefinition
+ parameterType :String
+ required :Boolean
+ defaultValue :String [0..1]
+ parameterExtensionsURI :URI [0..1]

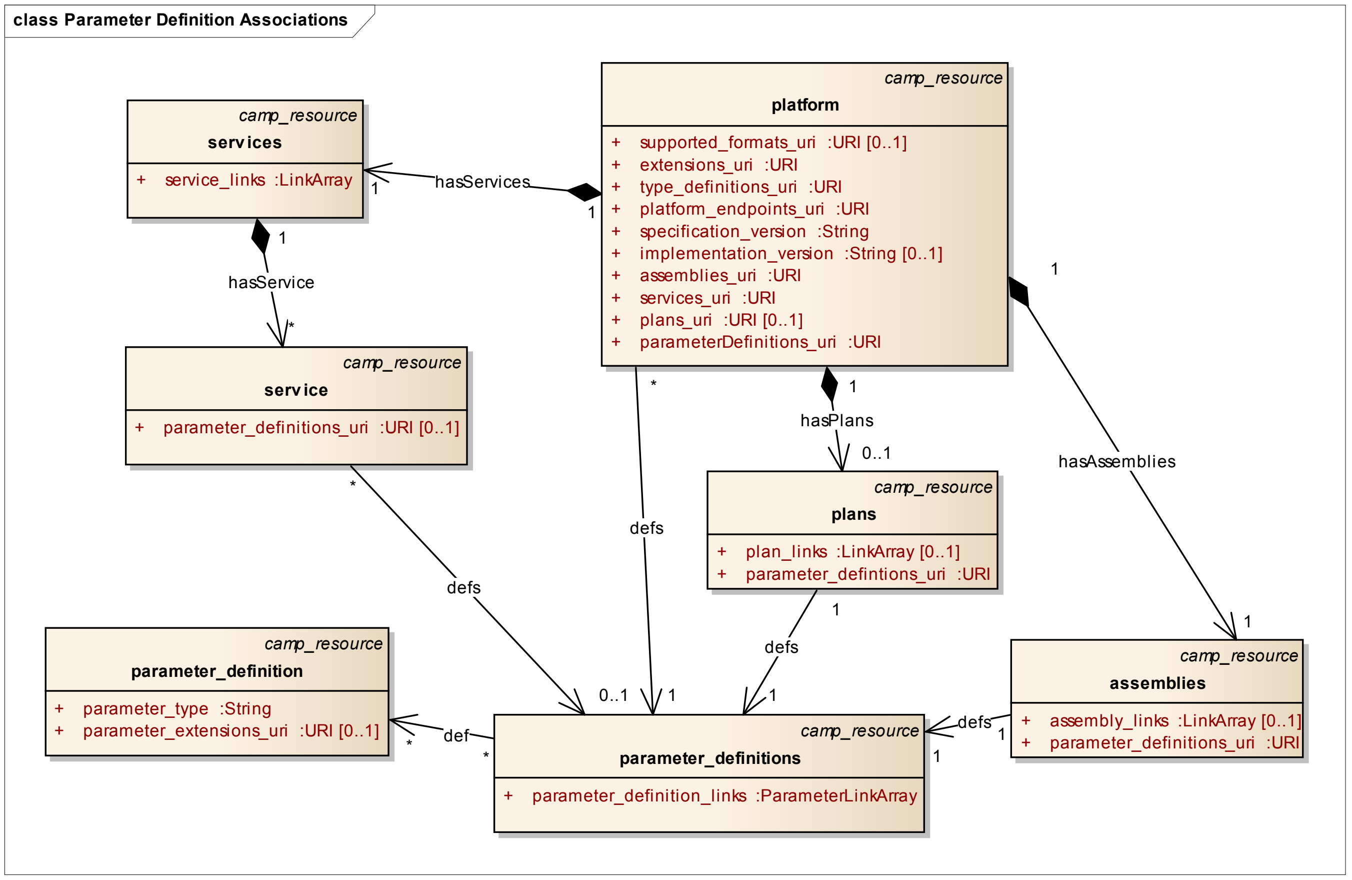
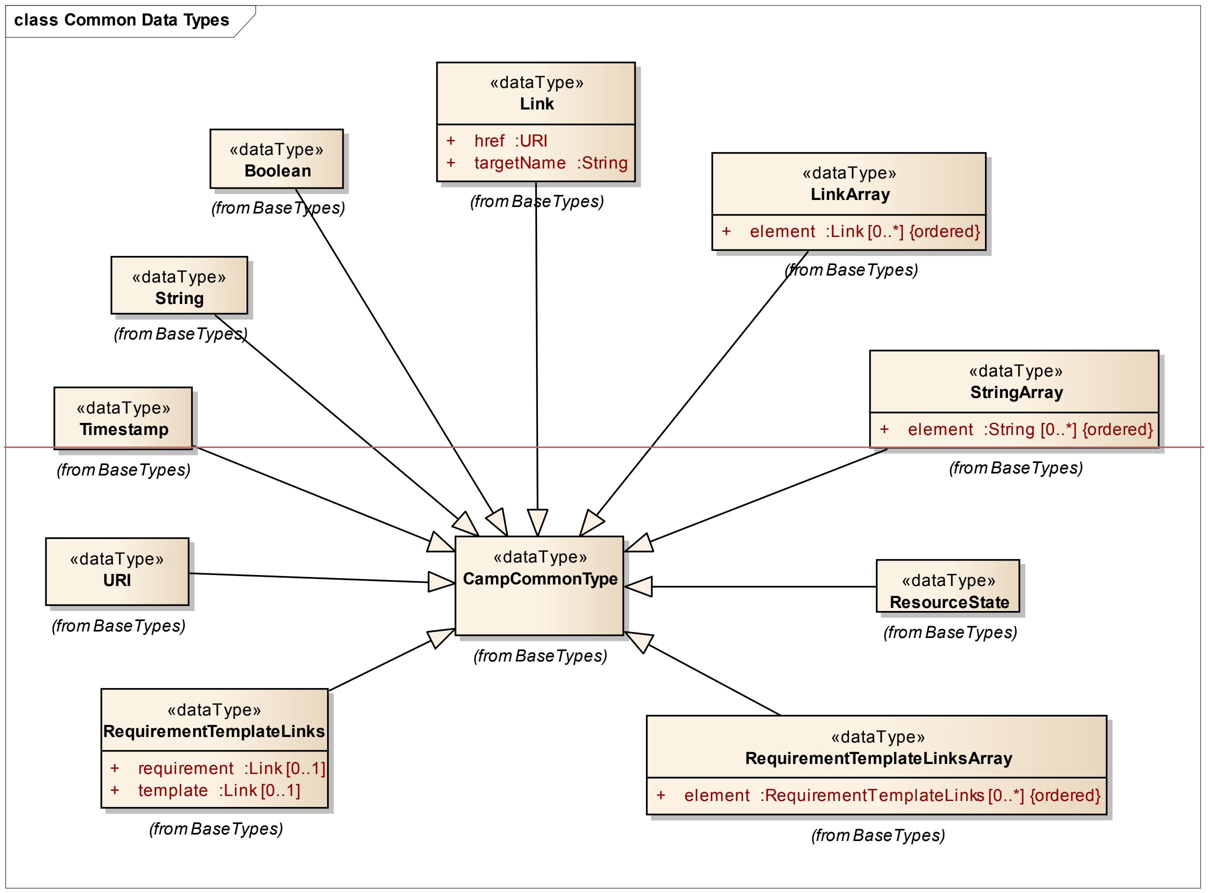

class Parameter Definition Associations
camp_resource
services
+ service_links :LinkArray
camp_resource
platform
+ supported_formats_uri :URI [0..1]
+ extensions_uri :URI
+ type_definitions_uri :URI
+ platform_endpoints_uri :URI
+ specification_version :String
+ implementation_version :String [0..1]
+ assemblies_uri :URI
+ services_uri :URI
+ plans_uri :URI [0..1]
+ parameterDefinitions_uri :URI
hasServices
hasService
camp_resource
service
+ parameter_definitions_uri :URI [0..1]
hasPlans
hasAssemblies
camp_resource
plans
+ plan_links :LinkArray [0..1]
+ parameter_defintions_uri :URI
defs
camp_resource
parameter_definition
+ parameter_type :String
+ parameter_extensions_uri :URI [0..1]
def
camp_resource
parameter_definitions
+ parameter_definition_links :ParameterLinkArray
camp_resource
assemblies
+ assembly_links :LinkArray [0..1]
+ parameter_definitions_uri :URI

Figure 2-10: Parameter Definition Relationships

~~3.12~~2.6 CAMP Common Attribute Types

Many of the attributes in the UML class diagrams have one of the CAMP common attribute types, specified in Section ~~5.1, "Common Types~~5.2, "Attribute Types". Figure 2-11 is a UML diagram showing the common data types as UML Data Types, which are used for the Types of these Resource Attribute definitions.



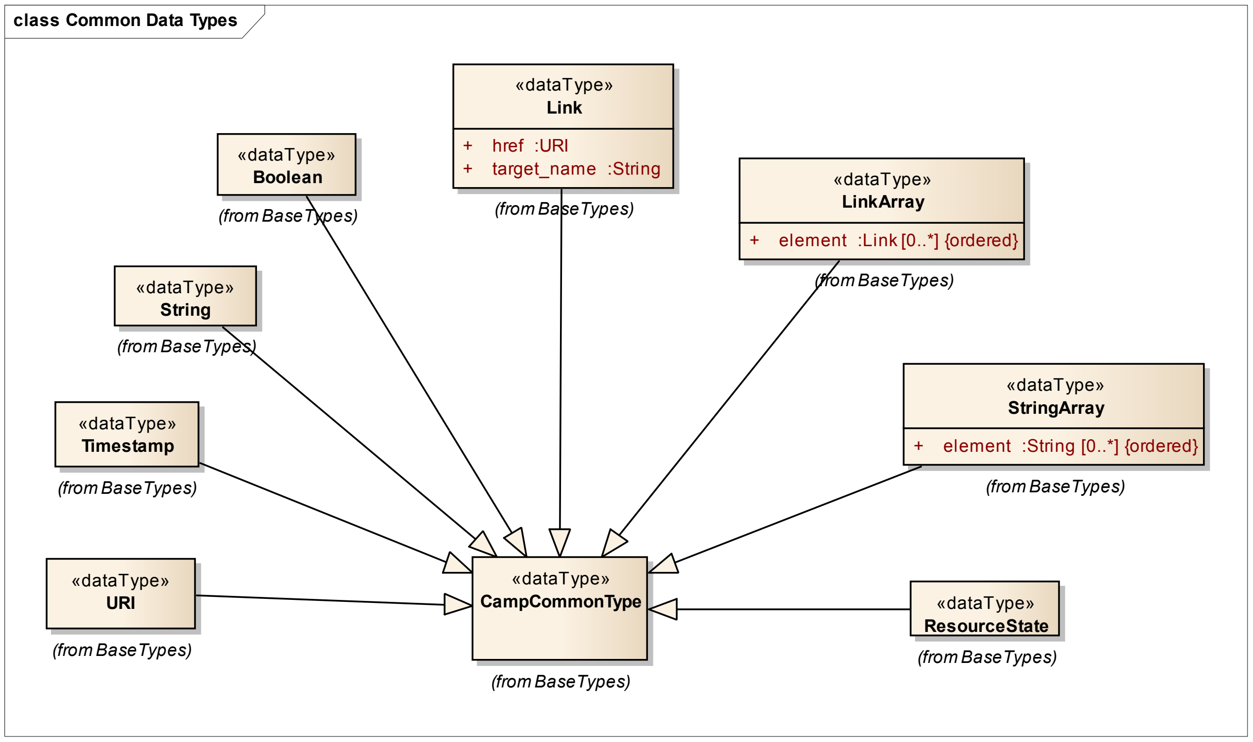


Figure 2-11: CAMP Common Base Types for Resource Attribute Definitions

Multi-valued member attributes are used to model the elements of the LinkArray~~,~~ and StringArray~~, and RequirementTemplateLinksArray types.~~. This is done for modeling purposes only; the attribute name "element" does not appear in the JSON serialization for these common types.

The~~ three~~ array types have their elements tagged as ordered.

~~3.13~~2.7 Representation Skew

There can be situations in which the information in the resources provided by the CAMP API is not a complete or accurate representation of the state of the underlying ~~platform ~~implementation. For example, while ~~instantiating~~generating a new instance of an application, a CAMP server might be asked to provide a representation of ~~an Application~~a Component that corresponds to a dataset that is in the process of being loaded onto a database~~ service~~. While the CAMP server might not be able to provide all of the information about this ~~Application ~~Component, it would be inaccurate to say that~~ Application~~ Component does not exist; it exists but it an intermediate state. It is expected that these sorts of situations will be the exception and that, during the majority of its existence, a CAMP resource will be in synch with the state of its underlying platform implementation.

The significance of this skew is the manner in which it affects the ~~client's~~Consumer's interactions with, and expectations about, the resource. In the above example, while the ~~client~~Consumer cannot ~~reasonably expect to ~~make any changes to the ~~Application ~~Component until it has reached a steady state, the ~~client~~Consumer can expect that the resource will reach this state in the near future. There are other situations in which, through some sort of error~~ or breakdown~~, the CAMP API cannot tell when or if the information in the resource will ever be synchronized with the underlying implementation.

Details on how this skew is exposed in the CAMP API are provided in Section ~~5.3.6~~1.1.1, "~~representationSkew~~representation_skew".

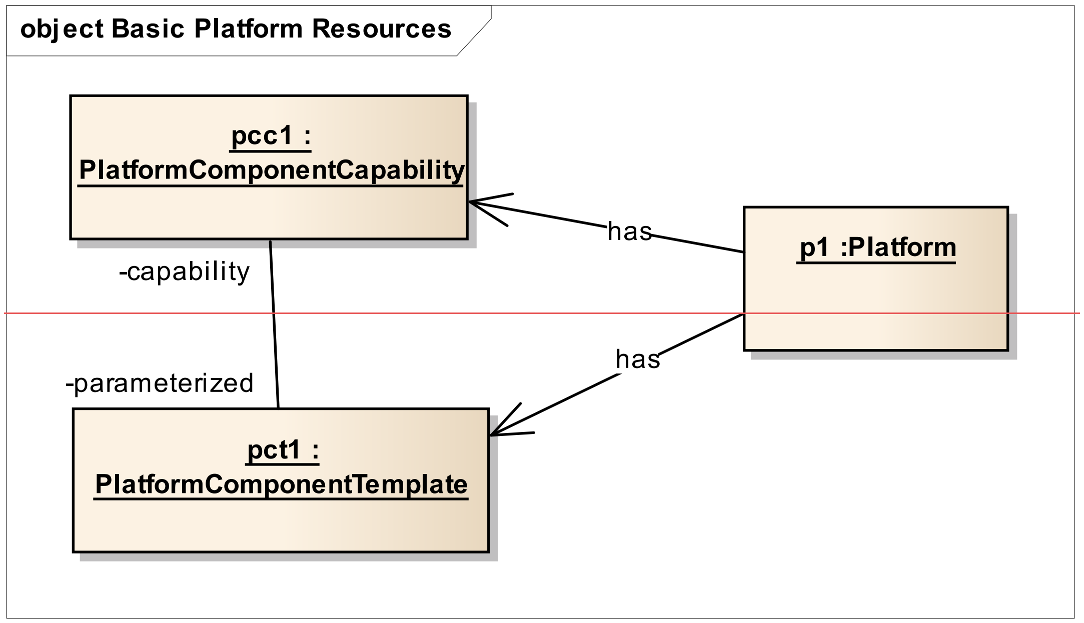
~~4~~3 Application Management Lifecycle

This section is informative. The figures in this section are UML object instance diagrams, which represent related ~~Resource instances~~Resources at various stages of Platform Resource ~~instantiation~~lifecycle. For simplification, attributes for these resources are not shown. For a comprehensive list of attributes for resources see Section 5, “Resources”.

Instances in these diagrams are indicated by boxes, with an underlined “*object-name: Class”* label. Relationships visible thru the API are shown using associations with navigation arrows. Implementation specific relationships are indicated using the association end notation, without navigation arrows.

~~4.1~~3.1 Initial Platform Resources

The CAMP model includes the resources below when no ~~Assemblies have been created~~*assembly resources* or *plan resources* have been created. Note that the support of the *plans resource* and *plan resources* is optional.


object Basic Platform Resources
pcc1 : PlatformComponentCapability
has
p1 :Platform
-capability
has
-parameterized
pct1 : PlatformComponentTemplate

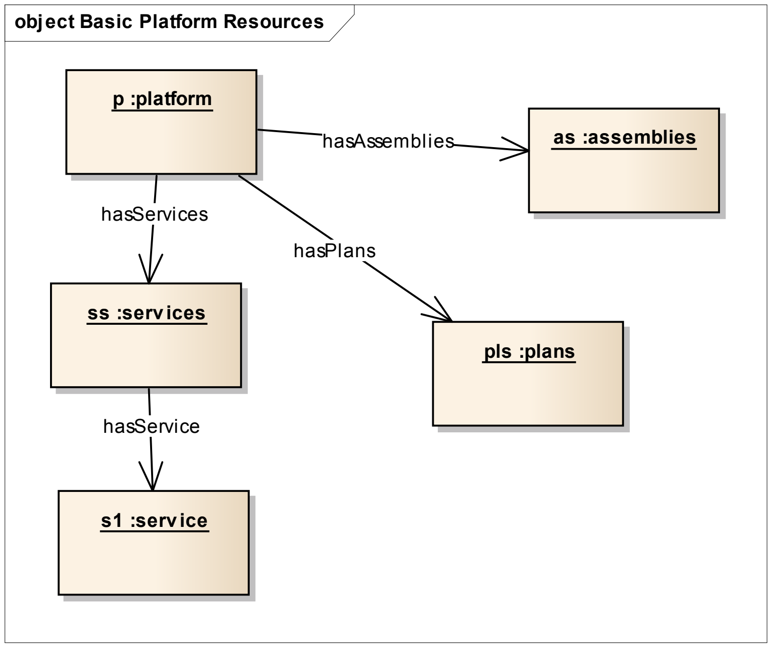


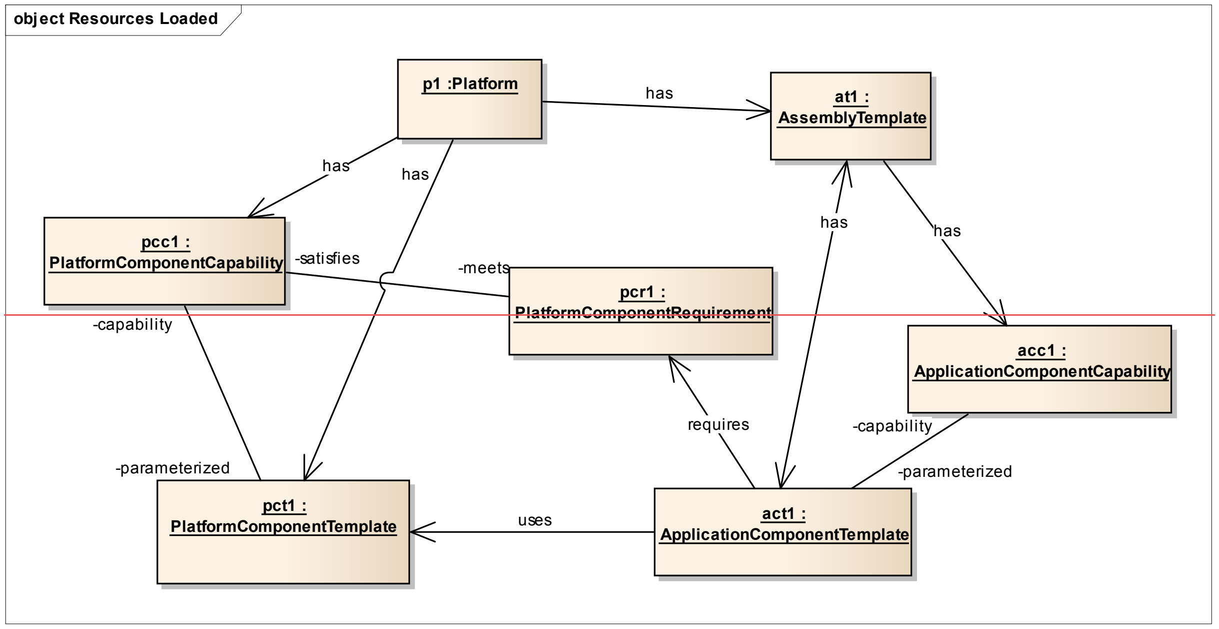
Figure 3-1: Initial Platform Resources

When the Application Administrator first accesses a new account a Platform will have a number of resources visible through the API. The ~~Platform~~*platform* *resource* is used to find the other resources in this diagram. The various ~~Platform Component Capabilities~~*service resources* allow for discovery of all the platform services that are available along with value ranges for each service's attributes.~~ The various Platform Component Templates may represent pools of previously configured platform resources and represent this configuration as specific values for each of the attributes. These values are chosen from the range of values in the corresponding Platform Component Capability.~~

~~4.2~~3.2 Creating an Assembly ~~Template~~ from a PDP or Plan File

A CAMP Consumer can create a new ~~Assembly Template~~ *assembly resource* by uploading either a PDP or a ~~DP~~Plan file to the ~~Platform~~*assemblies* *resource* URI using an HTTP POST request (see Section 0, "Deployment"). ~~When the Assembly Template is created other resources such as Application Component~~

~~Templates or Platform Component Requirements might also be created by the CAMP Provider to facilitate the future creation of an Assembly resource.~~ The loaded assembly model might then appear as follows~~:~~ (for simplification, the instantiated component resources are not shown in Figure 3-2):


object Resources Loaded
p1 :Platform
at1 : AssemblyTemplate
has
has
has
pcc1 : PlatformComponentCapability
-satisfies
-meets
pcr1 : PlatformComponentRequirement
has
has
acc1 : ApplicationComponentCapability
-capability
requires
-capability
-parameterized
-parameterized
pct1 : PlatformComponentTemplate
uses
act1 : ApplicationComponentTemplate

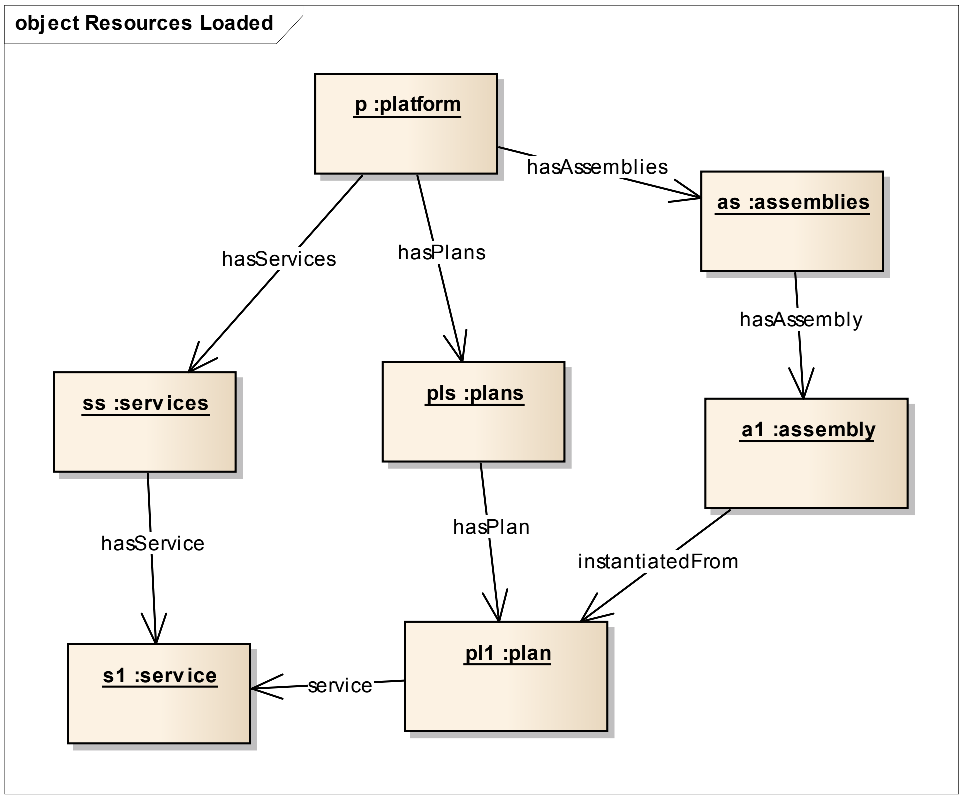


Figure 3-2: Loaded Assembly Resource~~Resources~~

~~An Assembly Template is now present with one or more Application Component Templates, Application Component Requirements and Platform Component Requirements. The Application Component Template resources represent code and/or associated resources that were loaded with the PDP. The Application Component Requirement resources were used to find Application Component Templates (pre-existing software libraries for example) that were not part of the PDP but are required for the~~

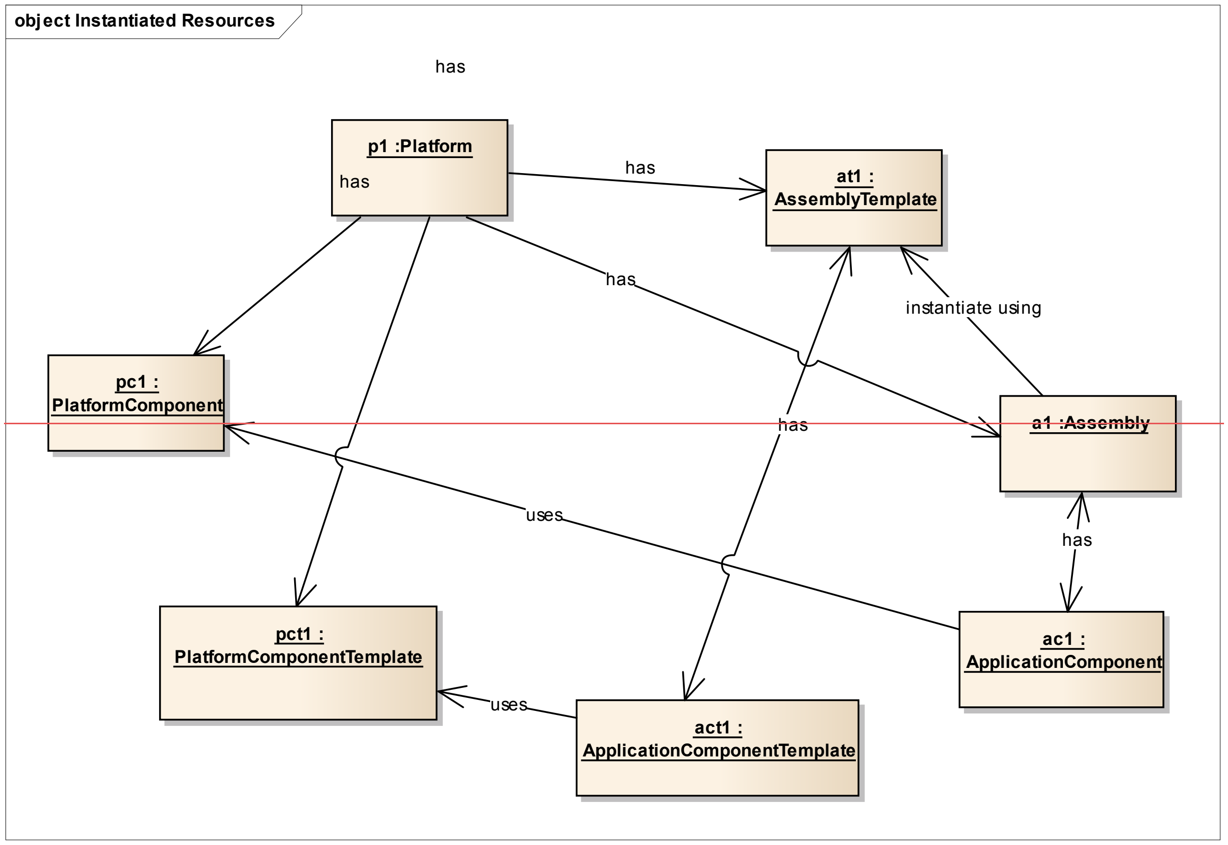
~~application. The Application Component Capability resources show the range of configurable attributes that the loaded Application Components can take on, when reused by future Application Components.~~

~~The Platform Component Requirement resources were used to find Platform Component Templates (pre-configured platform resources) that were part of the platform and required by the loaded Application Components, but perhaps unknown at the time the PDP was created (in an ADE for example).~~

If any of its requirements are not resolved, ~~an Assembly Template~~a PDP or Plan file could require modification before it can be used to create ~~Assembly~~*assembly* *resources*.

~~4.3~~ Creating ~~and Managing~~ an ~~Application~~ Assembly

3.3 ~~To start an application an Assembly is created~~ from ~~the Assembly Template using a POST request (see Section 2.2, "Deployment"). This also creates Application Components and Platform Components corresponding to their respective templates, and the model would look as follows:~~a plan resource


object Instantiated Resources
has
p1 :Platform
has
has
at1 : AssemblyTemplate
has
instantiate using
pc1 : PlatformComponent
has
a1 :Assembly
uses
has
pct1 : PlatformComponentTemplate
ac1 : ApplicationComponent
uses
act1 : ApplicationComponentTemplate

If a Provider supports the *plans resource*, a CAMP Consumer can create a new *plan resource* without creating an *assembly resource* by supplying the contents of, or a reference to, either a PDP or a Plan file to the *plans resource* URI in an HTTP POST request (see Section 6.12, "Registering a Plan"). The loaded *plan resource* model might then appear as follows:

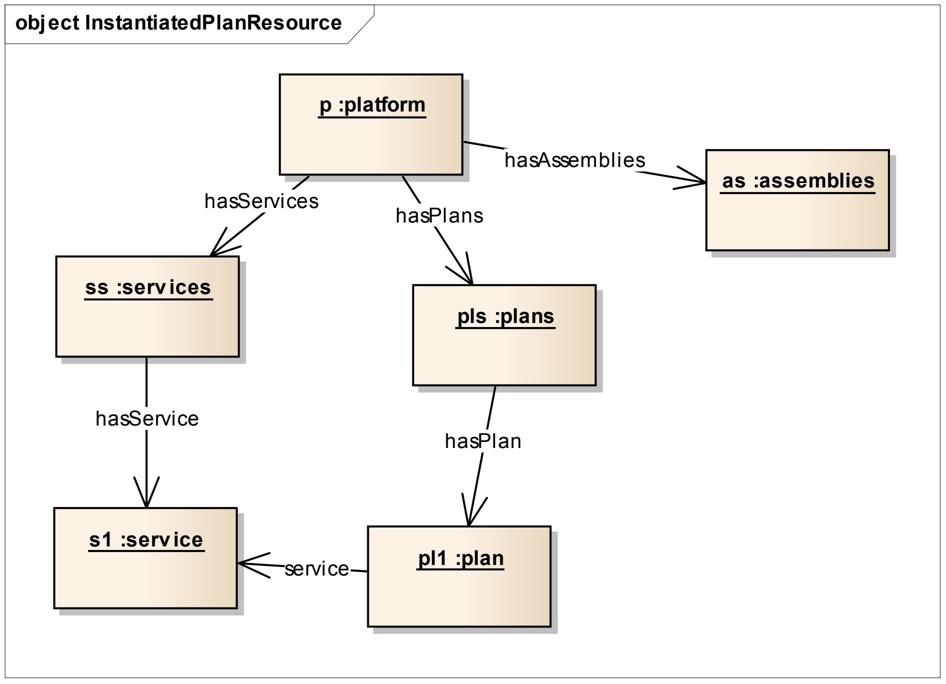


Figure 3-3: Loaded Plan Resource

A CAMP Consumer can create a new *assembly resource* from an existing *plan resource* by providing the reference to that *plan resource* to the *assemblies resource* URI in a HTTP POST request (see Section 6.11.1, "Deploying an Application by Reference").

Using this two-step process, the loaded *assembly resource* model would appear the same as when using the one-step process, as shown in Figure 3-2.

3.4 Managing an Application Assembly

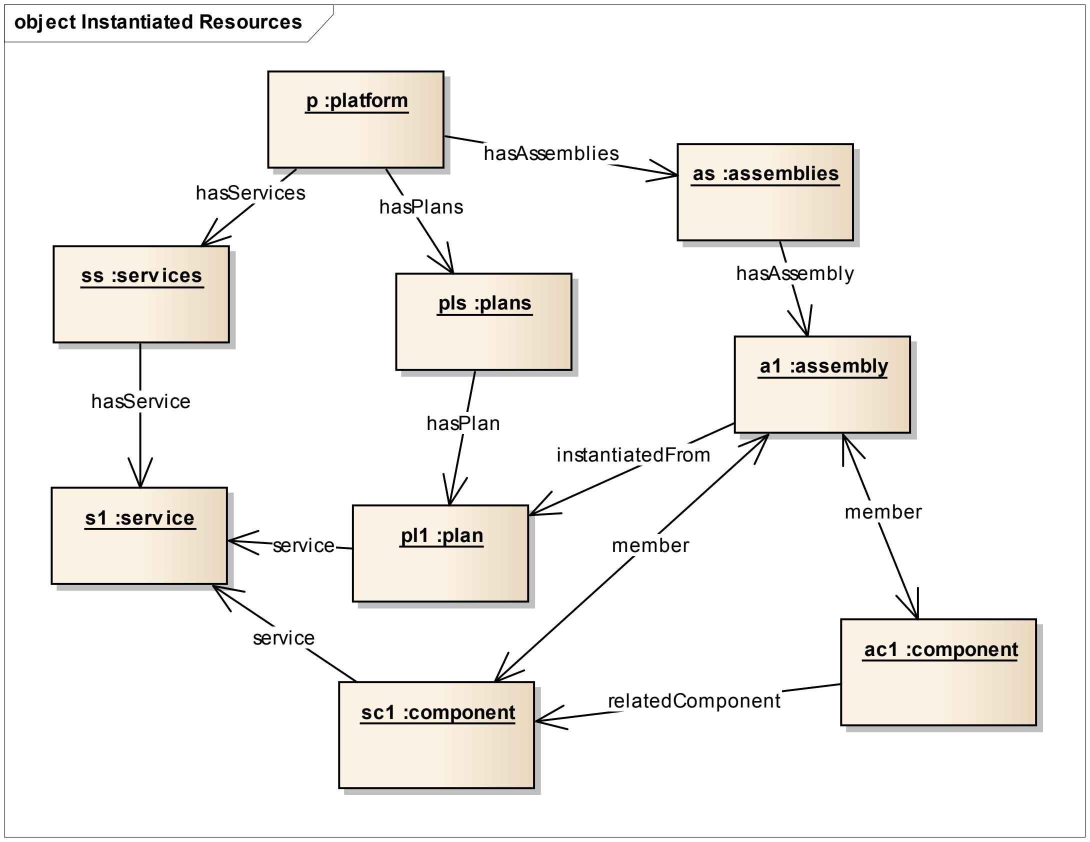


Figure 3~~-~~4: Instantiated Resources

To manage the operation of the application, the Application Administrator interacts with the ~~Assembly~~*assembly resource* and the related ~~Application Component and Platform Component~~*component resources*.

The traversal of the resources in the model can be accomplished by following the navigation arrows on the associations in these object instance diagrams, from each resource to the other resources it depends on.

The Application Administrator can observe real-time operational metrics through ~~Sensor~~*sensor resources* on ~~Assembly~~*assembly resources* and ~~Component~~*component resources*. In response to these metrics, the Application Administrator — or an automated process such as a management system — can affect changes to those resources through the ~~Operation~~*operation resources* linked from those same resources.

~~4.4~~3.5 Removing Assemblies ~~and Assembly Templates~~

When finished working with an application, an Application Administrator can delete an ~~Assembly~~ *assembly resource* using a DELETE request. The CAMP platform will typically soon thereafter remove the ~~Assembly~~*assembly resource* and all associated resources which are dedicated to that ~~Assembly, such as Application Components~~*assembly*. Where such a resource is not removed immediately, for example, when it is in the process of shutting down, it ought to present a representation skew of DESTROYING in the interim.

When the original ~~Assembly Template~~*plan resource* is no longer needed, an Application Administrator can~~, again,~~ delete it using a DELETE request. Again, the CAMP platform will typically delete the ~~Assembly Template~~*plan resource* and all associated resources which are dedicated to that ~~Assembly Template.~~*plan resource.* Where this deletion is accepted but not immediate, such as because an ~~Assembly~~*assembly resource* is in use that references the ~~Assembly Template~~*plan resource*, again the CAMP platform ought to present a representation skew of DESTROYING for the resources being deleted.

~~Following a lifecycle where an Assembly Template is created, then an Assembly is created, then the Assembly is deleted, then the Assembly Template is deleted, the model of Resources in the CAMP Provider will typically look similar to the initial model shown in Figure 3-1.~~

~~5~~4 Platform Deployment Package

The Platform Deployment Package (PDP) ensures portability across platforms. It can be created by a platform to export to another platform, which then imports it. It can also be created by an Application Development Environment running locally or deployed as Software as a Service in the cloud. The PDP (and the ~~Deployment~~ Plan ~~(DP),~~file, see Section ~~4.1.1, "Deployment Plan Overview~~4.2, "Plan Overview") defines the formats for ~~onboarding~~on-boarding new applications ~~into~~onto a CAMP-enabled Provider. ~~A Consumer can create a new Assembly Template resource in the Platform by submitting a valid PDP or DP to a Platform URI using a POST request (see Section 6.11, "Registering a PDP").~~

~~5.1~~4.1 PDP Package Structure

A PDP is an archive which contains a ~~Deployment~~ Plan ~~(DP)~~ file named `camp.yaml` at the root of the archive ~~(see Section 4.1.1, "Deployment Plan Overview").~~. A PDP archive MAY include other files related to the application including, but not limited to, language-specific bundles, resource files, application content files such as web archives, database schemas, scripts, source code, localization bundles, and icons; and metadata files such as manifests, checksums, signatures, and certificates. **[PDP-01]**

~~5.1.1~~4.1.1 Supported Archive Formats

A Provider SHALL support the following archive formats for a PDP:

- A PDP as a ZIP archive [ZIP] **[PDP-02]**
- A PDP as a TAR archive [TAR] **[PDP-03]**
- A PDP as a GZIP [~~GZIP~~RFC1952] compressed TAR archive **[PDP-04]**

Providers MAY support additional archive formats for the PDP. **[PDP-05]**

~~5.1.2~~4.1.2 Validating Integrity

A PDP MAY contain a manifest file, named camp.mf, at the root of the archive. **[PDP-06]** This file contains SHA256 [SHA256] digests of some or all files in the package. A Provider SHOULD reject a PDP if any digest listed in the manifest does not match the computed digest for that file in the package. **[PDP-07]**

A PDP MAY contain a certificate, named camp.cert, at the root of the archive. **[PDP-08]** This file contains a signed SHA256 digest for the manifest file and the corresponding X.509 certificate. A Provider SHOULD reject any PDP for which the signature verification fails. **[PDP-09]**

The format of the manifest file and the certificate file SHALL be as defined by the OVF specification [OVF]. **[PDP-10]**

~~5.2~~4.2 ~~Deployment~~ Plan Overview

The ~~Deployment~~ Plan ~~(DP)~~ provides a description of the artifacts that make up an application, the services that are required to execute or utilize those artifacts, and the relationship of the artifacts to those services. As discussed previously, Plans can be represented in two ways, either as YAML files or as CAMP resources. The examples in this section show Plans as YAML files.

Example 1: Minimal ~~DP~~Plan describing an application consisting of a single RPM file.

```
~~campVersion~~camp_version: CAMP 1.1
artifacts:
  -
    ~~artifactType~~artifact_type: org.rpm:RPM
    content: { href: my-app.rpm }
```

The above example describes an application that consists of a single RPM (RPM Package Manager) package, named “my-app.rpm”, which exists at the root of the PDP archive. The services required to install/run this package and the relationship of this package to those services are not specified.

~~5.2.1~~4.2.1 Types

~~Deployment~~ Plans can contain descriptions of artifacts, services and their relationships. However, it is outside the scope of this specification to provide detailed definitions of these entities. Instead ~~Deployment~~ Plans use ‘type’ nodes to identify these things. ‘Type’ nodes are Strings that describe entities that are managed by CAMP, but whose value and semantics are defined outside the CAMP specification. For example, a group of PaaS providers could agree to use the artifact type “org.rpm:RPM” to identify RPM packages. Line 03 in Example 1, above, is an example of the use of such a type.

To promote portability, both providers and consumers of the CAMP API are encouraged to namespace-qualify the types that they use. For example, if a PaaS provider supports a requirement type that expresses the relationship “deploy on a Spring container”, the value “com.paas-r-us.spring.DeployOn” is preferable to the value “DeployOn”, as the latter is likely to collide with similar types.

In addition to defining the labels for artifacts, services, and their relationships it is expected that those individuals and organizations that define such labels will also define additional attributes that qualify and constrain the entity that is referenced.

Example 2: Expanded ~~DP~~Plan describing details of how to install the RPM

```
00 ~~campVersion~~camp_version: CAMP 1.1
01 artifacts:
02   -
03     ~~artifactType~~artifact_type: org.rpm:RPM
04     content: { my-app.rpm }
05     requirements:
06       -
07         ~~requirementType~~requirement_type: org.rpm:Install
08         org.rpm.installopts.excludedocs: true
```

The above example is an expansion of Example 1 that uses a Requirement Specification (lines 07-08) to describe the relationship of the RPM artifact to some (unspecified) service. This relationship is labeled on line 07 with a value of “org.rpm:Install”. It is assumed that ~~this~~the semantics associated with this label are documented, and that this documentation also describes the value space and semantics of the “org.rpm.installopts.excludedocs” node used on line 08.

~~5.2.2~~4.2.2 Service Specifications

In addition to artifacts and their relationships to the platform, ~~Deployment~~ Plans can also contain descriptions of the services that ~~can be~~are used by those artifacts. These descriptions take the form of Service Specifications.

Example 3: Expanded DP that includes a Service Specification

```
00 ~~campVersion~~camp_version: CAMP 1.1
01 artifacts:
02   -
03     ~~artifactType~~artifact_type: org.rpm:RPM
04     content: { my-app.rpm }
05     requirements:
06       -
07         ~~requirementType~~requirement_type: org.rpm:Install
08         org.rpm.installopts.excludedocs: true
09         fulfillment:
10           characteristics:
11             -
12               ~~characteristicType~~characteristic_type: com.example:Linux
13               com.example.linux.kernelVersion: 3.9.6
14               org.iaas.bitsize: 64
```

The above example is an expansion of Example 2 that uses a Service Specification to outline the parameters of the service on which the RPM is to be installed. Lines 10-14 in the above example make up the Service Specification which, in this case, consists of a single Characteristic Specification (lines 12-14). Line 12 indicates that the target service for the RPM is a Linux instance. Line 13 indicates that this target has to run kernel version 3.9.6 and line 14 indicates that it has to be a 64-bit system.

~~5.2.2.1~~4.2.2.1 Shared Services

There are situations in which an application can have two or more artifacts that need to share the same runtime instance of a service.

Example 4: ~~DP~~Plan with shared Service Specification

```
00 ~~campVersion~~camp_version: CAMP 1.1
01 artifacts:
02   -
03     ~~artifactType~~artifact_type: com.java:WAR
04     content: { href: vitaminder.war }
05     requirements:
06       -
07         ~~requirementType~~requirement_type: com.java:HostOn
08         com.java.servlet.contextName: "/vitaM"
09         fulfillment:
10           …
11       -
12         ~~requirementType~~requirement_type: com.java.jdbc:ConnectTo
13         fulfillment:
14           id: db
15           characteristics:
16             -
17               ~~characteristicType~~characteristic_type: org.storage.db:RDBM
18               …
19             -
20               ~~characteristicType~~characteristic_type:
org.storage.db:Replication
21               …
22             -
23               ~~characteristicType~~characteristic_type: org.iso.sql:SQL
24               …
25   -
26     ~~artifactType~~artifact_type: org.sql:SqlScript
27     content: { href: vitaminder.sql }
28     requirements:
29       -
30         ~~requirementType~~requirement_type: org.sql:ExecuteAt
31         fulfillment: id:db
```

The above example describes an application with two components, a WAR file and an SQL script. In the case of this particular application, the SQL script is used to initialize the database that will be used by the WAR file. The two artifacts need to share a common database instance or the application will not work. Lines 14-24 describe the target database service. Line 14 is an 'id' node with the value 'db'. This node is used as the target for the 'fulfillment' node on line 31. The use of the "id:db" reference on line 31 indicates that the database instance that will be used to fulfill the "org.sql:ExecuteAt" relationship on line 30 is the same instance that will be used to fulfill the "com.java.jdbc:ConnectTo" relationship on line 12.

The ~~Deployment~~ Plan in Example 4 can also be expressed in the following manner:

Example 5: DP with ‘services’ section

```
00 ~~campVersion~~camp_version: CAMP 1.1
01 artifacts:
02   -
03     ~~artifactType~~artifact_type: com.java:WAR
04     content: { href: vitaminder.war }
05     requirements:
06       -
07         ~~requirementType~~requirement_type: com.java:HostOn
08         com.java.servlet.contextName: "/vitaM"
09         fulfillment:
10           …
11       -
12         ~~requirementType~~requirement_type: com.java.jdbc:ConnectTo
13         fulfillment: id:db
14   -
15     ~~artifactType~~artifact_type: org.sql:SqlScript
16     content: { href: vitaminder.sql }
17     requirements:
18       -
19         ~~requirementType~~requirement_type: org.sql:ExecuteAt
20         fulfillment: id:db
21 services:
22   -
23     id: db
24     characteristics:
25       -
26         ~~characteristicType~~characteristic_type: org.storage.db:RDBM
27         …
28       -
29         ~~characteristicType~~characteristic_type: org.storage.db:Replication
30         …
31       -
32         ~~characteristicType~~characteristic_type: org.iso.sql:SQL
33         …
```

This example is equivalent to Example 4, but places the specification of the shared database service in the ‘services’ section that begins on line 21. In this case, both the WAR file and SQL script artifacts use references (lines 13 and 20, respectively) to indicate the service that will be used to fulfill their particular relationships.

~~5.2.2.2~~4.2.2.2 Service Frameworks

There are situations in which the artifacts of an application are dynamically added (e.g. via a git [Git] push operation) after the creation of a “service framework” on which these artifacts can be deployed. Such a framework can be specified via a ~~Deployment~~ Plan that contains Service Specifications but no Artifact Specifications.

Example 6: ~~DP~~Plan with only Services Specifications

```
~~campVersion~~camp_version: CAMP 1.1
services:
  -
    name: Rails Runtime
    characteristics:
      -
        ~~characteristicType~~characteristic_type: org.ruby-lang:Ruby
        …
      -
        ~~characteristicType~~characteristic_type: org.rubyonrails:Rails
        …
  -
    name: Database
    characteristics:
      -
        ~~characteristicType~~characteristic_type: org.storage.db:RDBM
        …
  -
    name: Git Repo
    characteristics:
      -
        ~~characteristicType~~characteristic_type: com.git-scm:GIT
        …
```

The above example specifies a set of services onto which the user can deploy Rails components by pushing them to the git repository that will be created as a result of ~~registering this DP and instantiating the resulting Assembly Template~~deploying this Plan.

~~5.2.3~~4.2.3 Names, Description, and Tags

~~Deployment~~ Plans, artifacts and services can be decorated with names, descriptions, and tags. CAMP ~~platform~~ implementations can use this information when creating the resources that correspond to these entities. For example, the following ~~Deployment~~ Plan file:

Example 7: ~~DP~~Plan with names, descriptions, and tags

```
name: Mike's Drupal Instance
description: Drupal 6.28
tags: [ PHP, Drupal6, mikez ]
~~campVersion~~camp_version: CAMP 1.1
artifacts:
  -
    ~~artifactType~~artifact_type: net.php:Module
    content:
      href: ftp://ftp.drupal.org/files/projects/drupal-6.28.tar.gz
…
```

when successfully registered, could result in the creation of~~, among other resources,~~ the following ~~Assembly Template~~*plan resource*:

```
{
  "type": "~~assemblyTemplate~~plan",
  "uri~~:~~": "http://uswest.paas-r-us.com/camp/~~AssemblyTemplate~~plan/101",
  "name": "Mike's Drupal Instance",
  "description": "Drupal 6.28",
  "tags": [ "PHP", "Drupal6", "mikez" ],
  …
}
```

~~5.3~~4.3 ~~Deployment~~ Plan Schema

A Platform Deployment Package (PDP) SHALL contain a single ~~Deployment~~ Plan file. **[PDP-11]** The ~~Deployment~~ Plan file SHALL be located at the root of the PDP archive. **[~~PDP-12~~PLAN-01]** The ~~Deployment~~ Plan file SHALL be named "camp.yaml~~" and~~". **[PLAN-02]** The Plan file SHALL ~~consist of a well-formed~~ conform to YAML 1.1 [YAML 1.1~~]~~]. **[PLAN-08]** The Plan file ~~that conforms~~SHALL conform to the description provided in this section. ~~**[PDP-13]** Note the description of the structures and information in this section utilizes YAML's nomenclature.~~**[PLAN-09]**

~~5.3.1~~4.3.1 General ~~Attributes~~Nodes

~~The following nodes may appear as elements of the Deployment Plan~~Plans, Artifact Specifications, ~~or~~and Service Specifications~~.~~ can contain the following nodes:

~~5.3.1.1~~4.3.1.1 name

Type: String

Required: false

This node expresses the human-readable name of the Plan or Specification. Providers MAY reflect the value of this attribute in the names of any resources that are created in the processing the ~~Deployment~~ Plan. **[PDP-14]**

~~5.3.1.2~~4.3.1.2 description

Type: String

Required: false

This node expresses the human-readable description of the Plan or Specification. Providers MAY reflect the value of this attribute in the descriptions of the resources that are in the processing the ~~Deployment~~ Plan. **[PDP-15]**

~~5.3.1.3~~4.3.1.3 tags

Type: String[]

Required: false

This node expresses an array of human-readable tags for the Plan or Specification. Providers MAY reflect the values of this attribute in the tags of the resources that are created in the processing of the ~~Deployment~~ Plan. **[PDP-16]**

~~5.3.2 DeploymentPlan~~

4.3.2 ~~This type~~Plan

A Plan defines the structure ~~of~~ the ~~Deployment~~elements in a Plan~~.~~ file or resource. A ~~Deployment~~ Plan file SHALL contain a single instance of a ~~DeploymentPlan node.~~ ~~**[PDP-17]** This node~~Plan. **[PLAN-03]** A Plan has the following, general representation:

```
name: String ?
description: String ?
tags: ~~?~~
~~  - String +~~
~~campVersion~~[] ?
camp_version: String
origin: String ?
artifacts: ~~?~~
~~  -~~
~~    ArtifactSpecification +~~[] ?
services: ~~?~~
~~  -~~
~~    ServiceSpecification +~~[] ?
```

In addition to the general ~~attributes, the DeploymentPlan node~~nodes, a Plan contains the following ~~attributes~~nodes:

~~5.3.2.1 campVersion~~

4.3.2.1 camp_version

Type: String

Required: true

The value of this node expresses the version of the CAMP specification to which the ~~Deployment Plan conforms. This value SHALL be the Specification Version String of the CAMP specification to which this Deployment Plan conforms. [PDP-18~~Plan conforms. For Plans that conform to this document, the value of this node SHALL be as defined in Section 1.8 "Specification Version". **[PLAN-05]**

~~For Deployment Plans that conform to this document, the value of this node SHALL be "CAMP 1.1" as defined in Section **Error! Reference source not found.** "Specification Version". **[PDP-19]**~~

~~5.3.2.2~~4.3.2.2 origin

Type: String

Required: false

The value of this node specifies the origin of the ~~Deployment~~ Plan. For example, when exporting ~~an Assembly Template~~*a plan resource* into a PDP, a ~~platform instance~~Provider might use the URL of its ~~Platform~~*platform resource* for this value. Alternatively, an Application Development Environment could use its name and version.

~~5.3.2.3~~4.3.2.3 artifacts

Type: ArtifactSpecification[]

Required: false

This node ~~describes~~lists the artifacts that~~, together with~~ comprise the ~~Deployment Plan itself, make up~~application described by the ~~PDP.~~Plan. For portability reasons, Providers ~~SHALL NOT regard~~are cautioned against regarding the order of the ~~ArtifactSpecifications within~~elements in this array as ~~semantically~~ significant. ~~**[PDP-20]**~~

~~5.3.2.4~~4.3.2.4 services

Type: ServiceSpecification[]

Required: false

This node describes the services that the application ~~in~~described by the ~~PDP~~Plan requires in order to function. For portability reasons, Providers ~~SHALL NOT treat~~are cautioned against regarding the order of ~~ServiceSpecifications within~~ the elements in this array as ~~semantically~~ significant. ~~**[PDP-21]**~~

~~5.3.3~~4.3.3 ArtifactSpecification

~~This type~~An ArtifactSpecification describes an artifact of the application. The artifact MAY be contained within the PDP or MAY exist in some other location. **[PDP-22]** ~~This type~~An ArtifactSpecification has the following, general representation:

```
name: String ?
description: String ?
tags: ~~?~~
~~  - ~~String~~ +~~
~~artifactType~~[] ?
artifact_type: String
content: ContentSpecification
requirements: ~~?~~
~~ - ~~
~~    ~~RequirementSpecification~~ +~~[] ?
```

In addition to the general ~~attributes, the~~nodes, an ArtifactSpecification ~~type~~ contains the following ~~attributes~~nodes:

~~5.3.3.1 artifactType~~

4.3.3.1 artifact_type

Type: String

Required: true

~~This~~ The value of an artifact_type node ~~defines~~specifies the type of ~~the~~an artifact~~ described~~.

Note: Values for an artifact_type node are not defined by this ~~Artifact Specification.~~specification. See Section 4.2.1, "Types~~", for a general description of the definition and treatment of these values.~~".

~~5.3.3.2~~4.3.3.2 content

Type: ContentSpecification

Required: true

This node ~~defines~~identifies the location of the content of the artifact described by this Artifact Specification. See Section 4.3.5, "ContentSpecification", for ~~a description of this node's type~~details.

~~5.3.3.3~~4.3.3.3 requirements

Type: RequirementSpecification[]

Required: false

This array specifies the ways in which the artifact described by this Artifact Specification engages with the services provided by the platform. See Section 4.3.6, "RequirementSpecification", for ~~a description of the RequirementSpecification type.~~details. For portability reasons, Providers ~~SHALL NOT treat~~are cautioned against regarding the order of ~~RequirementSpecifications within~~the elements in this array as ~~semantically~~ significant.~~ [PDP-23]~~

~~5.3.4~~4.3.4 ServiceSpecification

A ServiceSpecification ~~outlines~~describes a service ~~that~~used by the application~~ contained within the PDP requires in order to function. This outline is~~.

Note: The description might not ~~intended to~~ be ~~a~~ complete ~~description of the service~~ but~~, instead, delineate those~~ can contain only characteristics ~~that are significant~~of significance to ~~the~~a particular application.~~ This type~~

A ServiceSpecification has the following, general representation:

```
name: String ?
description: String ?
tags: ~~?~~
~~  - ~~String~~ +~~[] ?
id: String ?
href: URI ?
characteristics: ~~?~~
~~  -~~
~~    ~~CharacteristicSpecification~~ +~~[] ?
```

In addition to the general ~~attributes, the~~nodes, a ServiceSpecification~~ type~~ contains the following ~~attributes~~nodes:

~~5.3.4.1~~4.3.4.1 id

Type: String

Required: false

The value of this node serves as an anchor for intra-~~Deployment~~ Plan references. See Section 4.3.6.2, "fulfillment", for information on how this anchor is used. Plans SHALL use id values that are unique within the scope of the Plan. **[PLAN-06]**

4.3.4.2 href

Type: URI

Required: false

The value of this node is a reference to a *service resource* (see Section 5.13, "service Resource") that resolves the service described by this ServiceSpecification. If a Consumer includes this node in a Plan, the value of this node SHALL reference a Consumer-visible resource within the target Platform. **[RMR-01]**

~~5.3.4.2~~4.3.4.3 characteristics

Type: CharacteristicSpecification[]

Required: true

This array provides the characteristics of the service described by this ~~Service Specification.~~ServiceSpecification. See Section 4.3.7, "CharacteristicSpecification", for ~~a description of the CharacteristicSpecification type.~~details. For portability reasons, Providers ~~SHALL NOT treat~~are cautioned against regarding the order of ~~CharacteristicSpecifications within~~the elements in this array as ~~semantically~~ significant. ~~**[PDP-24]**~~

~~5.3.5~~4.3.5 ContentSpecification

~~This type~~A ContentSpecification defines the content of a component. ~~Content Specifications SHALL declare either a String attribute "~~A ContentSpecification has one of two, mutually exclusive, nodes: `href`~~" that references the content~~ or ~~a String attribute "~~`data`~~" whose value is the data or, but not both. **[PDP-25]**~~. It has the following, general representation:

```
href: URI
```

or

```
data: String
```

When ~~"~~`href`~~"~~ is used in a ~~Content Specification~~ContentSpecification its value is interpreted as follows:

- ~~For IANA-assigned URI schemes (e.g. "http", "https", "ftp", etc.) the Provider SHALL engage the protocol as per the relevant spec. **[PDP-26]**~~ Providers SHALL support the "~~http" and "https" URI schemes.~~https" URI scheme as defined in RFC 2818 [RFC2818]. **[PDP-27]** A Provider MAY support additional URI schemes~~.~~ listed at http://www.iana.org/assignments/uri-schemes/uri-schemes.xhtml. **[PDP-28]**

- URL’s with the special scheme “pdp:” are interpreted as files contained in the PDP.
  - If the path segment (after the “pdp:”) begins with a “/” it is an absolute path.
  - If the path segment *is* “!” (i.e. the URL is “pdp:!”), the reference is to the PDP archive itself. This is useful in making an existing deployment package (such as a WAR) function as a PDP.
  - For any other path segment, the path is relative to the location of the file which contains the Content Specification, subject to the guidelines below.
  - Where the path segment contains the special character “!”, it is treated as a delimiter to look for the path to the right of “!” inside the archive at the path to the left of the “!". Providers SHALL understand this delimiter and SHALL NOT resolve any content if the archive format is unsupported. ~~**[PDP-29**~~**[PDP-29]** Consumers SHALL follow the syntax and semantics described here when using URIs with a “pdp” scheme. **[PLAN-07]** For example “pdp:/certs.zip!/id_rsa.pub” refers to a file “id_rsa.pub” contained at the root of a “certs.zip” file located at the root of the PDP, and is valid only on platforms which support the ZIP format in conjunction with “!”. On other platforms the link will not be resolved.
- Where the value is not a URI, it is interpreted as a “pdp:” protocol link, as though it were preceded by “pdp:/”.

Example 7: A ~~DP~~Plan describing an application consisting of the contents of the PDP

```
00 artifacts:
01   -
02     type: org.oasis-open.tosca:CSAR
03     content: { href: pdp:! }
04     requirements:
05       -
06         type: com.oasis-open.tosca:DeployOn
~~07           …~~
```

The above example illustrates the use of the “pdp:!” construct wherein the content being referenced (on line 03) is the PDP itself. In this case the PDP is also an OASIS TOSCA v1 Cloud Service Archive.

~~5.3.6~~4.3.6 RequirementSpecification

A RequirementSpecification describes the relationship between an artifact and a service. It has the following, general representation:

```
~~requirementType~~requirement_type: String
fulfillment: ~~[~~(String | ServiceSpecification~~] ?~~
~~…~~) ?
```

4.3.6.1 requirement_type

Type: String

Required: true

~~5.3.6.1 requirementType~~

The value of this node defines the relationship of the artifact that contains this RequirementSpecification to a service. For example, “com.java:HostOn”. See Section 4.2.1, “Types”, for a general description of the definition and treatment of these values.

It is expected that RequirementSpecifications will contain extension nodes that modify or provide additional information about the relationship that they describe. The value space and semantics of these extensions ought to be part of the definition of the value used in the “type” node. For example, the definition of the “com.java:HostOn” relationship might define a “com.java:contextPath” node whose value specifies the desired context path for the artifact when it is deployed on its selected service.

~~5.3.6.2~~4.3.6.2 fulfillment

Type: String or ServiceSpecification

Required: false

The value of this node either describes, or references a description of, the other party in the relationship (i.e. the service) defined by this RequirementSpecification. In the case where this node references a description, the value is a String that corresponds to the ~~“~~`id`~~”~~ node of a ServiceSpecification (e.g. “id:db”). In the case where this node contains the description, the value is a ServiceSpecification. See Section 4.3.4, “ServiceSpecification”, for ~~a description of this type~~details.

~~5.3.7~~4.3.7 CharacteristicSpecification

A CharacteristicSpecification describes a desired characteristic or capability of a service. It has the following, general representation.

```
~~characteristicType~~characteristic_type: String
~~… ?~~
```

The inclusion of a CharacteristicSpecification in a ServiceSpecification indicates that the characteristic being described is significant to the application, but the degree of this significance (e.g. “absolutely necessary” versus “would be nice to have”) is not indicated.

~~5.3.7.1 characteristicType~~

4.3.7.1 characteristic_type

Type: String

Required: true

The value of this node defines the characteristic being described by this CharacteristicSpecification. For example, “com.java:ServletContainer”. See Section 4.2.1, “Types”, for a general description of the definition and treatment of these values.

It is expected that CharacteristicSpecifications will contain extension nodes that modify or provide additional information about the characteristic that they describe. The value space and semantics of these extensions ought to be part of the definition of the value used in the ~~“characteristicType”~~`characteristic_type` node. For example, the definition of the “org.rubyonrails:Rails” characteristic might define a “org.rubyonrails:version” node whose value specifies the version of Rails provided by the service.

~~6~~5 Resources

The following sub-sections describe the resources defined by this specification.

~~Common~~When supporting such a Resource, a Provider SHALL implement it and serialize it as described in the corresponding sub-section. **[RE-70]**

A Consumer SHALL serialize Resource data in its requests based on the definition of this Resource as described in the corresponding sub-section. **[RE-71]**

5.1 Attribute Constraints

Resource attributes are constrained along a number of axes. These are:

5.1.1 Required

If the Required boolean constraint for an attribute of a resource type has a value of "true", then a resource of this type SHALL have the attribute present. **[RE-06]** If the value is "false" then the resource is valid with or without the attribute present.

5.1.2 Mutable

This boolean indicates the mutability of the attribute's value(s). "false" indicates that the value of the attribute, once set, SHALL NOT change for the lifetime of the resource. **[RE-07]** "true" indicates that the value of the attribute MAY change due to the actions or activity of either the provider or the Consumer. **[RE-08]**

5.1.3 Consumer-mutable

This boolean indicates the ability of a consumer to set the value of the attribute. It is only relevant for mutable attributes. "false" indicates that the value(s) of the attribute SHALL NOT be changed by Consumers. **[RE-09]** A value of "true" indicates that Consumers MAY change the value of the attribute. **[RE-10]** Note that a value of "true" does not preclude the Provider from changing the value of the attribute.

~~6.1~~5.2 Attribute Types

Resource attributes are defined using the following types:

~~6.1.1~~5.2.1 Boolean

As defined by JSON [RFC4627], a token having a literal value of either `true` or `false`. The use of this type is indicated in metadata by an *attribute_definition resource* with an `attribute_type` value of "Boolean".

~~6.1.2~~5.2.2 String

A UNICODE string as defined by JSON [RFC4627]. The use of this type is indicated in metadata by an *attribute_definition resource* with an `attribute_type` value of "String".

~~6.1.3~~5.2.3 URI

A String (see above) that conforms to the syntax defined in RFC 3986 [RFC3986]. The use of this type is indicated in metadata by an *attribute_definition resource* with an `attribute_type` value of "URI".

~~6.1.4~~5.2.4 Timestamp

A String (see above) that conforms to the syntax defined in ISO 8601 [ISO 8601:2004]. Consumers and Providers SHALL express Timestamps in UTC (Coordinated Universal Time), with the special UTC designator ("Z"). ~~[RE-65]~~**[RE-65]** The use of this type is indicated in metadata by an *attribute_definition resource* with an `attribute_type` value of "Timestamp".

~~6.1.5~~5.2.5 Link

The management model defined in this specification involves resource entity attribute values that link to other resource entities. ~~For example, one of the Platform resource entity attribute values points to Assembly Templates.~~ The "Link" type defined here is used for such attribute values.

```
{
  "href": URI
~~  "targetName,~~
  "target_name": String
~~  …~~
}
```

~~The following attributes SHALL be present in a Link.~~ ~~**[RE-01]**~~ ~~Other attributes, not defined in this specification, MAY also be present.~~ ~~**[RE-02]**~~

The use of this type is indicated in metadata by an *attribute_definition resource* with an `attribute_type` value of "Link".

~~6.1.5.1~~5.2.5.1 href

Type: URI

Required: true

Mutable: false

This attribute is the URI [~~RFC 3986~~RFC3986] of the resource referenced by this Link. ~~The value of this attribute MAY be changed by the Provider.~~ ~~**[RE-03]**~~ ~~Consumers SHALL NOT change the value of this attribute.~~ ~~**[RE-04]**~~

5.2.5.2 target_name

Type: String

Required: true

Mutable: true

Consumer-mutable: false

~~6.1.5.2 targetName~~

This attribute echoes the value of the ~~"~~`name`~~"~~ attribute of the resource referenced by this Link. The value of this attribute may be changed by the Platform. ~~Consumers SHALL NOT change the value of this attribute.~~ ~~**[RE-31]**~~

~~6.1.6 RequirementTemplateLinks~~

~~Requirement resources and Template resources are inherently related. The unsatisfied dependencies represented by Requirements are resolved by links to Templates that define services and capabilities that satisfy those dependencies. This relationship is surfaced in the CAMP resource model through the "RequirementTemplateLinks" type. By grouping together a link to a Requirement and a link to a Template this type supports the expression the concept that "this Template satisfies this Requirement".~~

```
~~{~~
~~  "requirement": Link ?,~~
~~  "template": Link ?~~
~~  …~~
~~}~~
```

~~Though both attributes of this type are optional, a valid RequirementTemplateLinks type SHALL have at least one of the following attributes: **[RE-05]**~~

~~6.1.6.1 requirement~~

~~This optional attribute is a Link type that references either a Platform Component Requirement or an Application Component Requirement resource. The absence of this attribute indicates that the enclosing attribute (i.e. the attribute that is of type 'RequirementTemplateLinks') is expressing a link to a Template with no corresponding Requirement.~~

~~6.1.6.2 template~~

~~This optional attribute is a Link type that references either a Platform Component Template or an Application Component Template resource. The absence of this attribute indicates that the enclosing attribute (i.e. the attribute that is of type 'RequirementTemplateLinks') is expressing an unsatisfied Requirement.~~

5.3 CAMP Resource Type Inheritance

Each CAMP resource has a resource type associated with it. This is specified by the attribute named `type` as defined in Section 5.4.5, "type". The resource type defines the attributes for that resource along with the constraints and semantics of those attributes. Resource types form an inheritance hierarchy with *camp_resource* (See Section 5.4, "camp_resource Resource") at its root. When a resource type (sub-type) inherits from another resource type (super-type), the sub-type inherits, and therefore includes, all the super-type's attributes along with its constraints and semantics. A sub-type can add additional attributes not present in its super-type(s). A sub-type MAY further restrict the constraints of an attribute inherited from its super-type(s). **[MO-01]** A sub-type SHALL NOT loosen the constraints of an attribute inherited from its super-type(s). **[MO-02]** As a consequence, a resource of a super-type can always be substituted with a resource of any of its sub-types. A resource type MAY inherit from more than one super-type. **[MO-03]** If there is an attribute name collision when a sub-type inherits from multiple super-types, the inherited attributes of the same name SHALL NOT contradict the constraints and semantics of the attributes defined in its super-types. **[MO-04]**

5.4 camp_resource Resource

All CAMP resources SHALL inherit directly or indirectly from this resource. **[MO-05]** This resource contains the following attributes:

~~6.21.1 Attribute Constraints~~

~~Resource attributes are constrained along a number of axes. These are:~~

~~6.2.11.1.1 Required~~

~~If the Required boolean constraint for an attribute of a resource type has a value of "true", then an instance of that resource type SHALL have the attribute present. **[RE-06]** If the value is "false" then the instance is valid with or without the attribute present.~~

~~6.2.21.1.1 Mutable~~

~~This boolean indicates the mutability of the attribute's value(s). "false" indicates that the value of the attribute, once set, SHALL NOT change for the lifetime of the resource. **[RE-07]** "true" indicates that the~~

~~value of the attribute MAY change due to the actions or activity of either the provider or the consumer. [RE-08]~~

~~6.2.31.1.1 Consumer-mutable~~

~~This boolean indicates the ability of a consumer to set the value of the attribute. It is only relevant for mutable attributes. “false” indicates that the value(s) of the attribute SHALL NOT be changed by the Consumers. **[RE-09]** A value of “true” indicates that consumers MAY change the value of the attribute. **[RE-10]** Note that a value of 'true' does not preclude the Provider from changing the value of the attribute.~~

~~6.3 Common Resource Attributes~~

~~All the resources in this specification contain the following common attributes:~~

~~6.3.1~~5.4.1 uri

Type: URI

Required: true

Mutable: false

This attribute expresses the URI of the resource.

~~6.3.2~~5.4.2 name

Type: String

Required: true

Mutable: true

Consumer-mutable: true

This attribute expresses the human-readable name of the resource.

~~6.3.3~~5.4.3 description

Type: String

Required: false

Mutable: true

Consumer-mutable: true

This attribute expresses the human-readable description of the resource.

~~6.3.4~~5.4.4 tags

Type: String[]

Required: false

Mutable: true

Consumer-mutable: true

This attribute is an array of String values that may be assigned by the provider or the user. These values can be used for keywording and terms-of-interest.

~~6.3.5~~5.4.5 type

Type: String

Required: true

Mutable: false

~~**Consumer-mutable: false**~~

This attribute expresses the CAMP resource type. Every CAMP resource type defined in this specification specifies the required value for this attribute.

~~6.3.6 representationSkew~~

5.4.6 representation_skew

Type: String

Required: false

Mutable: true

Consumer-mutable: false

The ~~`representationSkew`~~`representation_skew` attribute expresses the relationship between the information presented in the resource and the status of the platform implementation artifacts that are represented by that resource (see Section 2.7, "Representation Skew"). It is an optional, enumerated String. If present, ~~`representationSkew`~~`representation_skew` SHALL have one of the following values: **[RE-11]**

- "CREATING" – describes a resource that is in the process of being created. The client can expect that the resource will have a skew of "NONE" once this process has completed.
- "NONE" – is an assertion by the CAMP server that the information in the resource is an accurate representation of the underlying platform implementation. Absent some action by the client or some other event (e.g. platform shutdown), a resource with a skew of NONE can be expected to remain in synch with the platform implementation.
- "UNKNOWN" – indicates that the CAMP server cannot accurately depict the aspect of the platform implementation represented by this resource. Users can attempt to address the underlying issues(s) by manipulating this and/or other resources as specified by the API.
- "DESTROYING" – describes a resource that is in the process of being destroyed. The client can expect that the resource will cease to exist once this process has completed.

The absence of the `reprsentationSkew` attribute is semantically equivalent to ~~the~~a value of "NONE~~"~~ ~~value.~~".

~~The value of the `representationSkew`~~The value of the `representation_skew` attribute applies only to the resource of which it is part. The skew of any resources that are contained (via Link relationships) by another resource (e.g. in the manner in which the *assembly resource* contains *component resources*) is conveyed by the individual `representation_skew` of those sub-resources and not aggregated or "rolled up" into the containing resource.

The value of the `representation_skew` attribute affects the availability of the HTTP methods for that resource. For example, resources with a ~~`representationSkew`~~`representation_skew` value of CREATING might support the GET, HEAD and DELETE methods, but no other HTTP methods. The following table lists the methods that SHALL be supported for each ~~representationSkew~~representation_skew value. **[RE-12]**

~~representationSkew~~representation_skew value	Methods Available
CREATING	GET, DELETE
NONE	All supported methods for that resource.
UNKNOWN	All supported methods for that resource.
DESTROYING	GET

Table 5-1: ~~representationSkew~~representation_skew Available Methods

For each ~~representationSkew~~representation_skew value, CAMP Providers MAY support HTTP methods in addition to those listed in the corresponding row of Table 5-1. **[RE-13]**

~~6.4 Error Response Message Resource~~

~~Successful requests will generally return an HTTP status code of 200 (OK), 201 (Created), 202 (Accepted), or 204 (No Content), to indicate that the requested action has been successfully performed or submitted. In addition, they might include a response message body containing a representation of the requested information. However, it is possible for a number of things to go wrong. The various underlying causes are described (as discussed in the previous section) by various HTTP status codes in the range 400-499 (for client side errors) or 500-599 (for server side problems).~~

~~If a response is returned with an error status code (400-499 or 500-599), where appropriate, the server is encouraged to include a response message body containing an *ErrorMessage* object (as defined below). The text values of such messages might be used, for example, to communicate with a human user of the client side application.~~

~~The list of messages included in a single error response is encapsulated in the ErrorMessage data model.~~

~~**Field**~~	~~**Type**~~	~~**Occurs**~~	~~**Description**~~
~~message~~	~~Message~~	~~0..n~~	~~Zero or more message data for each individual message.~~

~~*Table 5-2: ErrorMessage Data Model*~~

~~An individual message contains the following fields:~~

~~**Field**~~	~~**Type**~~	~~**Occurs**~~	~~**Description**~~
~~code~~	~~String~~	~~0..1~~	~~Symbolic error code identifying the type of error reported by this message~~
~~field~~	~~String~~	~~0..1~~	~~Name of the field from the request data model that this message is associated with~~
~~hint~~	~~String~~	~~0..1~~	~~Localized text further describing the nature of the problem, possibly including potential workarounds that the client could try~~
~~text~~	~~String~~	~~1~~	~~Localized text describing the nature of the problem reported by this message~~
~~severity~~	~~String~~	~~0..1~~	~~Label indicating the severity of the error condition represented by this message~~ ~~Vendor shall publish the enumerators that are associated with this field and their semantics~~
~~stackTrace~~	~~String~~	~~0..1~~	~~Vendor specific stack trace associated with this message~~
~~source~~	~~String~~	~~0..1~~	~~Symbolic identifier of the implementation component that triggered this message~~
~~message-id~~	~~String~~	~~0..1~~	~~A unique string that identifies this particular message~~
~~profile~~	~~URI~~	~~0..1~~	~~A reference to the standard URI to indicate the meaning of this message~~

~~*Table 5-3: Message Data Model*~~

~~The *profile* attribute indicates the semantic meaning of the message which clients may handle automatically. Messages with the same profile shall adhere to the semantic requirements of that profile, but the payload (*hint*, *text*, *severity*, *stackTrace*) may be different. In other words, given a profile, clients processing the message should be able to subsequently interact with the providers in a consistent manner across.~~

~~Each provider may extend the profile to include specific scenarios and use cases.~~

~~The information captured in the *message* data element should be complementary to the HTTP status code, and could provide more detailed information. However, it SHALL NOT contradict the HTTP status code that is returned with the request. **[RE-14]**~~

~~The following table outlines the common profiles that would accompany this specification.~~

~~**Profile**~~	~~**Description**~~
~~/msg/unknown~~	~~Unknown error and information given is descriptive in nature~~
~~/msg/security~~	~~Security issues~~
~~/msg/security/authentication~~	~~An authentication error~~
~~/msg/access~~	~~Access violation error~~
~~/msg/allocation~~	~~Allocation related issues~~
~~/msg/allocation/insufficient~~	~~Insufficient resource to satisfy the request~~
~~/msg/infrastructure~~	~~Infrastructure related issues~~
~~/msg/infrastructure/maintenance~~	~~The request cannot be immediately responded due to the infrastructure being in maintenance status~~

~~*Table 5-4: Common Message Profiles*~~

~~6.5~~5.5 HTTP Method Support

As described in Section 6.1, "Transfer Protocol", Consumers use HTTP [RFC2616] to interact with CAMP-defined resources. To foster interoperability it is necessary to define the HTTP methods supported by each resource. Note that a requirement on the Provider to support a particular HTTP method on a resource does not ensure that all requests to that resource using that method will succeed; it simply guarantees that the Provider will not fail such requests with a 405 (Method Not Allowed) error.

Providers SHALL support the HTTP GET, PUT, and PATCH methods on all of the resources defined in this section. **[RE-53]** Requirements for the support of additional HTTP methods are outlined in the descriptions of each resource below. Providers MAY elect to support additional HTTP methods in addition to those described here. **[RE-54]**

~~6.6~~5.6 ~~PlatformEndpoints~~platform_endpoints Resource

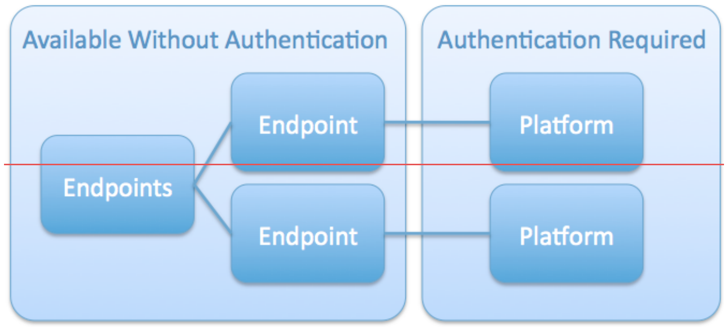
A Provider MAY concurrently offer multiple instances of the CAMP API. **[RE-15]** The primary example of why a provider might do this is to simultaneously support two or more incompatible versions/implementations of the CAMP API, but there are many reasons for a provider to offer multiple instances of the CAMP API.

Concurrent instances are supported through the use of multiple ~~instances of the Platform resource. The PlatformEndpoints~~*platform resources*. The platform_endpoints resource allows Consumers to discover all the instances of the CAMP API that are currently available. It contains an array of Links to ~~PlatformEndpoint~~platform_endpoint resources (that each reference ~~Platform~~*platform resources*), and has the following general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "~~platformEndpoints~~platform_endpoints",
  "description": String ?,
~~  "representationSkew": String ?,~~
  "tags": String[] ?,
~~  "platformEndpointLinks~~
  "representation_skew": String ?,
  "platform_endpoint_links": Link[]
}
```

Note: Because of the unique function of this resource, future versions of the CAMP specification ~~SHALL NOT make~~are cautioned against making non-backwards compatible changes to this resource. ~~[RE-16]~~

~~NOTE:~~ A Provider MAY expose the ~~PlatformEndpoints~~platform_endpoints and corresponding ~~PlatformEndpoint~~platform_endpoint resources in a way that allows for version discovery before the client has authenticated. **[RE-17]**



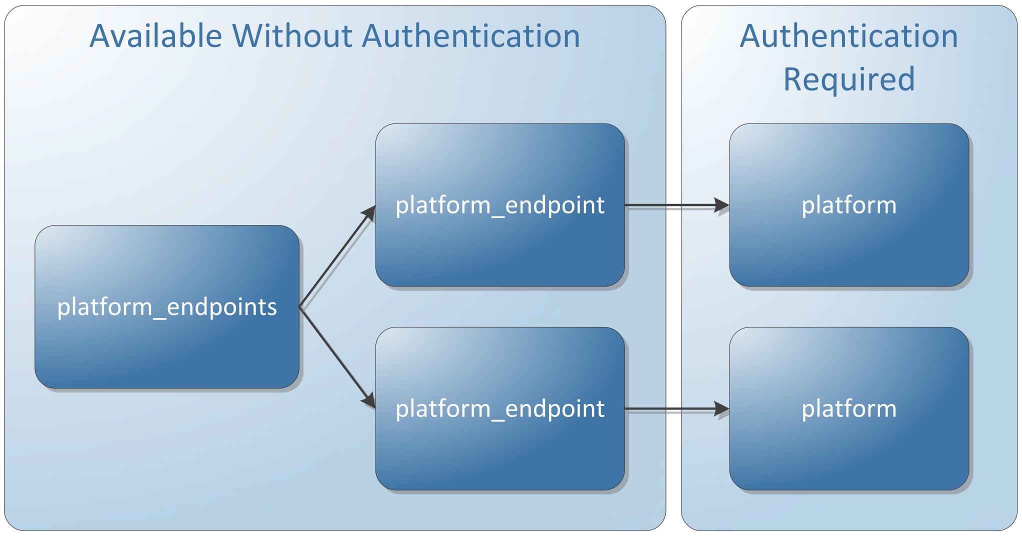


Figure 5-1: Example Implementation

The ~~PlatformEndpoints~~platform_endpoints resource contains the following attributes:

~~6.6.1 platformEndpointLinks~~

5.6.1 platform_endpoint_links

Type: Link []

Required: true

Mutable: false

This attribute is an array of Links to ~~PlatformEndpoint Resources.~~*platform_endpoint resources.* This array SHALL contain at least one ~~PlatformEndpoint Resource.~~Link. **[RE-18]** References between the resources (~~PlatformEndpoints, PlatformEndpoint~~*platform_endpoints, platform_endpoint*, and ~~Platform~~*platform*) SHALL be self-consistent. **[RE-19]**

~~6.7~~5.7 ~~PlatformEndpoint~~platform_endpoint Resource

Each ~~PlatformEndpoint Resource~~*platform_endpoint resource* SHALL refer to exactly one ~~Platform Resource~~*platform resource*, and indicate the versions supported by the Platform. **[RE-20]** This specification is deliberately silent about any relationship between resources within different ~~Platform~~*platform* trees. Each ~~PlatformEndpoint Resource~~*platform resource* could represent a different CAMP API “view” of the same applications and services. On the other hand, each ~~Endpoint~~*platform* could represent a completely independent system.

A ~~PlatformEndpoint Resource~~ *platform_endpoint resource* has the following general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "~~platformEndpoint~~platform_endpoint",
  "description": String ?,
~~  "representationSkew": String ?,~~
  "tags": String[], ?
  ~~"platformUri~~"representation_skew": String ?,
  "platform_uri": URI,
  "~~specificationVersion~~specification_version": String,

"~~backwardCompatibleSpecificationVersions~~backward_compatible_specification_vers
ions": String[] ?,
  "~~implementationVersion~~implementation_version": String~~,~~
~~  "backwardCompatibleImplementationVersions~~ ?,
  "backward_compatible_implementation_versions": String[] ~~?~~
~~}~~?,
  "auth_scheme": String ?
}
```

Note: Because of the unique function of this resource, future versions of the CAMP specification ~~SHALL NOT make~~are cautioned against making non-backwards compatible changes to this resource. ~~**[RE-21]**~~

Instances of the ~~PlatformEndpoint~~platform_endpoint resource contain the following attributes:

5.7.1 platform_uri

Type: URI

Required: true

Mutable: false

This attribute is the URI of the Platform Resource that this platform_endpoint Resource describes.

5.7.2 specification_version

Type: String

Required: true

Mutable: false

~~6.7.1 specificationVersion~~

~~**Type**: String~~

~~**Required**: true~~

~~**Mutable**: false~~

Each ~~instance of a Platform~~*platform* *resource* is the root of a tree of resources, the syntax and semantics of which conform to one or more versions of the CAMP specification. The value of this attribute is the Specification Version String of the CAMP specification that is supported by the resources rooted in the Platform referenced by the ~~platformUri~~`platform_uri` attribute of this resource.

For Platforms that implement this version of the CAMP specification, the value of this attribute SHALL be ~~"CAMP 1.1"~~ as defined in ~~section 1.7.~~Section 1.8, "Specification Version". **[RE-22]**

5.7.3 backward_compatible_specification_versions

Type: String[]

Required: false

Mutable: false

~~6.7.2 backwardCompatibleSpecificationVersions~~

~~**Type**: String[]~~

~~**Required: false**~~

~~**Mutable: false**~~

The values in this array identify each version of the CAMP specification that is backwards compatible with the current ~~specificationVersion~~`specification_version` of the Platform (referenced in the ~~platformUri~~`platform_uri` attribute of this resource). The values in this array SHALL be the Specification Version Strings of previous CAMP specification versions. **[RE-23]**

If this attribute is not present, the version of the CAMP specification implemented by the Platform (referenced in the ~~platformUri~~`platform_uri` attribute of this resource) is not backwards compatible with any previous version of the CAMP specification.

~~PlatformEndpoint~~*platform_endpoint* *resources* that reference ~~Specification Version~~*platform resources* with a `specification_version` value of "CAMP 1.1" ~~Platform Resources~~ SHALL NOT include this attribute because no previous versions are compatible. **[RE-24]**

~~6.7.3 implementationVersion~~

5.7.4 implementation_version

Type: String

Required: false

Mutable: false

Multiple implementations of the same CAMP specification MAY be offered concurrently. **[RE-25]** For example, a Provider could offer an initial beta version of "CAMP 1.1" and, later, a production version; allowing a period of overlap for their customers to migrate from the beta to the production version. The value of this attribute is an arbitrary String that expresses the Provider-specific implementation version supported by the resources rooted in the Platform (referenced in the ~~platformUri~~`platform_uri` attribute of this resource).

5.7.5 backward_compatible_implementation_versions

Type: String[]

~~6.7.4 backwardCompatibleImplementationVersions~~

Required: false

Mutable: false

~~**Type**: String[]~~

~~**Required**: false~~

~~**Mutable**: false~~

The values in this array list the provider-specific implementation versions that are backwards compatible with the implementation version of the Platform (referenced in the ~~platformUri~~`platform_uri` attribute of this resource). The values in this array are arbitrary Strings that correspond to previous ~~implementationVersion~~`implementation_version` Strings.

If this attribute is not present, the implementation version offered by the Platform (referenced in the ~~PlatformURI~~`platformURI` attribute of this resource) is not backwards compatible with any previous implementation versions.

~~6.7.5 platformUri~~

5.7.6 auth_scheme

Type: ~~URI~~String

Required: false

~~**Required: true**~~

Mutable: false

~~This attribute is the URI of the Platform Resource that this PlatformEndpoint Resource represents.~~

~~6.8 Platform Resource~~

~~A Platform~~The value of the auth_scheme attribute indicates the authentication scheme expected by the referenced Platform. For interoperability reasons, Providers are encouraged to offer at least one of the following three (case sensitive) values:

Value	**Description**
RFC2617	HTTP Basic Authentication [RFC2617]
RFC6749	OAuth2 [RFC6749]
KEYSTONE-2.0	OpenStack Keystone Authentication. [Keystone]
NONE	No authentication required.

Table 5-2 - auth_scheme values

Providers are allowed to extend this list, and provide values of their own. Absence of this attribute means that no authentication scheme is advertised.

Note: Omitting the auth_scheme attribute is discouraged for interoperability reasons.

Note: If Providers wish to offer multiple authentication schemes, they may use multiple platform_endpoint resources each with a different auth_scheme value.

5.8 platform Resource

The *platform resource* represents the Consumer’s initial view of the accessible resources and deployed entities. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "platform",
  "description": String ?,
  "~~representationSkew": String ?,~~
~~  "~~tags": String[] ?,
~~  "supportedFormatsUri~~
  "representation_skew": String ?,
  "supported_formats_uri": URI,
  "~~extensionsUri~~extensions_uri": URI,
  "~~typeDefinitionsUri~~type_definitions_uri": URI,
  "~~platformEndpointsURI~~platform_endpoints_uri": URI,
  "~~specificationVersion~~specification_version": String,
  "~~implementationVersion~~implementation_version": String ?,
  "~~assemblyTemplates": Link[] ?,~~
~~  "~~assemblies~~": Link[] ?,~~
~~  "platformComponentTemplates": Link[] ?,~~
~~  "platformComponentCapabilities": Link[] ?,~~
~~  "platformComponents": Link[] ?,~~
~~  "parameterDefinitionsUri~~_uri": URI,
  "services_uri": URI,
  "plans_uri": URI ?
}
```

~~In addition to the methods defined in Section 0, “HTTP Method Support”, Providers SHALL support the HTTP POST method on the Platform resource as described in Section 6.11, “Registering an Application”. [RE-55]~~

The ~~Platform~~*platform resource* contains the following attributes:

~~6.8.1 supportedFormatsUri~~

5.8.1 supported_formats_uri

Type: URI

Required: false

Mutable: false

This attribute is a ~~URI~~URL reference to the Formats resource for the purpose of identifying all Supported Formats for this Platform. See Section ~~5.19, “Formats Resource~~1.1.1, “formats Resource”, for details.

~~6.8.2 extensionsURI~~

~~**Type**: URI~~

~~**Required**: true~~

~~**Mutable**: false~~

5.8.2 extensions_uri

Type: URI

Required: true

Mutable: false

This attribute ~~provides~~is a ~~link~~URL reference to the Extensions this Platform supports. See Section 7.2, "~~Extensions Resource~~extensions Resource", for details.

~~6.8.3 typeDefinitionsURI~~

5.8.3 type_definitions_uri

Type: URI

Required: true

Mutable: false

This attribute ~~provides~~is a ~~link~~URL reference to the ~~TypeDefinitions~~type_definitions resource that provides information on the resource types that the Platform supports. See Section 5.18, "~~TypeDefinitions~~type_definitions Resource", for details.

5.8.4 platform_endpoints_uri

Type: URI

Required: true

~~6.8.4 platformEndpointsURI~~

~~**Type**: URI~~

~~**Requried: true**~~

Mutable: false

This attribute ~~provides~~is a ~~link~~URL reference to ~~a PlatformEndpoints~~the platform_endpoints resource. The ~~PlatformEndpoints~~platform_endpoints resource enumerates the currently available CAMP implementations. See Section 5.6, "~~PlatformEndpoints~~platform_endpoints Resource", for details.

~~6.8.5 specificationVersion~~

~~**Type**: String~~

~~**Required**: true~~

~~**Mutable**: false~~

5.8.5 specification_version

Type: String

Required: true

Mutable: false

Each ~~instance of a Platform~~*platform resource* is the root of a tree of resources, the syntax and semantics of which conform to one or more versions of the CAMP specification. The value of this attribute is the Specification Version String of the CAMP specification that is supported by the resources rooted in this Platform.

For Platforms that implement this version of the CAMP specification, the value of this attribute SHALL be ~~"CAMP 1.1" as defined in Section 1.7~~, ~~"Specification Version".~~as defined in Section 1.8, "Specification Version". **[RE-26]**

The value of this attribute SHALL exactly match the value of the ~~specificationVersion~~`specification_version` attribute of any ~~PlatformEndpoint~~*platform_endpoint resource* that references this ~~Platform Resource~~*platform resource*. **[RE-27]**

~~6.8.6 implementationVersion~~

5.8.6 implementation_version

Type: String

Required: false

Mutable: false

A Provider MAY choose to offer multiple implementations of the same CAMP specification. **[RE-28]** For example, a Provider could offer an initial beta version of "CAMP 1.1" and, later, a production version; allowing a period of overlap for their customers to migrate from the beta to the production version. The value of this attribute is an arbitrary String that expresses the Provider-specific implementation version supported by the resources rooted in this Platform.

The value of this attribute SHALL exactly match the value of the ~~implementationVersion~~`implementation_version` attribute of any ~~PlatformEndpoint~~*platform_endpoint* *resource* that references this ~~Platform~~*platform* *resource*. **[RE-29]**

5.8.7 assemblies_uri

Type: URI

~~6.8.7 assemblyTemplates~~

~~**Type**: Link[]~~

~~**Required**: false~~

Required: true

Mutable: ~~true~~false

~~**Consumer-mutable: false**~~

~~This attribute is an array of Links to the AssemblyTemplates that are accessible to the Consumer.~~

~~6.8.8 assemblies~~

This attribute is a URL reference to the *assemblies resource*. The *assemblies resource* enumerates the applications deployed on this platform. See Section 5.9, "assemblies Resource", for details.

5.8.8 services_uri

Type: ~~Link[]~~URI

~~**Required**: false~~

Required: true

Mutable: ~~true~~false

~~**Consumer-mutable: false**~~

This attribute is a URL reference to the *services resource*. The *services resource* enumerates the services available to the Consumer on this platform. See Section 5.12, "services Resource", for details.

5.8.9 plans_uri

Type: URI

Required: false

Mutable: false

This attribute is a URL reference to the *plans resource*. The (optional) *plans resource* enumerates the *plans* deployed on this platform. See Section 5.14, "plans Resource", for details.

5.9 assemblies Resource

This resource acts as a container for the *assembly resources* on this platform. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "assemblies",
  "description": String ?,
  "tags": String[] ?,
  "representation_skew": String ?,
  "assembly_links": Link[] ?,
  "parameter_definitions_uri": URI
}
```

In addition to the methods defined in Section 1.1, "HTTP Method Support", Providers SHALL support the HTTP POST method on the *assemblies resource* as described in Section 6.11, "Deploying an Application". **[RMR-02]**

The *assemblies resource* contains the following attributes:

5.9.1 assembly_links

Type: Link[]

Required: false

Mutable: true

Consumer-mutable: false

This attribute contains Links to the *assembly resources* that represent the applications deployed on this platform.

5.9.2 parameter_definitions_uri

Type: URI

Required: true

Mutable: false

The value of the `parameter_definitions_uri` attribute references a resource that contains links to *parameter_definitions resources* that describe the parameters accepted by this resource on an HTTP POST method. Each of the *parameter_definition resources* provides metadata for a parameter as described in Section 5.21, "parameter_definitions Resource". The *assemblies resource* accepts the `pdp_uri`, `plan_uri`, `pdp_file`, or `plan_file` parameters to create a new *assembly resource* upon a POST. The *assemblies resource* SHALL indirectly reference *parameter_definition resources* that describes the `pdp_uri`, `plan_uri`, `pdp_file`, and `plan_file` parameters. **[RMR-03]**

5.10 assembly Resource

An *assembly resource* represents an instantiated application at runtime. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "assembly",
  "description": String ?,
  "tags": String[] ?,
  "representation_skew": String ?,
  "components": Link[],
  "plan_uri": URI ?,
  "operations_uri": URI ?,
  "sensors_uri": URI ?
}
```

In addition to the methods defined in Section 1.1, “HTTP Method Support”, Providers SHALL support the HTTP DELETE method on the *assembly resource*. **[RE-61]** On reception of a DELETE request a Provider SHALL remove the *assembly resource* from the system along with any *component resources* referenced by that assembly resource. (i.e. the tree of resources that was created when the application was instantiated). **[RE-73]** On reception of a DELETE request a Provider SHALL remove the reference to the *assembly resource* from the *assemblies resource’s* `assembly_links` array. **[RE-74]**

An *assembly resource* contains the following attributes:

5.10.1 components

Type: Link[]

Required: true

Mutable: true

Consumer-mutable: false

The value of the `components` attribute is an array of Links to the *component resources* that make up this *assembly resource*. An *assembly resource* SHALL have at least one reference to a *component resource*. **[RE-39]**

5.10.2 plan_uri

Type: URI

Required: false

Mutable: false

The value of this optional attribute is a URL reference to the *plan resource* for this *assembly resource*. Providers that support Plans SHALL include this attribute in all *assembly resources*. **[RMR-04]**

5.10.3 operations_uri

Type: URI

Required: false

Mutable: false

This attribute contains the URI of the *operations resource*. The *operations resource* lists the *operation resource* links available for the *assembly resource*.

5.10.4 sensors_uri

Type: URI

Required: false

Mutable: false

This attribute contains a URI of the *sensors resource* listing the *sensor resources* available on this resource.

5.11 component Resource

A *component* represents a runtime component. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "component",
  "description": String ?,
  "tags": String[] ?,
  "representation_skew": String ?,
  "assemblies": Link[],
  "artifact": URI ?,
  "service": URI ?,
  "status": String,
  "external_management_resource": URI ?
  "related_components": Link[] ?,
  "operations_uri": URI ?,
  "sensors_uri": URI ?
}
```

In addition to the methods defined in Section 1.1, "HTTP Method Support", Providers SHALL support the HTTP DELETE method on the *component resource*. **[RE-62]** A successful DELETE request stops the underlying component, removes the *component resource* from the system, and removes its reference from the `components` array of its containing *assembly resource*.

Each *component resource* contains the following attributes:

5.11.1 assemblies

Type: URI

Required: true

Mutable: true

Consumer-mutable: false

The value of the `assemblies` attribute is an array of Links that reference to the *assembly resources* of which this *component resource* is a member.

5.11.2 artifact

Type: URI

Required: false

Mutable: false

The value of the `artifact` attribute is a URL reference to the artifact on which this *component resource* is based. This artifact is not a CAMP resource, but a representation of the actual artifact (e.g. WAR file, Ruby gem file, etc.)

The `artifact` attribute and the `service` attribute are mutually exclusive.

5.11.3 service

Type: URI

Required: false

Mutable: false

~~Assembly resources that~~ The value of the `service` attribute is a URL reference to the *service resource* on which this *component resource* is based.

The `service` attribute and the `artifact` attribute are ~~accessible to the Consumer~~mutually exclusive.

~~6.8.9 platformComponentTemplates~~

5.11.4 status

Type: ~~Link[]~~String

Required: ~~false~~true

Mutable: true

Consumer-mutable: false

The value of this attribute indicates the status of the component represented by the *component resource*. This attribute MAY have one of the following values:

- "RUNNING" – indicates that the component is functioning as expected.
- "ERROR" – indicates that the component has encountered some sort of error. **[RE-68]**

Providers MAY extend this list with additional values. **[RE-69]**

The value of this attribute can change in response to the invocation of an operation (see Section 5.24, "operation Resource") or as a result of some change in the underlying system.

As with other attributes, the value of this attribute cannot be construed to accurately reflect the status of the underlying component if the `representation_skew` has a value other than "NONE".

5.11.5 external_management_resource

Type: URI

Required: false

Mutable: false

A URI to an external management interface to manage the underlying component (such as an IaaS API to manage the virtual machines that support this component). The entity referred to by this attribute is platform dependent and requires external documentation to understand its meaning.

5.11.6 related_components

Type: Link[]

Required: false

Mutable: false

This attribute is an array of Links to the ~~PlatformComponentRequirement~~*component resources* that ~~are accessible to the Consumer~~this *component* is related to.

5.11.7 operations_uri

Type: URI

Required: false

Mutable: false

This attribute contains the URI of the *operations resource*. The *operations resource* lists the *operation resource* links available for the *component resource*.

5.11.8 sensors_uri

Type: URI

Required: false

Mutable: false

This attribute contains a URI of the *sensors resource* listing the *sensor resources* available on this resource.

5.12 services Resource

This resource acts as a container for the *service resources* of this platform. This resource has the following, general representation:

~~6.8.10~~ {
"uri": *URI*,
"name": *String*,
"type": "assemblies",
"description": *String* ?,
"tags": *String[]* ?,
"~~platformComponentCapabilities~~

```
representation_skew": String ?,
  "service_links": Link[] ?,
}
```

The Services resource contains the following attributes:

5.12.1 service_links

Type: Link[]

Required: false

Mutable: true

Consumer-mutable: false

This attribute ~~is an array of~~ contains Links to the ~~PlatformComponentCapability~~*service resources* that ~~are accessible~~represent the services available to the Consumer.

5.13 service Resource

A *service resource* represents a particular configuration of a service available for use by one or more applications. This resource has the following, general representation:

~~6.8.11 platformComponents~~

~~**Type**: Link[]~~

~~**Required**: false~~

~~**Mutable**: true~~

~~**Consumer-mutable**: false~~

~~This attribute is an array of Links to the PlatformComponent resources that are accessible to the Consumer.~~

~~6.8.12 parameterDefinitionsUri~~

~~**Type**: URI~~

~~**Required**: true~~

~~**Mutable**: true~~

~~**Consumer-mutable**: false~~

~~parameterDefinitionsURI points to a resource that contains links to parameterDefinitions that describe the parameters accepted by this resource on an HTTP POST method. Each of the parameterDefinitions provide metadata for a parameter as described in Section **Error! Reference source not found.**, "ParemeterDefinitions Resource". The Platform resource accepts the pdpUri parameter to create new~~

~~AssemblyTemplate resources upon a POST. The Platform resource SHALL indirectly reference a parameterDefinition resource that describes the pdpUri parameter. **[RE-30]**~~

~~6.9 AssemblyTemplate Resource~~

~~An Assembly Template is the template for the creation of Assemblies. This resource has the following, general representation:~~

```
~~{~~
~~  "uri": URI,~~
~~  "name": String,~~
~~  "type": "assemblyTemplate~~{
  "uri": URI,
  "name": String,
  "type": "service",
  "description": String ?,
~~  "representationSkew": String ?,~~
  "tags": String[] ?,
~~  "applicationComponentTemplates": Link[] ?,~~
  "representation_skew": String ?,
  "parameter_definitions_uri": URI ?
}
~~  "parameterDefinitionsUri": URI ?,~~
~~  "pdpUri": URI ?,~~
~~  "dpUri": URI ?~~
~~}~~
```

~~In addition to the methods defined in Section 0, "HTTP Method Support", Providers SHALL support the HTTP POST and DELETE methods on instances of the AssemblyTemplate resource. **[RE-56]** As described in Section 6.12, "Instantiating an Application", a successful POST request results in the instantiation of a new Assembly resource. As described in Section 6.14, "Deleting an Application Instance and a Deployed Application", a successful DELETE request removes the AssemblyTemplate from the system along with any ApplicationComponentTemplates referenced by the AssemblyTemplate and any PlatformComponentRequirements or ApplicationComponentRequirements referenced by those ApplicationComponentTemplates (i.e. the tree of resources that was created when the application was registered). A successful DELETE also removes the reference to the AssemblyTemplate from the Platform resource's `assemblyTemplates` array.~~

~~Instances of the AssemblyTemplate resource contain~~The *service resource* contains the following attributes:

5.13.1 parameter_definitions_uri

Type: URI

Required: false

Mutable: false

~~6.9.1 applicationComponentTemplates~~

~~**Type**: Link[]~~

~~**Required**: false~~

~~**Mutable**: true~~

~~**Consumer-mutable**: false~~

~~This attribute is an array of Links to the ApplicationComponentTemplate resources that are part of this AssemblyTemplate. An AssemblyTemplate MAY have zero~~ This attribute references ~~to ApplicationComponentTemplate resources, **[RE-32]** but Providers SHALL NOT instantiate an Assembly using an AssemblyTemplate that does not reference at least one ApplicationComponentTemplate. **[RE-33]**~~

~~6.9.2 parameterDefinitionsUri~~

~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: false~~

~~**Consumer-mutable: false**~~

~~This attribute references~~ the URI of the ~~ParameterDefinitions~~*parameter_definitions* resource that defines parameters that may be passed to this resource. The ~~ParameterDefinitions~~*parameter_definitions* resource referenced by this attribute SHALL define parameters to allow setting the 'name', 'description', and 'tags' attributes of any new resource created in the course of interacting with this resource. **[RE-~~34~~37]**

~~6.9.3 pdpUri~~

~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: true~~

~~**Consumer-mutable**: false~~

~~This~~If this attribute ~~specifies~~ is present in the ~~URI of the PDP from which the Assembly Template~~ resource ~~was created. If present, it allows one to register another Assembly Template from the same PDP using the~~, Providers SHALL support the POST method on that resource in addition to the methods defined in Section ~~6.11.1, "Registering a PDP by reference".~~1.1, "HTTP Method Support". **[RE-38]**

5.14 plans Resource

This optional resource acts as a container for the *plan resources* deployed by the Consumer. This resource has the following, general representation:

~~6.9.4 dpUri~~

~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: true~~

~~**Consumer-mutable**: false~~

~~This attribute specifies the URI of the Deployment Plan from which the Assembly Template resource was created. If present, it allows one to register another Assembly Template from the same DP using the method defined in Section 6.11.1, "Registering a PDP by reference".~~

~~6.10 ApplicationComponentTemplate Resource~~

~~An Application Component Template represents the definition of the deployable Component. This resource has the following, general representation:~~

```
{
  "uri": URI,
  "name": String,
  "type": "~~applicationComponentTemplate~~plans",
  "description": String ?,
~~  "representationSkew": String ?,~~
~~  "tags": String[] ?,~~
~~  "assemblyTemplate": Link,~~
~~  "applicationComponentDependencies": RequirementTemplateLinks[] ?,~~
~~  "platformComponentDependencies": RequirementTemplateLinks[] ?~~
  "tags": String[] ?,
  "representation_skew": String ?,
  "plan_links": Link[] ?,
  "parameter_definitions_uri": URI
}
```

In addition to the methods defined in Section 1.1, "HTTP Method Support", Providers SHALL support the HTTP POST method on the *plans resource* as described in Section 6.12, "Registering a Plan". **[RMR-05]**

The Plans resource contains the following attributes:

5.14.1 plan_links

Type: Link[]

Required: false

Mutable: true

Consumer-mutable: false

~~In addition to the methods defined in Section 0, "HTTP Method Support", Providers SHALL support the HTTP DELETE method on instances of the ApplicationComponentTemplate resource.~~ ~~**[RE-57]**~~ ~~A successful DELETE request removes the ApplicationComponentTemplate from the system along with any PlatformComponentRequirements or ApplicationComponentRequirements referenced by that ApplicationComponentTemplates (i.e. the tree of resources rooted in the ApplicationComponentTemplate). A successful DELETE also removes the reference to the ApplicationComponentTemplate from the `applicationComponentTemplates` array of its containing AssemblyTemplate.~~

~~Instances of the ApplicationComponentTemplate resource contain the following attributes:~~

~~6.10.1 assemblyTemplate~~

~~**Type:** Link~~

~~**Required**: true~~

~~**Mutable**: true~~

~~**Consumer-mutable: true**~~

~~The attribute contains the link to the AssemblyTemplate that contains this resource.~~ This attribute ~~SHALL be present.~~ ~~**[RE-35]**~~

~~6.10.2 applicationComponentDependencies~~

~~**Type:** RequirementTemplateLinks[]~~

~~**Required**: false~~

~~**Mutable**: true~~

~~**Consumer-mutable: true**~~

~~This attribute is an array of RequirementTemplateLinks that reference related pairs of ApplicationComponentRequirement and ApplicationComponentTemplate resources.~~

~~If one of the elements in this array has a 'requirement' attribute with no matching 'template' attribute that indicates that this ApplicationComponentTemplate has an unsatisfied requirement and that the ApplicationComponent described by this template cannot be instantiated.~~

~~6.10.3 platformComponentDependencies~~

~~**Type:** RequirementTemplateLinks[]~~

~~**Required: false**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~This attribute is an array of RequirementTemplateLinks that reference related pairs of PlatfromComponentRequirement and PlatformComponentTemplate resources.~~

~~If one of the elements in this array has a 'requirement' attribute with no match 'template' attribute that indicates that this ApplicationComponentTemplate has an unsatisfied requirement and that the ApplicationComponent described by this template cannot be instantiated.~~

~~6.11 ApplicationComponentRequirement Resource~~

~~An Application Component Requirement represents an Application Component's requirement on another Application Component and its required range of component capabilities. Each Application Component Requirement implements this class and adds its' own Attribute Ranges to the list below. This resource has the following, general representation:~~

```
~~{~~
~~  "uri": URI,~~
~~  "name": String,~~
~~  "type": "applicationComponentRequirement",~~
~~  "description": String ?,~~
~~  "representationSkew": String ?,~~
~~  "tags": String[] ?~~
~~}~~
```

~~In addition to the methods defined in Section 0, "HTTP Method Support", Providers SHALL support the HTTP DELETE method on instances of the ApplicationComponentRequirement resource. **[RE-58]** A successful DELETE request removes the ApplicationComponentRequirement from the system as well as removing its reference from the `applicationComponentDependencies` array of any ApplicationComponentTemplates that refer to it.~~

~~6.12 ApplicationComponentCapability Resource~~

~~An Application Component Capability represents the definition of an Applicaton Component's range of component capabilities.~~ contains Links to the *plan resources* that represent the blueprints~~This resource has the following, general representation:~~

```
~~{~~
~~  "uri": URI,~~
~~  "name": String,~~
~~  "type": "applicationComponentCapability",~~
~~  "description": String ?,~~
~~  "representationSkew": String ?,~~
~~  "tags": String[] ?~~
~~}~~
```

~~In addition to the methods defined in Section 0, "HTTP Method Support", Providers SHALL support the HTTP DELETE method on instances of the ApplicationComponentCapability resource. **[RE-59]**~~

~~6.13 PlatformComponentTemplate Resource~~

~~A Platform Component Template represents the desired configuration of a Platform Component with specific values~~ for applications deployed on the platform.

5.14.2 parameter_definitions_uri

Type: URI

Required: true

Mutable: false

~~the component capabilities. The specified value for each component attribute SHALL be in the range defined in the corresponding Platform Component Capability.~~ ~~**[RE-36]**~~ ~~This resource has the following, general representation:~~

```
~~{~~
~~  "uri": URI,~~
~~  "name": String,~~
~~  "type": "platformComponentTemplate",~~
~~  "description": String ?,~~
~~  "representationSkew": String ?,~~
~~  "tags": String[] ?,~~
~~  "parameterDefinitionsUri": URI ?~~
~~}~~
```

~~Instances of the PlatformComponentTemplate resource contain the following attributes:~~

~~6.13.1 parameterDefinitionsUri~~

~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: false~~

~~**Consumer-mutable: false**~~

~~This attribute references the URI of the ParameterDefinitions resource that defines parameters that may be passed to this resource. The ParameterDefinitions resource referenced by this attribute SHALL define parameters to allow setting the 'name', 'description', and 'tags' attributes of any new resource created in the course of interacting with this resource.~~ ~~**[RE-37]**~~

~~If this attribute is present in an instance of this resource, Providers SHALL support the POST method on that instance in addition to the methods defined in Section 0, "HTTP Method Support".~~ ~~**[RE-38]**~~

~~6.14 PlatformComponentRequirement Resource~~

~~A Platform Component Requirement represents an Application Component's requirement on a Platform Component and its~~ The value of the `parameter_definitions_uri` attribute references a resource that contains links to *parameter_definition resources* that describe the parameters accepted by this resource on an HTTP POST method. Each of the *parameter_definition resources* provides metadata for a parameter as described in Section 5.21, "parameter_definitions Resource". The Plans resource accepts the `pdp_uri`, `plan_uri`, `pdp_file`, or `plan_file` parameters to create a new *plan resource* upon a POST. The *plans resource* SHALL indirectly reference *parameter_definition resources* that describe the pdp_uri, plan_uri, pdp_file, and plan_file parameters. **[RMR-06]**

5.15 plan Resource

This optional resource stores the *plan* for an application. As discussed in Section 0, "Deployment", this information is supplied to the platform as part of the operation of deploying an application.~~required range of component capabilities.~~ This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "~~platformComponentRequirement~~plan",
  "description": String ?,
~~  "representationSkew": String ?,~~
  "tags": String[] ~~?~~
~~}~~
```

~~In addition to the methods defined in Section 0, "HTTP Method Support", Providers SHALL support the HTTP DELETE method on instances of the PlatformComponentRequirement resource.~~ ~~**[RE-60]**~~ ~~A successful DELETE request removes the PlatformComponentRequirement from the system as well as removing its reference from the `platformComponentDependencies` array of any ApplicationComponentTemplates that refer to it.~~

~~6.15 PlatformComponentCapability Resource~~

~~A Platform Component Capability represents the definition of Platform Component and its range of component capabilities. This resource has the following, general representation:~~

```
~~{~~
~~  "uri": URI,~~
?,
  "representation_skew": String ?,
  "camp_version": String,
  "origin": String ?,
  "artifacts": [
    {
      "name": String~~,~~
~~  "type": "platformComponentCapability",~~
 ?,
      "description": String ?,
~~  "representationSkew": String ?,~~

      "tags": String[] ~~?~~
~~}~~
```

~~6.16 Assembly Resource~~

~~An Assembly represents an instantiated Application at runtime. This resource has the following, general representation:~~

```
~~{~~
~~  "uri": URI,~~
?,
      "artifact_type": String,
      "content": { "href": URI },
      "requirements": [
        {
          "requirement_type": String,
          "fulfillment": {
            "name": String~~,~~
~~  "type": "assembly",~~
 ?,
            "description": String ?,
~~  "representationSkew": String ?,~~

            "tags": String[] ?,
~~  "applicationComponents": Link[] ?,~~
~~  "assemblyTemplate": Link~~
~~  "operationsUri": URI ?,~~
~~  "sensorsUri": URI ?~~
~~}~~
```

~~In addition to the methods defined in Section 0, “HTTP Method Support”, Providers SHALL support the HTTP DELETE methods on instances of the Assembly resource as described in Section 6.14, “Deleting an Application Instance and a Deployed Application”.~~ ~~**[RE-61]**~~ ~~A successful DELETE request removes the Assembly resource from the system along with any ApplicationComponents and PlatformComponents referenced by that Assembly. (i.e. the tree of resources that was created when the application was instantiated). A successful DELETE also removes the reference to the Assembly resource from the Platform resource’s `assemblies` array.~~

~~Instances of the Assembly resource contain the following attributes:~~

~~6.16.1 applicationComponents~~

~~**Type:** Link[]~~

~~**Required**: true..*~~

~~**Mutable**: false~~

~~This attribute is an array of Links to the ApplicationComponent resources that are part of this Assembly. An Assembly resource SHALL have at least one reference to an ApplicationComponent resource.~~ ~~**[RE-39]**~~

~~6.16.2 assemblyTemplate~~

~~**Type:** Link~~

~~**Required**: true~~

~~**Mutable**: false~~

~~This attribute is a Link to the AssemblyTemplate resource from which this Assembly was created.~~

~~6.16.3 operationsUri~~

~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: false~~

~~**Consumer-mutable: false**~~

~~This attribute contains a URI of the Operations resource listing the Operation resources available on this resource.~~

~~6.16.4 sensorsUri~~

~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: false~~

~~**Consumer-mutable: false**~~

~~This attribute contains a URI of the Sensors resource listing the Sensor resources available on this resource.~~

~~6.17 ApplicationComponent Resource~~

~~An Application Component represents an instantiated Component at runtime. This resource has the following, general representation:~~

```
~~{~~
~~  "uri": URI,~~
~~  "name": String,~~
~~  "type": "applicationComponent",~~
~~  "description": String ?,~~
~~  "representationSkew~~
                    "id": String ?,

                    "href": URI ?,
                    "characteristics": {
                      characteristic: String +
                    }
                  } ?
                } +
              ] ?,
            } +
          ] ?,
  "services": [
    {
      "name": String ?,
      "description": String ?,
      "tags": String[] ?,
~~  "assembly": Link,~~
~~  "applicationComponents": Link[] ?,~~
~~  "platformComponents": Link[] ?,~~
~~  "operationsUri": URI ?,~~
~~  "sensorsUri": URI~~
      "id": String ?,
      "href": URI ?,
      "characteristics": {
        characteristic: String +
      }
    } +
  ] ?
}
```

~~In addition to the methods defined in Section 0, "HTTP Method Support", Providers SHALL support the HTTP DELETE method on instances of the ApplicationComponent resource.~~ ~~**[RE-62]**~~ ~~A successful DELETE request stops the underlying component, removes the ApplicationComponent resource from the system, and removes its reference from the `applicationComponents` array of its containing Assembly.~~

~~Instances of the ApplicationComponent resource contain the following~~ The schema of the *plan resource* returned from a CAMP Provider SHALL conform to the schema for Plans described in Section 4.3, "Plan Schema", with the following additional requirements: **[RMR-07]**

- Representations of the *plan resource* SHALL be serialized as JSON, unless another format is negotiated. **[RMR-08]**

Any href attributes~~:~~

~~6.17.1 assembly~~

~~**Type:** Link~~

of ServiceSpecifications SHOULD refer~~**Required**: true~~

~~**Mutable**: false~~

- ~~This attribute is a Link~~ to ~~the Assembly resource of which this ApplicationComponent is a member.~~a Service resource. **[RMR-09]**

~~6.17.2 applicationComponents~~

~~**Type**: Link[]~~

~~**Required**: false~~

~~**Mutable**: false~~

~~This attribute is an array of Links to the ApplicationComponent resources that this ApplicationComponent depends on.~~

~~6.17.3 platformComponents~~

~~**Type**: Link[]~~

~~**Required**: false~~

~~**Mutable**: false~~

~~This attribute is an array of Links to the PlatformComponent resources that this ApplicationComponent depends on.~~

~~6.17.4 operationsUri~~

~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: false~~

~~**Consumer-mutable: false**~~

~~This attribute contains a URI of the Operations resource listing the Operation resources available on this resource.~~

~~6.17.5 sensorsUri~~

- All href attributes in the *plan resource* SHOULD be set to a consumer accessible URL. If the original Plan file referred to a local file, the URL indicates where the Provider stored the content. **[RMR-10]**
- The *plan resource* inherits from the *camp_resource* defined in Section 5.4, “camp_resource Resource”, and therefore inherits all its attributes. The value for the `type` attribute is “plan”.

For example, if the consumer-supplied Plan file describes an artifact with an href pointing to a file contained in a PDP, the platform-supplied *plan resource* will point to a copy of that artifact, such as one hosted at the platform or in an object store.

Support for the *plan resource* is uniform across a CAMP implementation. Regardless of whether a Consumer attempts to create an *assembly resource* by POSTing to the *assemblies resource* or creates a *plan resource* by POSTing to the *plans resource*, a Provider that supports *plans* and *plan resources* SHALL create a *plan resource* for every deployed application. **[RMR-11]**

5.15.1 Advertising Support for the Plan Resource

As an aid to interoperability it is helpful if Consumers can easily discover if a particular Provider supports the *plans resource* and *plan resources*. Section 7.2, "extensions Resource", defines a mechanism for advertising extensions to the CAMP specification. This mechanism is used to advertise support for the *plans resource* and *plan resources*.

Providers that support the *plans* and *plan*~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: false~~

~~**Consumer-mutable: false**~~

~~This attribute contains a URI of the Sensors resource listing the Sensor~~ *resources* ~~available on this resource.~~

~~6.18 PlatformComponent Resource~~

~~A Platform Component represents the runtime instance of a platform component and its configuration of component attributes as well as metrics associated with those attributes. This resource has~~SHALL advertise such support using the following~~, general representation:~~ *extension resource*: **[RMR-12]**

```
{
  "uri": ~~URI,~~<as appropriate>,
  "name": ~~String,~~
"CAMP Plans Extension",
  "type": "~~platformComponent~~extension",
  "description": ~~String ?,~~
~~  "representationSkew~~"indicates~~": String ?,~~
~~  "tags": String[] ?,~~
~~  "externalManagementResource": URI ?~~
~~  "operationsUri": URI ?,~~
~~  "sensorsUri": URI ?~~
~~}~~
```

~~In addition to the methods defined in Section 0, "HTTP Method Support", Providers SHALL~~ support ~~the HTTP DELETE method on instances of the PlatformComponent resource.~~ ~~**[RE-63]**~~ ~~A successful DELETE request stops the underlying component, removes the PlatformComponent resource from the system,~~for the plans and ~~removes its reference from the Platform resource's `platformComponents` array.~~

~~Instances of the Platform Component resource contain the following attributes:~~

~~6.18.1 externalManagementResource~~

~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: false~~

```
~~A URI to an external management interface to manage this platform component
(such as an IaaS API to manage the virtual machines that support this
component). This is platform dependent and requires external ~~plan resources",
  "version": "CAMP 1.1",
  "documentation~~ to understand its meaning.~~": "http://docs.oasis-
open.org/camp/camp-spec/v1.1/camp-spec-v1.1.pdf"
}
```

~~6.18.2 operationsUri~~

~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: false~~

~~**Consumer-mutable: false**~~

~~This attribute contains a URI of the Operations resource listing the Operation resources available on this resource.~~

~~6.18.3 sensorsUri~~

formats~~**Type**: URI~~

~~**Required**: false~~

~~**Mutable**: false~~

~~**Consumer-mutable: false**~~

~~This attribute contains a URI of the Sensors resource listing the Sensor resources available on this resource.~~

~~6.19~~5.16 ~~Formats~~ Resource

The Formats resource contains an array of Links to Format resources. It allows the identification of Supported Formats. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "formats",
  "description": String ?,
  "~~representationSkew": String ?,~~
~~  "~~tags": String[] ?,
~~  "formatLinks~~
  "representation_skew": String ?,
  "format_links": Link[]
}
```

The Formats resource contains the following attribute:

~~6.19.1 formatLinks~~

5.16.1 format_links

Type: Link[]

Required: true

Mutable: false

This attribute contains Links to Format resources that contain information about data serialization formats supported by the Platform. For every format that the Platform supports, there SHALL be a Format resource Link that represents such a format. **[RE-40]** The Required JSON Format Resource SHALL be listed first in the ~~formatLinks~~format_links array. **[RE-41]**

~~6.20~~5.17 ~~Format~~format Resource

A Format resource represents exactly one supported data serialization format. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "format",
  "description": String ?,
  "~~representationSkew": String ?,~~
~~  "~~tags": String[] ?,
~~  "mimeType~~
  "representation_skew": String ?,
  "mime_type": String,
  "version": String~~,~~ ?,
  "documentation": URI
}
```

Instances of the Format resource contain the following attributes:

~~6.20.1 mimeType~~

5.17.1 mime_type

Type: String

Required: true

Mutable: false

This attribute contains the mime-type to be used by the Platform in HTTP [RFC2616] compliant content negotiation for this Format. For example: “application/json”.

~~6.20.2~~5.17.2 version

Type: String

Required: false

Mutable: false

This attribute contains the version identifier of the data serialization format used.

~~6.20.3~~5.17.3 documentation

Type: URI

Required: true

Mutable: false

~~6.20.4~~5.17.4 Required JSON Format Resource

The Required JSON Format Resource is defined as:

```
{
  "uri": URI,
  "name": "JSON",
  "type": "format",
  "description": "JavaScript Object Notation",
  "tags": ~~[~~ String~~, + ], ?~~
~~  "mimeType~~[] ?,
  "mime_type": "application/json",
  "version": "RFC4627",
  "documentation": "http://www.ietf.org/rfc/rfc4627.txt~~",~~"
}
~~  "representationSkew": String ?~~
~~}~~
```

The *name*, ~~*mimeType*~~*mime_type*, *version*, and *documentation* attribute values for the JSON Format Resource SHALL reflect the above values. **[RE-42]**

~~6.21~~5.18 ~~TypeDefinitions~~type_definitions Resource

This resource contains an array of Links to ~~all TypeDefinition~~the *type_definition* *resources*. The ~~Platform~~*platform* *resource* SHALL provide a Link to the ~~TypeDefinitions~~*type_definitions* *resource* in the required attribute named ~~typeDefinitionsUri~~`type_definitions_uri`. **[RE-43]** This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "typeDefinitions",
  "description": String ?,
  "~~representationSkew": String ?,~~
~~  "~~tags": String[] ?,
~~  "typeDefinitionLinks~~
  "representation_skew": String ?,
  "type_definition_links": Link[]
}
```

The ~~TypeDefinitionss~~*type_definitions* *resource* contains the following attribute:

5.18.1 type_definition_links

Type: Link[]

Required: true

Mutable: false

~~6.21.1 typeDefinitionLinks~~

~~**Type**: Link[]~~

~~**Required**: true~~

~~**Mutable**: false~~

This attribute contains Links to ~~TypeDefinition~~*type_definition* *resources* that contain information about resource types supported by the Platform. If the Platform does not extend this specification to add new resource types then the array can be empty. If the array is non-empty, for every resource type that the Platform supports, there SHALL be a ~~TypeDefinition~~*type_definition* *resource* Link that represents such a resource type. **[RE-44]** To help developers implement this requirement a package containing the *type_definition resources* for every resource defined in this specification is provided as a non-normative auxiliary file.

~~6.22~~5.19 ~~TypeDefinition~~type_definition Resource

A ~~TypeDefinition~~*type_definition* *resource* ~~represents exactly one~~describes a resource type supported by the Platform. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "typeDefinition",
  "description": String ?,
~~  "representationSkew": String ?,~~
  "tags": String[] ?,
  "representation_skew": String ?,
  "documentation": URI,
  "~~attributeDefinitionLinks~~inherits_from": Link[],
  "attribute_definition_links": AttributeLink[]
}
```

~~Instances of the TypeDefinition resource contain the following attributes:~~

The value of the `name` attribute in a *type_definition resource* SHALL match the value of the `type` attribute for the resource type that it describes. **[RE-75]** This constraint allows Consumers to locate the metadata that describes a resource in the `typeDefinitions` array of the *type_definitions resource* using the `type` value of that resource as a key.

The *type_definition resource* contains the following attributes:

~~6.22.1~~5.19.1 documentation

Type: URI

Required: true

Mutable: false

This attribute contains a URI that points to the documentation for the resource type. For resource types that are defined in this specification, the URI can point to this specification.

5.19.2 inherits_from

Type: Link[]

Required: false

Mutable: false

This attribute contains an array of Links. Each Link in this array points to a *type_definition resource* that the described resource's type inherits from. Links in this array SHALL NOT either directly or transitively point to the described resource. **[MO-06]** If a type inherits only from the *camp_resource* type then this attribute MAY be absent. **[MO-07]**

5.19.3 attribute_definition_links

Type: AttributeLink[]

Required: true

Mutable: false

~~This attribute contains a URI that points to the documentation for the resource type. For resource types that are defined in this Specification, the URI can point to this Specification.~~

~~6.22.2 attributeDefinitionLinks~~

~~**Type**: Link[]~~

~~**Required**: true~~

~~**Mutable**: false~~

This attribute contains an array extended of ~~Links.~~Link elements termed "AttributeLinks". Each ~~Link~~ AttributeLink in this array ~~points to~~references an ~~AttributeDefinition~~*attribute_definition* *resource*. Each of these ~~AttributeDefinition~~*attribute_definition resources* represents an attribute of the type ~~represented~~described by ~~the Type~~this *type_definition resource*.

For every attribute of the type not inherited from its super-types, there SHALL be an ~~AttributeDefinition resource Link~~AttributeLink that references the *attribute_definition resource* that defines that ~~represents the~~ attribute. **[RE-45]** In cases where a sub-type adds additional constraints to an attribute inherited from its super-types (e.g. makes an optional attribute required), a Provider SHALL include an AttributeLink that references the *attribute_defintion resource* for that attribute. **[RE-76]** For more information on the ~~AttributeDefinition~~*attribute_definition resource* see the next ~~Section~~section.

AttributeLinks are extensions of the Link attribute type with the following, additional sub-attributes:

5.19.3.1 required

Type: Boolean

Required: true

Mutable: false

The value of the `required` attribute determines if the attribute defined by the *attribute_definition resource* referenced by this AttributeLink is required for resources of the type defined by the containing *type_definition resource*. A value of "true" indicates that the referenced attribute will always be present in resources of the type defined by the containing *type_definition resource*.

5.19.3.2 mutable

Type: Boolean

Required: true

Mutable: false

The value of the `mutable` attribute determines if the attribute defined by the *attribute_definition resource* referenced by this AttributeLink is mutable for resources of the type defined by the containing *type_definition resource*. A value of "true" indicates that the referenced attribute can change during the lifetime of resources of the type defined by the containing *type_definition resource*.

5.19.3.3 consumer_mutable

Type: Boolean

Required: false

Mutable: false

~~AttributeDefinition~~The value of the `consumer_mutable` attribute determines if the attribute defined by the *attribute_definition resource* referenced by this AttributeLink is writable by Consumers for resources of the type defined by the containing *type_definition resource*. A value of "true" indicates that Consumers can change the referenced attribute for resources of the type defined by the containing *type_definition resource*. This attribute is not required in cases when the attribute defined by the *attribute_definition resource* referenced by this AttributeLink is not mutable (see above).

~~6.23~~5.20 attribute_definition Resource

An ~~AttributeDefinition~~*attribute_definition resource* represents exactly one supported attribute of one or more resource types. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "attributeDefinition",
  "description": String ?,
  ~~"representationSkew": String ?,~~
  "tags": String[] ?,
  "representation_skew": String ?,
  "documentation": URI,
  "~~attributeType~~attribute_type": String~~,~~
  ~~"required": Boolean,~~
  ~~"mutable": Boolean,~~
  ~~"consumerMutable": Boolean~~
}
```

Instances of the ~~AttributeDefinition~~*attribute_definition resource* contain the following attributes:

~~6.23.1~~5.20.1 documentation

Type: URI

Required: true

Mutable: false

~~This~~The value of the `documentation` attribute ~~contains~~is a URI that ~~points to~~references the documentation for the attribute that this resource represents. For attributes that are defined in this ~~Specification, the URI can point to~~specification, this ~~Specification~~URI references this specification.

~~6.23.2 attributeType~~

5.20.2 attribute_type

Type: String

Required: true

Mutable: false

~~This~~The value of the `attribute_type` attribute specifies the type of the attribute that this ~~resource represents. For example, "String", "Timestamp".~~described by this resource. See Section 5.2, “Attribute Types”, for a list of the values defined by this specification.

~~6.23.31.1.1.1 required~~

~~**Type**: Boolean~~

~~**Required**: true~~

~~**Mutable**: false~~

~~This attribute specifies if the attribute that this resource represents is required.~~

~~6.23.41.1.1.1 mutable~~

~~**Type**: Boolean~~

~~**Required**: true~~

~~**Mutable**: false~~

~~This attribute specifies the mutability of the attribute that this resource represents.~~

~~6.23.5 consumerMutable~~

~~**Type**: Boolean~~

~~**Required**: true~~

~~**Mutable**: false~~

~~This attribute specifies if the attribute this resource represents is writable by a CAMP client.~~

~~ParameterDefinitions~~The appearance of the square bracket symbols, "[]", appended to the value of the `attribute_type` attribute indicates that the value of the attribute that is described by this resource is an array of the specified type. For example, an `attribute_type` value of "Link[]" indicates that the value of the attribute being described by is an array of Links.

~~6.24~~5.21 parameter_definitions Resource

~~A ParameterDefinitions~~A *parameter_definitions resource* represents a collection of supported parameters for a particular resource. Multiple resources MAY reference the same ~~ParameterDefinitions Resource~~*parameter_definitions resource*. **[RE-46]** This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "~~parameterDefinitions~~parameter_definitions",
  "description": String ?,
  "tags": String[] ?,
  "~~representationSkew~~representation_skew": String ?,
  ~~"parameterDefinitionLinks": Link~~
  "parameter_definition_links": ParameterLink[]
}
```

~~Instances of the ParameterDefinitions resource~~*parameter_definitions resources* contain the following attributes:

~~6.24.1 parameterDefinitionLinks~~

5.21.1 parameter_definition_links

Type: ~~Link~~ParameterLink[]

Required: true

Mutable: ~~true~~false

~~This~~The value of the `parameter_definition_links` attribute is an array of ~~Links to ParameterDefinition resources.~~extended Link elements termed "ParameterLinks". Each ~~Link~~ParameterLink in this array refers to one *parameter_definition resource.*

ParameterLinks are extensions of the Link attribute type with the following, additional sub-attributes:

5.21.1.1 required

Type: Boolean

Required: true

Mutable: false

~~ParameterDefinition~~The value of the `required` attribute specifies whether the parameter defined by the *parameter_definition resource*~~.~~ referenced by this ParameterLink is required for HTTP POST requests on the resource that references the containing *parameter_definitions resource*. A value of "true" indicates that the referenced parameter is required for all POST requests on the resource that references the containing *parameter_definitions resource*.

5.21.1.2 default_value

Type: As defined by referenced *parameter_definition resource*.

Required: false

Mutable: false

~~ParameterDefinition~~The value of the `default_value` attribute, when present, specifies the default value for the parameter defined by the *parameter_definition resource* referenced by this ParameterLink. If the Consumer does not supply a value for the parameter defined by the *parameter_definition* resource referenced by this ParameterLink, the value of this attribute will be used. Note that the presence of the `default_value` attribute is mutually exclusive with a `required` value (see above) of "true".

~~6.25~~5.22 parameter_definition Resource

~~A ParameterDefinition~~A *parameter_definition resource* represents exactly one supported parameter of one or more resource ~~type~~types. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "~~parameterDefinition~~parameter_definition",
  "description": String ?,
  "tags": String[] ?,
  "~~representationSkew~~representation_skew": String ?,
  "~~parameterType~~parameter_type": String,
  "~~required": Boolean,~~
~~  "defaultValue~~parameter_extension_uri": String~~,~~
~~  "parameterExtensionUri": String ?~~
~~}~~ ?
}
```

~~Instances of the ParameterDefinition resource~~*parameter_definition resources* contain the following attributes:

~~6.25.1 parameterType~~

5.22.1 parameter_type

Type: String

Required: true

Mutable: false

This attribute specifies the type of the attribute that this resource represents. For example, "String", "Timestamp".

~~6.25.2 required~~

~~**Type**: Boolean~~

5.22.2 parameter_extension_uri

Type: URI

~~**Required**: true~~

~~**Mutable**: false~~

~~This attribute specifies if the parameter that this resource represents is required.~~

~~6.25.3 defaultValue~~

~~**Type**: String~~

Required: false

Mutable: false

~~This attribute specifies the default value for this parameter, when present.~~

~~6.25.4 parameterExtensionUri~~

~~**Type**: URI~~

~~**Required: false**~~

~~**Mutable**: false~~

If this parameter is handled by an extension, this attribute refers to the ~~Extension Resource~~*extension resource* that represents that ~~extension~~Extension and documents how the parameter is handled.

~~6.26~~5.23 ~~Operations~~operations Resource

An ~~Operations~~*operations* *resource* represents a collection of ~~Operation~~*operation* *resources* available on a target resource. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "operations",
  "description": String ?,
  "tags": String[] ?,
  "~~representationSkew~~representation_skew": String ?,
  "~~targetResource~~target_resource": URI,
  "~~operationLinks~~operation_links": Link[]
}
```

Instances of the ~~Operations~~*operations* *resource* contain the following attributes:

~~6.26.1 targetResource~~

5.23.1 target_resource

Type: URI

Required: true

Mutable: false

This attribute indicates the CAMP resource on which the linked operations are invoked. Linked operations are those referred to by the ~~operationLinks~~operation_links attribute. We use the term “target resource” to identify the resource referred to by this attribute.

5.23.2 operation_links

Type: Link[]

Required: true

Mutable: false

~~6.26.2 operationLinks~~

~~**Type**: Link[]~~

~~**Required**: true~~

~~**Mutable: false**~~

This attribute contains Links to the ~~Operation~~*operation* *resources* available on the target resource.

~~6.27~~5.24 ~~Operation~~operation Resource

An ~~Operation~~*operation* *resource* represents exactly one operation or action available on a target resource. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "operation",
  "description": String ?,
  "tags": String[] ?,
  "~~representationSkew~~representation_skew": String ?,
  "documentation": URI,
  "~~targetResource~~target_resource": URI,
  "parameter_definitions_uri": URI ?
}
```

In addition to the methods defined in Section ~~0~~1.1, “HTTP Method Support”, Providers SHALL support the HTTP POST method on ~~instances of~~ the ~~Operation~~*operation* *resource*. **[RE-64]**

A POST request on the ~~Operation~~*operation* *resource* invokes the ~~operation~~Operation on the target resource. The Operation MAY require content in the body of the POST, such as parameters. **[RE-47]** The response to a POST request on an ~~Operation~~*operation* *resource* SHOULD indicate what changes were made on the target resource. **[RE-48]** For asynchronous operations, the response SHOULD indicate how to track the progress of the request operation. **[RE-49]**

NOTE: For asynchronous operations, a Provider can accept a webhook URL from the Consumer as a parameter to the Operation POST request~~,~~ and notify the client at that URL upon completion of the operation. It can also allow for polling of the resource to indicate completion.

Instances of the ~~Operation~~*operation* *resource* contain the following attributes:

~~6.27.1~~5.24.1 name

Type: String

Required: true

Mutable: false

This attribute contains the name of the operation that this resource represents. For example, “deploy” or “resize”.

~~6.27.2~~5.24.2 documentation

Type: URI

Required: true

Mutable: false

This attribute contains a URI of documentation for the operation this resource represents. The documentation SHOULD describe the behavior of the operation, the form of the body expected in POST requests, and the semantics and form of the response to such requests. **[RE-50]**

~~6.27.3 targetResource~~

5.24.3 target_resource

Type: URI

Required: true

Mutable: false

This attribute indicates the CAMP resource on which the linked operation is invoked.

5.24.4 parameter_definitions_uri

Type: URI

Required: false

Mutable~~Sensors~~: false

The value of the `parameter_definitions_uri` attribute is a URI that references a *parameter_definitions resource* that contains links to the *parameter_definition resources* that describe the parameters accepted by this resource on an HTTP POST method. Each of the *parameter_definition resources* provides metadata for a parameter as described in Section 5.21, "parameter_definitions Resource".

~~6.28~~5.25 sensors Resource

A ~~Sensors~~*sensors resource* represents a collection of ~~Sensor~~*sensor resources* available on a target resource. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "sensors",
  "description": String ?,
  "tags": String[] ?,
  "~~representationSkew~~representation_skew": String ?,
  "~~targetResource~~target_resource": URI,
  "~~sensorLinks~~sensor_links": Link[]
}
```

Instances of the ~~Sensors~~*sensors resource* contain the following attributes:

~~6.28.1 targetResource~~

5.25.1 target_resource

Type: URI

Required: true

Mutable: false

This attribute indicates the CAMP resource for which the linked sensors supply runtime data. Linked sensors are those referred to by the ~~sensorLinks~~sensor_links attribute. We use the term "target resource" to identify the resource referred to by this attribute.

~~6.28.2 sensorLinks~~

~~**Type**: Link[]~~

~~**Required**: true~~

~~**Mutable**: false~~

5.25.2 sensor_links

Type: Link[]

Required: true

Mutable: false

This attribute contains Links to the ~~Sensor~~*sensor resources* available on the target resource.

~~6.29~~5.26 ~~Sensor~~sensor Resource

A ~~Sensor~~*sensor resource* represents exactly one supported sensor on one or more resources. ~~Sensor resources represent~~A *sensor resource* represents dynamic data about resources, such as metrics or state. ~~Sensor~~A *sensor resources* ~~are~~is useful for exposing data that changes rapidly, or that may need to be fetched from a secondary system. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "sensor",
  "description": String ?,
  "tags": String[] ?,
  "~~representationSkew~~representation_skew": String ?,
  "documentation": URI,
  "~~targetResource~~target_resource": URI,
  "~~sensorType~~sensor_type": String,
  "value": <~~sensorType~~sensor_type> ?,
  "timestamp": Timestamp ?,
  "~~operationsURI~~operations_uri": URI ?
}
```

Instances of the ~~Sensor~~*sensor resource* contain the following attributes:

~~6.29.1~~5.26.1 documentation

Type: URI

Required: true

Mutable: false

This attribute contains a URI that points to the documentation for the sensor this resource represents.

~~6.29.2 targetResource~~

5.26.2 target_resource

Type: URI

Required: true

Mutable: false

This attribute indicates the CAMP resource for which this ~~Sensor~~*sensor resource* supplies runtime data.

~~6.29.3 sensorType~~

5.26.3 sensor_type

Type: String

Required: true

Mutable: false

This attribute specifies the type of the data that this ~~Sensor~~*sensor resource* collects. For example, "String", "Timestamp". ~~Common Types~~Attribute types are defined in Section ~~5.1, "Common Types". TypeDefinitions~~5.2, "Attribute Types". type_definitions may also be used to specify types. See Section 5.18, "~~TypeDefinitions~~type_definitions Resource".

~~6.29.4~~5.26.4 value

Type: As defined in ~~*sensorType*~~sensor_type

Required: false

Mutable: true

Consumer-mutable~~**Required**: false~~

~~**Mutable**~~: false

This attribute contains the current or most recent available value for this sensor. It can be omitted, for example, to indicate that no current value is available; either because no data has been collected or the collected data is stale.

~~6.29.5~~5.26.5 timestamp

Type: Timestamp

Required: false

Mutable: false

This attribute contains the timestamp of the last collection or relevant activity of the sensor. When a “value” attribute is supplied, any timestamp provided in this attribute SHOULD correspond to when that value was observed. **[RE-51]**

~~6.29.6 operationsUri~~

5.26.6 operations_uri

Type: URI

Required: false

Mutable: false

This attribute contains the URI of the ~~Operations~~*operations* *resource* ~~listing the Operation resources~~. The *operations resource* lists the *operation resource* links available for ~~this~~the *sensor* *resource*.

Extensions MAY be defined to govern common sensor management operations, such as enabling, disabling, adjusting collection frequency, specifying the history of values which should be remembered, or collecting immediately. **[RE-52]**

~~7~~6 Protocol

~~7.1~~6.1 Transfer Protocol

The CAMP API is based on the Hypertext Transfer Protocol, version 1.1 [RFC2616]. ~~Each request will be authenticated using HTTP Basic Authentication [RFC2617] unless otherwise noted. Therefore, requests~~Requests sent from Consumers across unsecured networks SHOULD use the HTTPS protocol. **[PR-40]** TLS 1.1 [RFC4346] SHALL be implemented by the Provider. **[PR-41]** TLS 1.2 [RFC5246] is RECOMMENDED. **[PR-42]** When TLS is implemented, the following cipher suites are RECOMMENDED to ensure a minimum level of security and interoperability between implementations:

- TLS_RSA_WITH_AES_128_CBC_SHA (mandatory for TLS 1.1/1.2) **[PR-43]**
- TLS_RSA_WITH_AES_256_CBC_SHA256 (addresses 112-bit security strength requirements) **[PR-44]**

Note: For interoperability reasons, Providers are encouraged to support a common authentication scheme in order to simplify the implementation of client tools that are intended to work with multiple Providers. The *platform_endpoint resource* `auth_scheme` attribute (see Section 1.1.1, "auth_scheme") makes available authentication schemes discoverable by unauthenticated clients.

~~7.2~~6.2 URI Space

The resources in the system are identified by URIs. Dereferencing the URI will yield a representation of the resource containing resource attributes and links to associated resources.

Note: Consumers ~~SHALL NOT make~~are cautioned against making assumptions about the layout of the URIs or the structure of the URIs of the resources. ~~[PR-45]~~

~~7.3~~6.3 Media Types

~~7.3.1~~6.3.1 Required Formats

In this specification~~,~~ resource representations~~, request bodies, and error response messages~~ are encoded in JSON, as specified in [RFC4627]. The media-type associated with CAMP JSON resource ~~and error response message~~ representations is "application/json".

Providers SHALL provide representations of all available resources in JSON. **[PR-01]**

~~7.3.1.1~~6.3.1.1 Duplicate Keys in JSON Objects

~~Duplicate keys in~~CAMP defined JSON objects ~~are allowed by **[RFC4627]**. This specification prohibits duplicate keys for interoperability reasons. Both~~ do not contain duplicate keys. Consumers and Providers SHALL NOT transmit JSON objects that contain duplicate keys. **[PR-02]**

If a Consumer sends a Provider a request containing duplicate keys in a JSON object, the Provider SHOULD reject the request by sending back a '400 Bad Request' status code. **[PR-03]** If a Provider sends a Consumer a response containing duplicate keys in a JSON object, the Consumer SHOULD raise an error to the user indicating the response from the server was malformed. **[PR-04]**

Note: Duplicate keys in JSON objects are allowed by JSON [RFC4627]. This specification prohibits duplicate keys for interoperability reasons.

~~7.3.2~~6.3.2 Supported Formats

If Supported Formats besides JSON are referenced in the ~~supportedFormatsUri~~ attribute of the ~~Platform~~*platform* *resource*, then resource representations~~, request bodies, and error response messages~~ are allowed in the Supported Formats.

~~Supported Formats SHALL be applied uniformly for all resources defined by this specification.~~For each Supported Format, Consumers MAY request any resource from the Provider in that format. **[PR-45]** Providers SHALL respond in the requested Supported Format. **[PR-05]**

A client can request any Supported Format using HTTP content negotiation.

~~7.4~~6.4 Request Headers

This API does not impose any requirements on clients' use of HTTP headers. All PUT requests that update a resource SHOULD contain the If-Match header field with a single entity tag value. **[PR-06]** If the If-Match header field value in the request does not match the one on the server-side, the Provider SHALL send back a '412 Precondition Failed' status code. **[PR-07]**

~~7.5~~6.5 Request Parameters

To retrieve a subset of the attributes in a resource, the Consumer MAY use the '~~SelectAttr'~~select_attr' request parameter in conjunction with the HTTP GET method. **[PR-08]** A Provider SHALL return only the attributes of the queried resource that match the attribute names passed as arguments of 'select_attr'. **[PR-47]**

Format	Description	Example
?~~SelectAttr~~select_attr=attr1,attr2,…	Comma (",") separated attribute names of the resource to return. If an attribute is not part of the resource, an HTTP 4XX status code SHALL be returned. **[PR-09]**	~~Assembly132?SelectAttr~~assembly132?select_attr =name,description,tags Would access only "name", "description", "tags" attributes of ~~Assembly132~~assembly132.

Table 6-1: Request Parameters

The "~~SelectAttr~~select_attr" query parameter MAY appear more than once (separated by an "&"). **[PR-10]** The Consumer SHALL URL encode the request parameters. **[PR-11]**

When one or more request parameters are specified for a PUT request, a Consumer SHALL NOT include attributes in the request entity body that are not specified in the request parameter. **[PR-12]** Upon receiving such a request the Provider SHALL respond with a 400 status code. **[PR-13]**

~~7.6~~6.6 POST Body Parameters

Parameters MAY be included when performing a POST request on any resource with a ~~parameterDefinitionsUri~~`parameter_definitions_uri` attribute defined. **[PR-14]** Supported parameters are defined by the ~~parameterDefinitions~~parameter_definitions resource referenced by the ~~parameterDefinitionsUri~~parameter_definitions_uri attribute of the resource handling the POST request.

Example of a POST Parameter:

```
POST /<assembly-template-resource-url> HTTP/1.1
Host: example.org
Content-Type: application/json
Content-Length: ...
```

```
{ "EXAMPLE:someParameter": "bar" }

HTTP/1.1 201 Created
Location: http://example.org/paas/assembly/1
Content-Type: ...
Content-Length: ...
```

~~7.6.1~~6.6.1 Parameter Handling

Parameters allow customizing ~~Resources~~resources upon creation. Parameters MAY have the same name as an ~~Attribute on~~attribute of the ~~Resource~~resource. **[PR-15]** In such cases the Provider SHOULD set ~~the Attribute~~that attribute to take the value of the ~~Parameter~~parameter OR clearly document alternate behavior. **[PR-16]** The ~~parameterExtensionUri~~`parameter_extension_uri` MAY be used to reference the ~~Extension~~extension which documents how the parameter is handled. **[PR-17]**

If a POST request body does not contain a value for a required parameter, a “400 Bad Request” response SHALL be returned~~ with an Error Response Message Resource.~~. **[PR-18]**

If a POST request body does not contain an acceptable value for a parameter, a “400 Bad Request” response SHALL be returned~~ with an Error Response Message Resource.~~. **[PR-19]**

~~7.7~~6.7 Response Headers

Responses returned by the Provider make standard use of HTTP headers. All HTTP responses that return representation of a resource SHOULD use strong Etag response header field indicating the current value of the entity tag for the resource. **[PR-20]**

~~7.8~~6.8 HTTP Status Codes

The API returns standard HTTP response codes.

~~7.9~~6.9 Mutability of Resource Attributes

Consumers SHALL NOT send a request that changes the value of a resource attribute that is declared with a constraint of 'Mutable=false' or 'Consumer-mutable=false'. **[PR-21]** On receiving such a request the Provider SHALL generate an HTTP response with 403 HTTP status code. **[PR-22]**

~~7.10~~6.10 Updating Resources

Attributes of the resources defined with “~~consumerMutable~~consumer_mutable: true” can be modified by Consumers in two ways. Consumers MAY use the HTTP PUT method to replace the representation of a resource, in its entirety, with a new representation that adds, omits or replaces the values for some of the attributes. **[PR-23]** Alternatively, Consumers MAY use the HTTP PATCH [HTTP PATCH] method and the “application/json-patch+json” media type ~~[JSON Patch~~[RFC6902] to add, delete, or replace specific attributes. **[PR-24]**

~~7.10.1~~6.10.1 Updating with PUT

HTTP PUT requests are requests for complete replacement of the resource identified by the request URL.

On successfully processing an HTTP PUT request a Provider SHALL update all the Consumer-mutable attributes of the target resource, and only these, with the values of the matching attributes in the request. **[PR-48]** If a resource attribute is present on a resource and if an HTTP PUT request omits that attribute, it SHOULD be treated by the Provider as a request to delete the attribute. **[PR-25]**

~~7.10.2~~6.10.2 Updating with JSON Patch

JSON Patch [~~JSON Patch~~RFC6902] defines a JSON document structure for expressing a sequence of operations to apply to a JSON document, suitable for use with the HTTP PATCH method. The "application/json-patch+json" media type is used to identify such patch documents.

Providers SHALL support the HTTP PATCH method in conjunction with the "application/json-patch+json" media type with the following, additional provisions with respect to the operations defined in ~~section~~Section 4 of the JSON Patch specification ~~[JSON Patch]:~~: **[PR-26]**

- Providers SHALL support the 'add', 'remove', and 'replace' operations. **[PR-27]**
- Providers MAY support the 'move', 'copy, and 'test' operations. **[PR-28]**

~~7.11~~6.11 ~~Registering~~Deploying an Application

~~As indicated in Section 3.2,"Creating an Assembly Template from a PDP", registering~~Deploying an application ~~moves it to~~uploads the ~~deployed state.~~ artifacts and metadata that make up the application, allocates the necessary resources and services, and, in the successful case, creates a running application (represented by an *assembly resource*).

There are two ways to ~~register a~~ deploy an application using a PDP: by POSTing the entire PDP to the ~~Platform~~*assemblies resource* (by value) or by POSTing the URI of the PDP to the ~~Platform~~*assemblies resource* (by reference). Similarly, there are two ways to ~~register~~deploy an application using a ~~DP~~Plan: by POSTing the entire ~~DP~~Plan file to the ~~Platform~~*assemblies resource* (by value) or by POSTing the URI of the ~~DP~~Plan file or *plan resource* to the ~~Platform~~*assemblies resource* (by reference). All of these methods are described below. ~~A Provider supports registering a PDP using~~Providers SHALL support PDPs that use either the ZIP [ZIP], TAR [TAR], ~~and~~or GZIP [~~GZIP~~RFC1952] compressed TAR ~~format.~~formats. **[RMR-13]**

6.11.1 Deploying an Application by Reference

To deploy an application by reference, a Consumer sends a reference to either a PDP, Plan file, or *plan resource* in a POST request to the *assemblies resource*. Providers SHALL support the deployment of applications via HTTP POST requests on the *assemblies resource* as described in this section. **[PR-49]** The entity body of the request contains a URI that identifies the PDP, Plan file, or *plan resource* that is being deployed. If the URI that identifies the PDP, Plan file, or *plan resource* is a relative URI, its base URI is that of the *platform resource*. When deploying a PDP the JSON serialization of the HTTP request entity body is:

```
{"pdp_uri": "<uri-of-the-pdp>"}
```

When deploying a Plan file or *plan_resource* the JSON serialization of the HTTP request entity body is:

```
{"plan_uri": "<uri-of-the-plan>"}
```

Where, the value of `pdp_uri` is the URI of the PDP to be deployed and the value of `plan_uri` is the URI of the Plan to be deployed. To support the deployment of applications via a reference to either a PDP, Plan file, or *plan resource*, Providers SHALL accept the "application/json" media type. **[PR-68]** The JSON object MAY contain additional name-value pairs that are not defined in this specification. **[PR-33]**

Note that the value of `plan_uri` can refer to either a Plan file or a *plan resource*. A referenced *plan resource* can exist either on the same CAMP Platform as the target *assemblies resource*, or on some other CAMP Platform. In the case where the referenced *plan resource* exists within the same Platform as the target *assemblies resource*, Consumers are advised to use a relative URL for the *plan resource* reference to help Providers identify the *plan resource* as local.

An example HTTP request-response is as follows:

```
POST /paas/assemblies HTTP/1.1
Host: example.org
Content-Type: application/json
Content-Length: 46
…

{"pdp_uri": "/paas/pdp/1",
 "description": "Mike’s other Drupal instance"}

HTTP/1.1 201 Created
Location: http://example.org/paas/assembly/11
…
```

Note the inclusion of `description` parameter/attribute in the body of the above POST request.

On successfully processing the request the Provider SHALL create an *assembly resource* and return a 201 Created status code in the HTTP response. **[PR-50]** The Provider SHALL include the *Location* header in the HTTP response and the value of this header SHALL reference the newly created *assembly resource*. **[PR-51]** The Provider SHALL update the `assembly_links` attribute of the *assemblies resource* to include a reference to the newly created *assembly resource*. **[PR-52]**

6.11.2 Deploying an Application by Value

To deploy an application by value, a Consumer transmits the contents of either a PDP or a Plan file in a POST request to the *assemblies resource*. The Consumer can do this in two ways, either by including the data as a part in a multipart MIME message or by simply including the data in the entity body of the HTTP request.

To support the deployment of applications using a PDP, Providers SHALL accept the media types associated with the various formats ~~SHALL be~~ as follows:

- ZIP: "application/x-zip" **[PR-29]**
- TAR: "application/x-tar" **[PR-30]**
- GZIP compressed TAR: "application/x-tgz" **[PR-31]**

~~Since~~To support the ~~DP is a YAML file, when registering an application~~deployment of applications using a ~~DP the associated media type~~Plan file, Providers SHALL ~~be~~ accept the use of the "application/x-yaml~~".~~" media type. **[PR-32]**

~~7.11.1 Registering an Application by Reference~~

~~To register an application by~~ On successfully processing the request the Provider SHALL create an *assembly resource* and return a 201 *Created* status code in the HTTP response. **[PR-53]** The Provider SHALL include the *Location* header in the HTTP response and the value of this header SHALL reference~~,~~ ~~a~~ the newly created *assembly resource*. **[PR-54]** The Provider SHALL update the `assembly_links` attribute of the *assemblies resource* to include a reference to the newly created *assembly resource*. **[PR-55]**

For large PDPs, the Consumer ~~sends a~~ can use existing HTTP facilities like chunked transfer encoding.

6.11.2.1 Deploying an Application by Value Using MIME

Providers SHALL support the deployment of applications via HTTP POST ~~HTTP request to the Platform URL.~~requests on the *assemblies resource* as described in this section. **[PR-74]** The entity body of the request is a multipart MIME message that contains the ~~URI that identifies the~~ PDP or the ~~DP~~Plan file that is being ~~registered. If the URI~~deployed. The value of the HTTP Content-Type header is “multipart/form-data” [RFC2388]. The MIME part that ~~identifies the~~ contains the file data has the value of the `name` parameter of its `Content-Disposition` header set to “pdp_file”, if a PDP is being deployed, or ~~the DP is a relative URI, its base URI is the Platform URI. When registering~~“plan_file”, if a ~~PDP the JSON serialization of the HTTP~~ Plan file is being deployed. CAMP parameters can be included in the request

~~entity body is:~~as additional MIME parts using the value of the `name` parameter of the `Content-Disposition` header to indicate the CAMP parameter being included.

```
~~{"pdpUri" : "<uri-of-the-pdp>"}~~
```

~~When registering a DP the JSON serialization of the HTTP request entity body is:~~

```
~~{"dpUri" : "<uri-of-the-dp>"}~~
```

~~Where, the value of *pdpUri* is the URI of the PDP to be registered and the value of *dpUri* is the URI of the DP to be registered. The JSON object MAY contain additional name-value pairs that are not defined in this specification.~~ ~~**[PR-33]** On successful registration of the application, the Provider creates an AssemblyTemplate resource and sends a 201 Created HTTP status code with the *Location* header in the HTTP response. The *Location* header points to the newly created AssemblyTemplate resource. The Provider also updates the *assemblyTemplates* attribute of the Platform resource to include a reference to the newly created AssemblyTemplate. When a PDP is used to register the application, the Provider SHALL include the pdpUri attribute, which identifies the PDP from which the template was created, in the newly created AssemblyTemplate resource.~~ ~~**[PR-34]** When a DP is used to register the application, the Provider SHALL include the dpUri attribute, which identifies the DP from which the template was created, in the newly created AssemblyTemplate resource.~~ ~~**[PR-35]**~~

~~An example HTTP request-response is as follows:~~

```
~~POST /<platform-url> HTTP/1.1~~
~~Host: example.org~~
~~Content-Type: application/json~~
~~Content-Length: ...~~

~~{"pdpUri": "/paas/pdp/1"}~~

~~HTTP/1.1 201 Created~~
~~Location: http://example.org/paas/asm_template/1~~
~~Content-Type: ...~~
~~Content-Length: ...~~

~~...~~
```

~~7.11.2 Registering an Application by Value~~

~~To register an application by value, a Consumer sends a POST HTTP request to the Platform URL. The entity body of the request contains the PDP or the DP that is being registered. On successful registration of the PDP, the Provider creates an AssemblyTemplate resource and sends a 201 Created HTTP status code with the *Location* header in the HTTP response. The *Location* header points to the newly created AssemblyTemplate resource. The Provider also updates the *assemblyTemplates* attribute of the Platform resource to include a reference to the newly created AssemblyTemplate. When a PDP is used to register the application, the Provider SHALL include the pdpUri attribute, which identifies the PDP from which the template was created, in the newly created AssemblyTemplate resource.~~ ~~**[PR-36]** When a DP is used to register the application, the Provider SHALL include the dpUri attribute, which identifies the DP from which the template was created, in the newly created AssemblyTemplate resource.~~ ~~**[PR-37]**~~

~~For example, the *pdpUri* can point to an entry in the repository used by the Platform to manage its deployment artifacts.~~

An example HTTP request-response is as follows:

```
POST ~~/<platform-url>~~/paas/assemblies HTTP/1.1
Host: example.org
Content-Length: 9768506
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryU6AnBoMx
…

------WebKitFormBoundaryU6AnBoMx
Content-Disposition: form-data; name="pdp_file"; filename="Mike_Drupal2.pdp"
Content-Type: application/x-zip
~~Transfer-Encoding: chunked~~
~~Content-Transfer-Encoding:~~
… binary
~~...~~

~~... binary PDP ZIP octets ...~~ octets …
------WebKitFormBoundaryU6AnBoMx
Content-Disposition: form-data; name="description"

Mike's other Drupal instance
------WebKitFormBoundaryU6AnBoMx--

HTTP/1.1 201 Created
Location: http://example.org/paas/~~asm_template~~assembly/12
Content-Type: application/json

~~...~~…
```

Note the inclusion of the description parameter as a separate MIME part.

Deploying~~For large PDPs, the Consumer can use existing HTTP facilities like chunked transfer encoding. Please note that, unlike registration by reference, it is not possible to include additional parameters when registering by value.~~

~~7.12~~6.11.2.2 ~~Instantiating~~ an Application by Value Without MIME

~~Once the application is in~~ Providers SHALL support the ~~deployed state, a Consumer can instantiate the application by sending a~~ deployment of applications via HTTP POST ~~HTTP request to the corresponding AssemblyTemplate URL. The~~ requests on the *assemblies resource* in which the entity body of the request ~~can be empty. Interpretation of a non-empty entity body of the request is implementation-dependent. On success~~contains the PDP or the Plan file that is being deployed. **[PR-60]**

An example HTTP request-response is as follows:

```
~~server creates an Assembly resource and sends a~~ POST /paas/assemblies HTTP/1.1
Host: example.org
Content-Length: 976361
Content-Type: application/x-zip
Content-Transfer-Encoding: binary
…

… binary PDP ZIP octets …

HTTP/1.1 201 Created~~ HTTP status code with the~~
Location: http://example.org/paas/assembly/12
…
```

Note that it is not possible to include additional parameters when using this mechanism to deploy by an application.

6.12 Registering a Plan

As described in Section 0, "Deployment", CAMP implementations can choose to support the expression of Plans as CAMP resources. This feature allows Consumers to register an application (upload the artifacts and metadata, validate the Plan, resolve dependencies, etc.) without creating a running instance of that application. Consumers can later instantiate an application from the *plan resource*.

Similarly to deploying an application, Plans can be registered using either a PDP or a Plan file. The PDP or Plan file can be supplied by value or by reference.

The archive formats available to the PDP are identical to those used to deploy an application.

6.12.1 Registering a Plan by Reference

To register a Plan by reference, a Consumer sends a reference to either a PDP or a Plan file in a POST request to the plans resource. Providers that support the *plans resource* and *plan resources* SHALL support the registration of Plans via an HTTP POST request on the *plans resource* as described in this section. **[PR-56]** The entity body of the request contains a URI that identifies the PDP or the Plan file that is being registered. If the URI that identifies the PDP or the Plan file is a relative URI, its base URI is that of the *platform resource*. When registering a PDP the JSON serialization of the HTTP request entity body is:

```
{"pdp_uri": "<uri-of-the-pdp>"}
```

When registering a Plan file the JSON serialization of the HTTP request entity body is:

```
{"plan_uri": "<uri-of-the-plan>"}
```

Where, the value of `pdp_uri` is the URI of the PDP to be registered and the value of `plan_uri` is the URI of the Plan to be registered. To support the registration of Plans via a reference to either a PDP or a Plan file, Providers SHALL accept the "application/json" media type. **[PR-69]** The JSON object MAY contain additional name-value pairs that are not defined in this specification. **[PR-46]**

An example HTTP request-response is as follows:

```
POST /paas/plans HTTP/1.1
Host: example.org
Content-Type: application/json
Content-Length: 46
…

{"pdp_uri": "/paas/pdp/1",
 "description": "Mike’s other Drupal instance"}

HTTP/1.1 201 Created
Location: http://example.org/paas/plan/9
…
```

Note the inclusion of `description` parameter/attribute in the body of the above POST request.

On successfully processing the request the Provider SHALL create a *plan resource* and return a 201 Created status code in the HTTP response. **[PR-57]** The Provider SHALL include the *Location* header in the HTTP response~~. The *Location*~~ and the value of this header ~~points to~~ SHALL reference the newly created ~~*Assembly*~~*plan resource*. **[PR-58]** The ~~server also updates~~Provider SHALL update the ~~*AssemblyInstances*~~`plan_links` attribute of the ~~Platform~~*plans resource* to include a reference to the newly created *plan resource*. **[PR-59]**

6.12.2 Registering a Plan by Value

To register a Plan by value, a Consumer transmits the contents of either a PDP or a Pan file in a POST request to the *plans resource*. The Consumer can do this in two ways, either by including the data as a part in a multipart MIME message or by simply including the data in the entity body of the HTTP request.

To support the registration of Plans using a PDP, Providers SHALL accept the media types associated with the various formats as follows:

- ZIP: "application/x-zip" **[PR-70]**

- TAR: "application/x-tar" **[PR-71]**
- GZIP compressed TAR: "application/x-tgz" **[PR-72]**

To support the registration of Plans using a Plan file, Providers SHALL accept the use of the "application/x-yaml" media type. **[PR-73]**

On successfully processing the request the Provider SHALL create a *plan resource* and return a 201 Created status code in the HTTP response. **[PR-62]** The Provider SHALL include the *Location* header in the HTTP response and the value of this header SHALL reference the newly created *plan resource*. **[PR-63]** The Provider SHALL update the `plan_links` attribute of the *plans resource* to include a reference to the newly created *plan resource*. **[PR-64]**

For large PDPs, the Consumer can use existing HTTP facilities like chunked transfer encoding.~~assembly~~

6.12.2.1 Registering a Plan by Value Using MIME

Providers that support the plans resource and plan resources SHALL support the registration of Plans via HTTP POST requests on the plans resource as described in this section. **[PR-75]** The entity body of the request is a multipart MIME message that contains the PDP or the Plan file that is being registered. The value of the HTTP `Content-Type` header is "multipart/form-data" [RFC2388]. The MIME part that contains the file data has the value of the `name` parameter of its `Content-Disposition` header set to "pdp_file", if a PDP is being registered, or "plan_file", if a Plan file is being registered. CAMP parameters can be included in the request as additional MIME parts using the value of the `name` parameter of the `Content-Disposition` header to indicate the CAMP parameter being included.

An example HTTP request-response is as follows:

```
POST /paas/~~asm_template/1~~plans HTTP/1.1
Host: example.org

Content-Length: 1685
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryE733b300
…

------WebKitFormBoundaryE733b300
Content-Disposition: form-data; name="plan_file"; filename="Mike_Drupal.yaml"
Content-Type: application/x-yaml

… unicode characters …
------WebKitFormBoundaryE733b300
Content-Disposition: form-data; name="description"

Mike's other Drupal instance
------WebKitFormBoundaryE733b300--

HTTP/1.1 201 Created
Location: http://example.org/paas/~~assembly/1~~
~~Content-Type: ...~~
~~Content-Length: ...~~

~~...~~plan/9
…
```

~~7.13 Suspending and Resuming an Application~~

~~To suspend, or resume an application, a client sends a POST HTTP request to~~Note the ~~assembly~~inclusion of the description parameter as a separate MIME part.

6.12.2.2 Registering a Plan by Value Without MIME

Providers that support the *plans* resource ~~URL. The~~ and *plan resources* SHALL support the registration of Plans via HTTP POST requests on the *plans resource* in which the entity body of the request contains the

~~value of the new state for the application. The JSON serialization of the HTTP request entity body~~ PDP or the Plan file that is~~:~~ being registered. **[PR-61]**

```
~~{"new_state" : "<new-state-value>"}~~
```

~~Where, *new_state* specifies the new desired value for the application state. This specification defines two such values: "suspend", and "resume," whose semantics are as defined in Section 0. A Provider MAY have additional state values that it allows.~~ ~~**[PR-38]**~~ ~~The JSON object MAY contain additional name-value pairs that are not defined in this specification.~~ ~~**[PR-39]**~~ ~~An implementation can define additional state values or name-value pairs, to allow clients to specify the scale at which the application is suspended and resumed.~~

An example HTTP request-response is as follows:

```
POST ~~/<assembly-resource-url>~~/paas/plans HTTP/1.1
Host: example.org
Content-Length: 1465
Content-Type: application/~~json~~
~~Content-Length: ...~~

~~{"new_state": "suspend"}~~

x-yaml
…

… unicode characters …

HTTP/1.1 ~~200 OK~~201 Created
Location: http://example.org/paas/plan/9
…
```

~~Deleting~~Note that it is not possible to include additional parameters when using this mechanism to register a Plan.

~~7.14~~6.13 Stopping an Application Instance ~~and a Deployed Application~~

~~To delete~~One way of stopping an application instance ~~(~~is to send an ~~Assembly), a client sends a~~HTTP DELETE ~~HTTP~~ request to the ~~Assembly resource~~ URL~~. Similarly, to delete a deployed application, a client sends a DELETE HTTP request to~~ of the ~~Assembly template URL~~corresponding *assembly resource*.

~~8~~7 Extensions

Features provided by this specification can be extended to provide additional information and functionality. ~~Using Requirements and Capabilities is RECOMMENDED instead of Extensions, if possible.~~ ~~[EX-01]~~ Extensions MAY be added by registering the new functionality in the ~~Extensions~~*extensions* *resource*. **[EX-02]** Extensions SHALL NOT change or remove any features or functionality of this specification. **[EX-03]** Each ~~Extension~~extension SHALL satisfy all the criteria in ~~the Conformance section~~Section 8, "Conformance", and SHALL NOT contradict any normative statements in this document. **[EX-04]** The following extensions are allowed:

Category	**Functionality**	**Description**
API Extension	New HTTP Request Verbs	Support for additional HTTP Request Verbs that are not used by this specification, such as HEAD.
API Extension	HTTP Header Handlers	Processing of specific HTTP headers provided by clients. For example, an API Extension may require an authentication token header.
API Extension	New Resources	Addition of new ~~resources~~ resource types that MAY handle HTTP requests such as POST or PUT to create ~~instantiations~~ new resources of ~~a new resource~~ this type.
API Extension	New Resource Methods	Allow the creation of new methods or actions that may cause different sequences of state changes than happen by default.
PDP Extension	New Metadata in the PDP	Additional metadata provided in the PDP to allow for more sophisticated handling of the bundled data.
Resource Extension	New Resource Types	Addition of new resource types.
Resource Extension	New Resource Attributes	Addition of new attributes to existing resources.
Resource Extension	New States in any Application Lifecycle	Addition of new application states, such as an intermediate state between the states defined by the specification.

Table 7-1: Extension Categories and Functionality

~~8.1~~7.1 Unique Name Requirement

Entities

- Resources
- Attributes
- Methods
- PDP Metadata Keys

Table 7-2: Entities

Entities are enumerated in Table 7-2. The Extension Developer SHALL use a unique name for new ~~Entities~~entities within an existing namespace. **[EX-05]** Entities added by an ~~Extension~~extension SHALL NOT interfere with names of existing entities, including any added by another ~~Extension~~extension. **[EX-06]**

NOTE: Each resource has its own namespace. It is acceptable to create a resource named *example.org:Foo*, and another resource named *example.org:Bar*, where both resources have an attribute named *fooBar*.

The use of your registered ICAAN Internet domain name followed by a colon (":") character as a prefix to all your entity names is RECOMMENDED to comply with these requirements. **[EX-07]**

Example: New Attribute "foo" added by Example Organization

```
example.org:foo
```

Example: New Attribute "foo" added by Example Inc.

```
EXAMPLE-INC:foo
```

Extension Category	New Entity	Exception
API Extension	Adding HTTP Request Verbs	Unique name not required for HTTP verbs
API Extension	Adding HTTP Header Handlers	Unique name not required for HTTP headers

Table 7-3: Unique Name Exceptions

A unique name is not required for entities listed in Table 7-3.

NOTE: RFC-3986 identifies Unreserved Characters that may be used in a URI without any encoding. Percent-Encoding allows any character to be represented in a URI. Special characters such as ":" and "." have specific meanings in scripting languages such as JavaScript. Special characters must be properly escaped in order to use them as part of a name string. Your data serialization format may not escape all problematic characters, so you may need to add logic to your clients to escape special characters to enable interaction with an Extension.

~~8.2~~7.2 ~~Extensions~~extensions Resource

The ~~Extensions~~*extensions resource* contains an array of Links to ~~Extension~~*extension resources*. It allows the identification of ~~Extensions~~extensions. The ~~Extensions~~*extensions resource* is represented as:

```
{
  "uri": URI,
  "name": String,
  "type": "~~types~~extensions",
  "description": String~~, ?~~ ?,
  "tags": String~~[], ?~~[] ?,
  "~~representationSkew~~representation_skew": String ?,
  "~~extensionLinks~~extension_links": Link[]
}
```

~~Instances of the Extensions contain~~ The *extensions resource* contains the following attribute:

7.2.1 extension_links

Type: Link[]

Required: true

Mutable: false

~~8.2.1 extensionLinks~~

~~**Type**: Link[]~~

~~**Required**: true~~

~~**Mutable**: false~~

This attribute contains Links to ~~Extension~~*extension* *resources* that contain information about ~~Extensions~~the extensions available on the Platform. For every ~~Extension~~extension available, there SHALL be an ~~Extension~~*extension* *resource* Link that represents the ~~Extension~~extension. **[EX-08]** The ~~Platform~~*platform* *resource* SHALL provide a Link to the ~~Extensions~~*extensions* *resource* in the required attribute named ~~extensionsUri.~~`extensions_uri`. **[EX-09]**

Example of an ~~extensionLinks~~`extension_links` value:

```
[
  {
~~    "targetName~~
    "target_name" : "EXAMPLE:Auth",

    "href": "http://example.org/paas1/extension/1"
~~  },~~
~~  {~~
~~    "targetName~~
  },
  {
    "target_name" : "EXAMPLE:PDPforFooLang",

    "href" : "http://example.org/paas1/extension/2"
  }
  …
]
~~  }~~
~~…~~
~~]~~
```

~~8.3~~7.3 ~~Extension~~extension Resource

An ~~Extension~~*extension* *resource* represents new functionality added to the Platform. This resource has the following, general representation:

```
{
  "uri": URI,
  "name": String,
  "type": "extension",
  "description": String~~,~~
 ?,
  "tags": String[] ?,
  "representation_skew": String ?,
  "version": String,
  "documentation": URI ?
}
```

The ~~Extension~~*extension* *resource* contains the following attributes:

~~8.3.1~~7.3.1 version

Type: String

Required: true

Mutable: false

This attribute contains a string identifier of the version of this ~~Extension~~extension.

~~8.3.2~~7.3.2 documentation

Type: ~~Link~~URI

Required: false

Mutable: false

This attribute is a ~~Link to~~URI that references a human readable document that describes the ~~Extension in depth~~extension.

~~8.4~~7.4 Extending Existing Resources

New attributes MAY be added to an existing resource using an ~~Extension~~*extension resource* if the Unique Name Requirement in ~~0~~7.1 is met. **[EX-10]** A new resource type is not required in order to add new attributes.

Example of an ~~Extended Extension Resource~~extended *extension resource*:

```
{
  "uri": URI,
  "name": String,
  "type": "extension",
  "description": String,
  "version": String,
  "documentation": URI ?,
  "acme.com:foo": String ?
}
```

Note that in the above example, the new attribute ~~"~~acme.com:foo~~"~~ was added, and the type attribute remained set to the original value "extension".

~~9~~8 Conformance

There are three conformance targets defined in this specification:

- CAMP Provider
- CAMP Consumer
- Platform Deployment Package
- Plan

~~9.1~~8.1 CAMP Provider

An implementation claiming to conform to the requirements of a CAMP Provider defined in this specification SHALL comply with all of the CAMP Provider or Provider mandatory ~~statements~~requirements listed in this specification, as summarized in Appendix C.1, "Mandatory Statements~~", related to CAMP Provider or Provider.~~".

~~9.2~~8.2 CAMP Consumer

An implementation claiming to conform to the requirements of a CAMP Consumer defined in this specification SHALL comply with all of the CAMP Consumer or Consumer mandatory ~~statements~~requirements listed in this specification, as summarized in Appendix C.1, "Mandatory Statements~~", related to CAMP Consumer or Consumer.~~".

~~9.3~~8.3 Platform Deployment Package

For a document to be a valid PDP, it SHALL comply with all of the PDP mandatory ~~statements~~requirements listed in this specification, as summarized in Appendix C.1, "Mandatory Statements~~", related~~".

8.4 Plan

For a document to ~~the PDP.~~be a valid Plan, it SHALL comply with all of the Plan mandatory requirements listed in this specification, as summarized in Appendix C.1, "Mandatory Statements".

Appendix A. Acknowledgments

This section is informative. The following individuals have participated in the creation of this specification and are gratefully acknowledged:

Participants:

Roshan Agrawal	Rackspace Hosting, Inc.
Michael Behrens	US Department of Defense (DoD)
Bhaskar Reddy Byreddy	Software AG, Inc.
Mark Carlson	Oracle
Martin Chapman	Oracle
Francesco D'Andria	Cloud4SOA
Jacques Durand	Fujitsu Limited
Panagiotis Gouvas	Cloud4SOA
Keith Grange	JumpSoft
Alex Heneveld	Cloudsoft Corporation Limited
Gershon Janssen	Individual Member
David Jilk	Standing Cloud, Inc.
Duncan Johnston-Watt	Cloudsoft Corporation Limited
Anish Karmarkar	Oracle
Tobias Kunze	Red Hat
Eugene Luster	US Department of Defense (DoD)
Ashok Malhotra	Oracle
Alex McDonald	NetApp
Rich Miller	Cloudsoft Corporation Limited
Jeff Mischkinsky	Oracle
Adrian Otto	Rackspace Hosting, Inc.
Derek Palma	Vnomic
Gilbert Pilz	Oracle
Krishna Raman	Red Hat
Tom Rutt	Fujitsu Limited
Zhexuan Song	Huawei Technologies Co., Ltd.
Charles Tupitza	JumpSoft
Jeffrey West	Oracle
Prasad Yendluri	Software AG, Inc.

Appendix B. Glossary

Application – a set of components that act together to provide useful functions and are typically exposed as a service to Application end-users. ~~An application is represented by different resources (e.g. Assembly Template, Assembly) throughout its lifecycle.~~

~~**Application Component** –~~**Artifact -** a ~~collection~~static element of ~~code and/or, resources (optionally accompanied by metadata)~~an application that either provides a set of related services ~~or~~and functionality or contains a set of related information. Code examples include Ruby gems, Java libraries, and PHP modules. Examples of resources include data sets, identity sets (i.e. collections of user account and attribute information), and collections of graphical images.

~~**Application**~~ **Component** ~~**Capability**~~ – a ~~management resource~~dynamic element of an application that ~~represents an Application Component's capabilities.~~

~~**Application Component Requirement** –~~provides a ~~management resource that represents a requirement on an Application Component, expressed with attributes that may have value ranges.~~

~~**Application Component Template** - a management resource that represents an unrealized Application Component~~set of related services and ~~includes a reference to the executable code as well as metadata for configuring the Application Component~~functionality. Examples include Ruby processes, Spring web applications, and ~~referencing its platform and other~~ database instances~~components~~.

Application Development Environment (ADE) – a developer tool used to create an application (can be an offline tool installed locally or part of the platform offering itself).

Assembly – a management resource that represents a running application.

~~**Assembly Template** - a management resource that represents an unrealized Assembly and includes a reference to the Application Component Templates used within the Application as well as metadata for configuring the Application Component and referencing its platform and other components.~~

Deploy – the ~~step~~act of ~~creating one~~using a PDP or ~~more management resources~~Plan to create a running application (represented by an *assembly resource*) on the platform~~. Deployment can be done through the API for individual management resources (i.e via a POST to a URI), or can be done as part of the import of a Platform Deployment Package.~~

Extension - a systematic representation of additional features and functionality added by an Extension Developer.

~~**Deployment**~~ **Plan** - packaging management meta-data that provides a description of the artifacts that make up an application, the services that are required to execute or utilize those artifacts, and the relationship of the artifacts to those services~~.. Deployment Plans are an essential a required part of a Platform Deployment Package~~.

~~**Extension** - a systematic representation of additional features and functionality added by an Extension Developer.~~

Platform – The collection of management resources that constitute the consumer visible view of the Platform as a Service offering. The Platform management resource is an aggregation and discovery point for all the Applications and their dependencies currently deployed and running.

Platform as a Service (PaaS) - A type of cloud computing in which the service provider offers customers/consumers access to one or more instances of a running application computing platform or application service stack.

~~**Platform Component** – a management resource that represents an application's use of a realized and running Platform Component.~~

~~**Platform Component Capability** – a management resource that represents a Platform Component's capabilities.~~

~~**Platform Component Requirement** – a management resource that represents a requirement on a Platform Component, expressed with attributes that may have value ranges.~~

~~**Platform Component Template** - a management resource that represents an unrealized Platform Component and includes references to metadata for configuring an instance of that Platform Component.~~

Platform Deployment Package (PDP) - an archive of executable images, dependency descriptions and metadata (Management Resources serialized into a Deployment Plan) that can be used to move an Application and its Components from Platform to Platform, or between an Application Development Environment and a Platform (e.g. a storefront application with component binaries, database images and all the configurations needed to install and run).

Register – the act of creating a Plan on the platform.

Supported Formats - one or more data serialization format for data representation. JSON format is required, but other data serialization formats are also allowed. The ~~Platform~~*platform* *resource* identifies all Supported Formats in the optional ~~supportedFormatsUri~~supported_formats_uri attribute. If the ~~supportedFormatsUri~~supported_formats_uri attribute is absent from the ~~Platform~~*platform* *resource*, then only JSON is supported.

Appendix C. Normative Statements

C.1 Mandatory Statements

Tag	Statement
~~[CO-01]~~	~~An Application Component Template SHALL be referenced by a single Assembly Template.~~
~~[CO-02]~~	~~An Assembly Template SHALL NOT be instantiated until all of its Application Component Templates are successfully instantiated.~~
[PDP-02]	A Provider SHALL support the following archive formats for a PDP: • A PDP as a ZIP archive [ZIP]
[PDP-03]	A Provider SHALL support the following archive formats for a PDP: • A PDP as a TAR archive [TAR]
[PDP-04]	A Provider SHALL support the following archive formats for a PDP: • A PDP as a GZIP [~~GZIP~~RFC1952] compressed TAR archive
[PDP-10]	The format of the manifest file and the certificate file SHALL be as defined by the OVF specification [OVF].
[PDP-11]	A Platform Deployment Package (PDP) SHALL contain a single ~~Deployment~~ Plan file.
~~[PDP-12]~~	~~The Deployment Plan SHALL be located at the root of the PDP archive.~~
~~[PDP-13]~~	~~The Deployment Plan file SHALL be named "`camp.yaml`" and SHALL consist of a well-formed YAML 1.1 [YAML 1.1] file that conforms to the description provided in this section.~~
~~[PDP-17]~~	~~A Deployment Plan SHALL contain a single instance of a DeploymentPlan node.~~
~~[PDP-18]~~	~~This value SHALL be the Specification Version String of the CAMP specification to which this Deployment Plan conforms.~~
~~[PDP-19]~~	~~For Deployment Plans that conform to this document, the value of this node SHALL be "CAMP 1.1" as defined in Section **Error! Reference source not found.** "Specification Version".~~
~~[PDP-20]~~	~~Providers SHALL NOT regard the order of the ArtifactSpecifications within this array as semantically significant.~~
~~[PDP-21]~~	~~Providers SHALL NOT treat the order of ServiceSpecifications within this array as semantically significant.~~
~~[PDP-23]~~	~~Providers SHALL NOT treat the order of RequirementSpecifications within this array as semantically significant.~~
~~[PDP-24]~~	~~Providers SHALL NOT treat the order of CharacteristicSpecifications within this array as semantically significant.~~
~~[PDP-25]~~	~~Content Specifications SHALL declare either a String attribute "href" that references the content or a String attribute "data" whose value is the data or, but not both.~~
~~[PDP-26]~~	~~For IANA-assigned URI schemes (e.g. "http", "https", "ftp", etc.) the Provider SHALL engage the protocol as per the relevant spec.~~

[PDP-27] Providers SHALL support the “~~http” and “~~https” URI ~~schemes.~~scheme as defined in RFC 2818 [RFC2818].

[PDP-29] Providers SHALL understand this delimiter and SHALL NOT resolve any content if the archive format is unsupported.

~~[RE-01]~~[PLAN-01] The ~~following attributes~~Plan file SHALL be ~~present in a Link~~located at the root of the PDP archive.

[PLAN-02] The Plan file SHALL be named “`camp.yaml`”.

[PLAN-03] A Plan file SHALL contain a single instance of a Plan.

~~[RE-04]~~[PLAN-05] ~~Consumers SHALL NOT change the value of this attribute.~~For Plans that conform to this document, the value of this node SHALL be as defined in Section 1.8 “Specification Version”.

~~[RE-05]~~[PLAN-06] ~~Though both attributes of this type are optional, a valid RequirementTemplateLinks type SHALL have at least one of the following attributes:~~Plans SHALL use id values that are unique within the scope of the Plan.

[PLAN-07] Consumers SHALL follow the syntax and semantics described here when using URIs with a “pdp” scheme.

[PLAN-08] The Plan file SHALL conform to YAML 1.1 [YAML 1.1].

[PLAN-09] The Plan file SHALL conform to the description provided in this section.

[RE-06] If the Required boolean constraint for an attribute of a resource type has a value of "true", then ~~an instance of that~~a resource of this type SHALL have the attribute present.

[RE-07] This boolean indicates the mutability of the attribute’s value(s). “false” indicates that the value of the attribute, once set, SHALL NOT change for the lifetime of the resource.

[RE-09] “false” indicates that the value(s) of the attribute SHALL NOT be changed by ~~the~~ Consumers.

[RE-11] If present, ~~`representationSkew`~~`representation_skew` SHALL have one of the following values:

- “CREATING” – describes a resource that is in the process of being created. The client can expect that the resource will have a skew of “NONE” once this process has completed.
- “NONE” – is an assertion by the CAMP server that the information in the resource is an accurate representation of the underlying platform implementation. Absent some action by the client or some other event (e.g. platform shutdown), a resource with a skew of NONE can be expected to remain in synch with the platform implementation.
- “UNKNOWN” – indicates that the CAMP server cannot accurately depict the aspect of the platform implementation represented by this resource. Users can attempt to address the underlying issues(s) by manipulating this and/or other resources as specified by the API.
- “DESTROYING” – describes a resource that is in the process of being destroyed. The client can expect that the resource will cease to exist once this process has completed.

[RE-12] The following table lists the methods that SHALL be supported for each ~~`representationSkew`~~`representation_skew` value.

~~representationSkew~~representation_skew value	Methods Available
CREATING	GET, DELETE
NONE	All supported methods for that resource.
UNKNOWN	All supported methods for that resource.
DESTROYING	GET

~~[RE-14]~~	~~However, it SHALL NOT contradict the HTTP status code that is returned with the request.~~
~~[RE-16]~~	~~Because of the unique function of this resource, future versions of the CAMP specification SHALL NOT make non-backwards compatible changes to this resource.~~

[RE-18] This array SHALL contain at least ~~one PlatformEndpoint Resource~~oneLink.

[RE-19] References between the resources (~~PlatformEndpoints, PlatformEndpoint~~*platform_endpoints, platform_endpoint*, and ~~Platform~~*platform*) SHALL be self-consistent.

[RE-20] Each ~~PlatformEndpoint Resource~~*platform_endpoint resource* SHALL refer to exactly one ~~Platform Resource~~*platform resource*, and indicate the versions supported by the Platform.

~~[RE-21]~~	~~Because of the unique function of this resource, future versions of the CAMP specification SHALL NOT make non-backwards compatible changes to this resource.~~

~~[RE-22]~~[RE-22] For Platforms that implement this version of the CAMP specification, the value of this attribute SHALL be ~~"CAMP 1.1" as defined in section 1.7.~~as defined in Section 1.8, "Specification Version".

[RE-23] The values in this array SHALL be the Specification Version Strings of previous CAMP specification versions.

[RE-24] ~~PlatformEndpoint~~*platform_endpoint resources* that reference ~~Specification Version~~*platform resources* with a specification_version value of "CAMP 1.1" ~~Platform Resources~~ SHALL NOT include this attribute because no previous versions are compatible.

[RE-26] For Platforms that implement this version of the CAMP specification, the value of this attribute SHALL be ~~"CAMP 1.1" as defined in Section 1.7, "Specification Version".~~as defined in Section 1.8, "Specification Version".

[RE-27] The value of this attribute SHALL exactly match the value of the ~~specificationVersion~~specification_version attribute of any ~~PlatformEndpoint~~*platform_endpoint resource* that references this ~~Platform Resource~~*platform resource*.

[RE-29] The value of this attribute SHALL exactly match the value of the ~~implementationVersion~~implementation_version attribute of any ~~PlatformEndpoint~~*platform_endpoint resource* that references this ~~Platform Resource~~*platform resource*.

~~[RE-30]~~	~~The Platform resource SHALL indirectly reference a parameterDefinition resource that describes the pdpUri parameter.~~
~~[RE-31]~~	~~Consumers SHALL NOT change the value of this attribute.~~
~~[RE-33]~~	~~but Providers SHALL NOT instantiate an Assembly using an AssemblyTemplate that does not reference at least one ApplicationComponentTemplate.~~

~~[RE-34]~~	~~The ParameterDefinitions resource referenced by this attribute SHALL define parameters to allow setting the ‘name’, ‘description’, and ‘tags’ attributes of any new resource created in the course of interacting with this resource.~~
~~[RE-35]~~	~~This attribute SHALL be present.~~
~~[RE-36]~~	~~The specified value for each component attribute SHALL be in the range defined in the corresponding Platform Component Capability.~~

[RE-37]	The ~~ParameterDefinitions~~*parameter_definitions resource* referenced by this attribute SHALL define parameters to allow setting the ‘name’, ‘description’, and ‘tags’ attributes of any new resource created in the course of interacting with this resource.
[RE-38]	If this attribute is present in ~~an instance of this~~the resource, Providers SHALL support the POST method on that ~~instance~~resource in addition to the methods defined in Section ~~0~~1.1, “HTTP Method Support”.
[RE-39]	An ~~Assembly~~*assembly resource* SHALL have at least one reference to ~~an ApplicationComponent~~a *component resource*.
[RE-40]	For every format that the Platform supports, there SHALL be a Format resource Link that represents such a format.
[RE-41]	The Required JSON Format Resource SHALL be listed first in the ~~formatLinks~~format_links array.
[RE-42]	The name, ~~mimeType~~mime_type, version, and documentation attribute values for the JSON Format Resource SHALL reflect the above values.
[RE-43]	The ~~Platform~~*platform resource* SHALL provide a Link to the ~~TypeDefinitions~~*type_definitions resource* in the required attribute named ~~`typeDefinitionsUri`~~`type_definitions_uri`.
[RE-44]	If the array is non-empty, for every resource type that the Platform supports, there SHALL be a ~~TypeDefinition~~*type_definition resource* Link that represents such a resource type.
[RE-45]	For every attribute of the type not inherited from its super-types, there SHALL be an ~~AttributeDefinition resource Link~~AttributeLink that ~~represents~~references the *attribute_definition resource* that defines that attribute.
[RE-53]	Providers SHALL support the HTTP GET, PUT, and PATCH methods on all of the resources defined in this section.

~~[RE-55]~~	~~In addition to the methods defined in Section 0, “HTTP Method Support”, Providers SHALL support the HTTP POST method on the Platform resource as described in Section 6.11, “Registering an Application”.~~
~~[RE-56]~~	~~In addition to the methods defined in Section 0, “HTTP Method Support”, Providers SHALL support the HTTP POST and DELETE methods on instances of the AssemblyTemplate resource.~~
~~[RE-57]~~	~~In addition to the methods defined in Section 0, “HTTP Method Support”, Providers SHALL support the HTTP DELETE method on instances of the ApplicationComponentTemplate resource.~~
~~[RE-58]~~	~~In addition to the methods defined in Section 0, “HTTP Method Support”, Providers SHALL support the HTTP DELETE method on instances of the ApplicationComponentRequirement resource.~~
~~[RE-59]~~	~~In addition to the methods defined in Section 0, “HTTP Method Support”, Providers SHALL support the HTTP DELETE method on instances of the~~

	~~ApplicationComponentCapability resource.~~
~~[RE-60]~~	~~In addition to the methods defined in Section 0, “HTTP Method Support”, Providers SHALL support the HTTP DELETE method on instances of the PlatformComponentRequirement resource.~~
[RE-61]	In addition to the methods defined in Section ~~0~~1.1, “HTTP Method Support”, Providers SHALL support the HTTP DELETE ~~methods on instances of the Assembly resource as described in Section 6.14, “Deleting an Application Instance and a Deployed Application”.~~method on the *assembly resource*.
[RE-62]	In addition to the methods defined in Section ~~0~~1.1, “HTTP Method Support”, Providers SHALL support the HTTP DELETE method on ~~instances of~~ the ~~ApplicationComponent~~*component resource*.
~~[RE-63]~~	~~In addition to the methods defined in Section 0, “HTTP Method Support”, Providers SHALL support the HTTP DELETE method on instances of the PlatformComponent resource.~~
[RE-64]	In addition to the methods defined in Section ~~0~~1.1, “HTTP Method Support”, Providers SHALL support the HTTP POST method on ~~instances of~~ the ~~Operation~~*operation resource*.
[RE-65]	Consumers and Providers SHALL express Timestamps in UTC (Coordinated Universal Time), with the special UTC designator ("Z").
[RE-70]	When supporting such a Resource, a Provider SHALL implement it and serialize it as described in the corresponding sub-section.
[RE-71]	A Consumer SHALL serialize Resource data in its requests based on the definition of this Resource as described in the corresponding sub-section.
[RE-73]	On reception of a DELETE request a Provider SHALL remove the *assembly resource* from the system along with any *component resources* referenced by that assembly resource. (i.e. the tree of resources that was created when the application was instantiated).
[RE-74]	On reception of a DELETE request a Provider SHALL remove the reference to the *assembly resource* from the *assemblies resource’s* `assembly_links` array.
[RE-75]	The value of the `name` attribute in a *type_definition resource* SHALL match the value of the `type` attribute for the resource type that it describes.
[RE-76]	In cases where a sub-type adds additional constraints to an attribute inherited from its super-types (e.g. makes an optional attribute required), a Provider SHALL include an AttributeLink that references the *attribute_defintion resource* for that attribute.
[PR-01]	Providers SHALL provide representations of all available resources in JSON.
[PR-02]	~~Both~~ Consumers and Providers SHALL NOT transmit JSON objects that contain duplicate keys.
[PR-05]	~~Supported Formats SHALL be applied uniformly for all resources defined by this specification.~~Providers SHALL respond in the requested Supported Format.
[PR-07]	If the *If-Match* header field value in the request does not match the one on the server-side, the Provider SHALL send back a '412 Precondition Failed' status code.
[PR-09]	If an attribute is not part of the resource, an HTTP 4XX status code SHALL be returned.
[PR-11]	The Consumer SHALL URL encode the request parameters.

[PR-12] When one or more request parameters are specified for a PUT request, a Consumer SHALL NOT include attributes in the request entity body that are not specified in the request parameter.

[PR-13] Upon receiving such a request the Provider SHALL respond with a 400 status code.

[PR-18] If a POST request body does not contain a value for a required parameter, a "400 Bad Request" response SHALL be returned~~ with an Error Response Message Resource.~~.

[PR-19] If a POST request body does not contain an acceptable value for a parameter, a "400 Bad Request" response SHALL be returned~~ with an Error Response Message Resource.~~.

[PR-21] Consumers SHALL NOT send a request that changes the value of a resource attribute that is declared with a constraint of 'Mutable=false' or 'Consumer-mutable=false'.

[PR-22] On receiving such a request the Provider SHALL generate an HTTP response with 403 HTTP status code.

[PR-26] Providers SHALL support the HTTP PATCH method in conjunction with the "application/json-patch+json" media type with the following, additional provisions with respect to the operations defined in ~~section~~Section 4 of the JSON Patch specification ~~[JSON Patch]:~~:

[PR-27] Providers SHALL support the 'add', 'remove', and 'replace' operations.

[PR-29] ~~The~~To support the deployment of applications using a PDP, Providers SHALL accept the media types associated with the various formats ~~SHALL be~~ as follows:

- ZIP: "application/x-zip"

[PR-30] ~~The~~To support the deployment of applications using a PDP, Providers SHALL accept the media types associated with the various formats ~~SHALL be~~ as follows:

- TAR: "application/x-tar"

[PR-31] ~~The~~To support the deployment of applications using a PDP, Providers SHALL accept the media types associated with the various formats ~~SHALL be~~ as follows:

- GZIP compressed TAR: "application/x-tgz"

[PR-32] ~~Since~~To support the ~~DP is a YAML file, when registering an application~~deployment of applications using a ~~DP the associated media type~~Plan file, Providers SHALL ~~be~~accept the use of the "application/x-yaml~~".~~" media type.

~~[PR-34]~~	~~When a PDP is used to register the application, the Provider SHALL include the *pdpUri* attribute, which identifies the PDP from which the template was created, in the newly created AssemblyTemplate resource.~~
~~[PR-35]~~	~~When a DP is used to register the application, the Provider SHALL include the *dpUri* attribute, which identifies the DP from which the template was created, in the newly created AssemblyTemplate resource.~~
~~[PR-36]~~	~~When a PDP is used to register the application, the Provider SHALL include the *pdpUri* attribute, which identifies the PDP from which the template was created, in the newly created AssemblyTemplate resource.~~
~~[PR-37]~~	~~When a DP is used to register the application, the Provider SHALL include the *dpUri* attribute, which identifies the DP from which the template was created, in the newly created AssemblyTemplate resource.~~

[PR-41] TLS 1.1 [RFC4346] SHALL be implemented by the Provider.

[PR-47]	A Provider SHALL return only the attributes of the queried resource that match the

	attribute names passed as arguments of ‘select_attr’.
[PR-48]~~[PR-45]~~	~~Consumers~~On successfully processing an HTTP PUT request a Provider SHALL ~~NOT make assumptions about~~update all the ~~layout~~Consumer-mutable attributes of the ~~URIs or~~target resource, and only these, with the ~~structure~~values of the ~~URIs of~~matching attributes in the ~~resources~~request.
[PR-49]	To deploy an application by reference, a Consumer SHALL send an HTTP POST request to the URL of the *assemblies resource* as described in this section.
[PR-50]	On successfully processing the request the Provider SHALL create an *assembly resource* and return a 201 Created status code in the HTTP response.
[PR-51]	The Provider SHALL include the *Location* header in the HTTP response and the value of this header SHALL reference the newly created *assembly resource*.
[PR-52]	The Provider SHALL update the `assembly_links` attribute of the *assemblies resource* to include a reference to the newly created *assembly resource*.
Error! Reference source not found.	On successfully processing the request the Provider SHALL create an *assembly resource* and return a 201 Created status code in the HTTP response.
Error! Reference source not found.	The Provider SHALL include the *Location* header in the HTTP response and the value of this header SHALL reference the newly created assembly resource.
Error! Reference source not found.	The Provider SHALL update the `assembly_links` attribute of the *assemblies resource* to include a reference to the newly created *assembly resource*.
[PR-56]	Providers that support the *plans resource* and *plan resources* SHALL support the registration of Plans via an HTTP POST request on the *plans resource* as described in this section.
[PR-57]	On successfully processing the request the Provider SHALL create a *plan resource* and return a 201 Created status code in the HTTP response.
[PR-58]	The Provider SHALL include the *Location* header in the HTTP response and the value of this header SHALL reference the newly created *plan resource*.
[PR-59]	The Provider SHALL update the `plan_links` attribute of the *plans resource* to include a reference to the newly created *plan resource*.
[PR-60]	Providers SHALL support the deployment of applications via HTTP POST requests on the *assemblies resource* in which the entity body of the request contains the PDP or Plan file that is being deployed.
[PR-61]	Providers that support the *plans resource* and *plan resources* SHALL support the registration of Plans via HTTP POST requests on the *plans resource* in which the entity body of the request contains the PDP or the Plan file that is being registered.
[PR-62]	On successfully processing the request the Provider SHALL create a *plan resource* and return a 201 Created status code in the HTTP response.
[PR-63]	The Provider SHALL include the *Location* header in the HTTP response and the value of this header SHALL reference the newly created *plan resource*.

[PR-64]	The Provider SHALL update the `plan_links` attribute of the Plans resource to include a reference to the newly created *plan resource*.
[PR-68]	To support the deployment of applications via a reference to either a PDP, Plan file, or *plan* resource, Providers SHALL accept the "application/json" media type.
[PR-69]	To support the registration of Plans via a reference to either a PDP or a Plan file, Providers SHALL accept the "application/json" media type.
Error! Reference source not found.	To support the registration of Plans using a PDP, Providers SHALL accept the media types associated with the various formats as follows: • ZIP: "application/x-zip"
Error! Reference source not found.	To support the registration of Plans using a PDP, Providers SHALL accept the media types associated with the various formats as follows: • TAR: "application/x-tar"
Error! Reference source not found.	To support the registration of Plans using a PDP, Providers SHALL accept the media types associated with the various formats as follows: • GZIP compressed TAR: "application/x-tgz"
Error! Reference source not found.	To support the registration of Plans using a Plan file, Providers SHALL accept the use of the "application/x-yaml" media type.
[PR-74]	Providers SHALL support the deployment of applications via HTTP POST requests on the *assemblies resource* as described in this section.
[PR-75]	Providers that support the *plans resource* and *plan resources* SHALL support the registration of Plans via HTTP POST requests on the *plans resource* as described in this section.
[EX-03]	Extensions SHALL NOT change or remove any features or functionality of this specification.
[EX-04]	Each ~~Extension~~extension SHALL satisfy all the criteria in ~~the Conformance section~~Section 8, “Conformance”, and SHALL NOT contradict any normative statements in this document.
[EX-05]	The Extension Developer SHALL use a unique name for new ~~Entities~~entities within an existing namespace.
[EX-06]	Entities added by an ~~Extension~~extension SHALL NOT interfere with names of existing entities, including any added by another ~~Extension~~extension.
~~Error! Reference source not found.~~[EX-08]	For every ~~Extension~~extension available, there SHALL be an ~~Extension~~*extension resource* Link that represents the ~~Extension~~extension.
[EX-09]	The *platform resource* SHALL provide a Link to the *extensions resource* in the required attribute named `extensions_uri`.
[RMR-01]	If a Consumer includes this node in a Plan, the value of this node SHALL reference a Consumer-visible resource within the target Platform.

[RMR-02]	In addition to the methods defined in Section 1.1, "HTTP Method Support", Providers SHALL support the HTTP POST method on the *assemblies resource* as described in Section 6.11, "Deploying an Application".
[RMR-03]	The *assemblies resource* SHALL indirectly reference *parameter_definition resources* that describe the `pdp_uri`, `plan_uri`, `pdp_file`, and `plan_file` parameters.
[RMR-04]	Providers that support Plans SHALL include this attribute in all *assembly resources*.
[RMR-05]	In addition to the methods defined in Section 1.1, "HTTP Method Support", Providers SHALL support the HTTP POST method on the *plans resource* as described in Section 6.12, "Registering a Plan".
[RMR-06]	The *plans resource* SHALL indirectly reference *parameter_definition resources* that describe the `pdp_uri`, `plan_uri`, `pdp_file`, and `plan_file` parameters.
[RMR-07]	The schema of the *plan resource* returned from a CAMP Provider SHALL conform to the schema for Plans described in Section 4.3, "Plan Schema", with the following additional requirements:
[RMR-08]	Representations of the *plan resource* SHALL be serialized as JSON, unless another format is negotiated.
~~**Error! Reference source not found.**~~**[RMR-11]**	~~The Platform resource SHALL provide a Link to the Extensions resource in the required attribute named `extensionsUri`.~~Regardless of whether a Consumer attempts to create an *assembly resource* by POSTing to the *assemblies resource* or creates a *plan resource* by POSTing to the *plans resource*, a Provider that supports the *plans* and *plan resources* SHALL create a *plan resource* for every deployed application.

[RMR-12]	Providers that support *plans* and *plan resources* SHALL advertise such support using the following *extension resource*: **[RMR-12]**
	`{`
	`  "uri": <as appropriate>,`
	`  "name": "CAMP Plans Extension",`
	`  "type": "extension",`
	`  "description": "indicates support for plans and plan resources",`
	`  "version": "CAMP 1.1",`
	`  "documentation": "http://docs.oasis-open.org/camp/camp-spec/v1.1/camp-spec-v1.1.pdf"`
	`}`

[RMR-13]	Providers SHALL support PDPs that use either the ZIP [ZIP], TAR [TAR], or GZIP [RFC1952] compressed TAR formats.
[MO-02]	A sub-type SHALL NOT loosen the constraints of an attribute inherited from its super-type(s).
[MO-03]	A resource type MAY inherit from more than one super-type.
[MO-04]	If there is an attribute name collision when a sub-type inherits from multiple super-types, the inherited attributes of the same name SHALL NOT contradict the constraints and semantics of the attributes defined in its super-types.
[MO-05]	All CAMP resources SHALL inherit directly or indirectly from this resource.
[MO-06]	Links in this array SHALL NOT either directly or transitively point to the described resource.

C.2 Non-Mandatory Statements

Tag	Statement
[PDP-01]	A PDP archive MAY include other files related to the application including, but not limited to, language-specific bundles, resource files, application content files such as web archives, database schemas, scripts, source code, localization bundles, and icons; and metadata files such as manifests, checksums, signatures, and certificates.
[PDP-05]	Providers MAY support additional archive formats for the PDP.
[PDP-06]	A PDP MAY contain a manifest file, named `camp.mf`, at the root of the archive.
[PDP-07]	A Provider SHOULD reject a PDP if any digest listed in the manifest does not match the computed digest for that file in the package.
[PDP-08]	A PDP MAY contain a certificate, named `camp.cert`, at the root of the archive.
[PDP-09]	A Provider SHOULD reject any PDP for which the signature verification fails.
[PDP-14]	Providers MAY reflect the value of this attribute in the names of any resources that are created in the processing the ~~Deployment Plan.~~*plan.*
[PDP-15]	Providers MAY reflect the value of this attribute in the descriptions of the resources that are in the processing the ~~Deployment Plan.~~*plan.*
[PDP-16]	Providers MAY reflect the values of this attribute in the tags of the resources that are created in the processing of the ~~Deployment Plan.~~*plan.*
[PDP-22]	The artifact MAY be contained within the PDP or MAY exist in some other location.
[PDP-28]	A Provider MAY support additional URI schemes~~.~~ listed at http://www.iana.org/assignments/uri-schemes/uri-schemes.xhtml.
~~**[RE-02]**~~	~~Other attributes, not defined in this specification, MAY also be present.~~
~~**[RE-03]**~~	~~The value of this attribute MAY be changed by the Provider.~~
[RE-08]	"true" indicates that the value of the attribute MAY change due to the actions or activity of either the provider or the consumer.
[RE-10]	A value of "true" indicates that ~~consumers~~Consumers MAY change the value of the attribute.
[RE-13]	For each ~~`representationSkew`~~`representation_skew` value, CAMP Providers MAY support HTTP methods in addition to those listed in the corresponding row of Table 5-1.
[RE-15]	A Provider MAY concurrently offer multiple instances of the CAMP API.
[RE-17]	A Provider MAY expose the ~~PlatformEndpoints~~platform_endpoints and corresponding ~~PlatformEndpoint~~platform_endpoint resources in a way that allows for version discovery before the client has authenticated.
[RE-25]	Multiple implementations of the same CAMP specification MAY be offered concurrently.
[RE-28]	A Provider MAY choose to offer multiple implementations of the same CAMP specification.
~~**[RE-32]**~~	~~An AssemblyTemplate MAY have zero references to ApplicationComponentTemplate resources~~
~~**Error!**~~	Multiple resources MAY reference the same ~~ParameterDefinitions~~

~~Reference source not found.~~[RE-46]	~~Resource~~*parameter_definitions resource*.
[RE-47]	The Operation MAY require content in the body of the POST, such as parameters.
[RE-48]	The response to a POST request on an ~~Operation~~*operation resource* SHOULD indicate what changes were made on the target resource.
[RE-49]	For asynchronous operations, the response SHOULD indicate how to track the progress of the request operation.
[RE-50]	The documentation SHOULD describe the behavior of the operation, the form of the body expected in POST requests, and the semantics and form of the response to such requests.
[RE-51]	When a "value" attribute is supplied, any timestamp provided in this attribute SHOULD correspond to when that value was observed.
[RE-52]	Extensions MAY be defined to govern common sensor management operations, such as enabling, disabling, adjusting collection frequency, specifying the history of values which should be remembered, or collecting immediately.
[RE-54]	Providers MAY elect to support additional HTTP methods in addition to those described here.
[RE-68]	This attribute MAY have one of the following values: • "RUNNING" – indicates that the component is functioning as expected. • "ERROR" – indicates that the component has encountered some sort of error.
[RE-69]	Providers MAY extend this list with additional values.
[PR-03]	If a Consumer sends a Provider a request containing duplicate keys in a JSON object, the Provider SHOULD reject the request by sending back a '400 Bad Request' status code.
[PR-04]	If a Provider sends a Consumer a response containing duplicate keys in a JSON object, the Consumer SHOULD raise an error to the user indicating the response from the server was malformed.
[PR-06]	All PUT requests that update a resource SHOULD contain the *If-Match* header field with a single entity tag value.
[PR-08]	To retrieve a subset of the attributes in a resource, the Consumer MAY use the '~~SelectAttr'~~select_attr' request parameter in conjunction with the HTTP GET method.
[PR-10]	The "~~SelectAttr~~select_attr" query parameter MAY appear more than once (separated by an "&").
[PR-14]	Parameters MAY be included when performing a POST request on any resource with a ~~parameterDefinitionsUri~~`parameter_definitions_uri` attribute defined.
[PR-15]	Parameters MAY have the same name as an ~~Attribute on~~attribute of the ~~Resource~~resource.
[PR-16]	In such cases the Provider SHOULD set ~~the Attribute~~that attribute to take the value of the ~~Parameter~~parameter OR clearly document alternate behavior.
[PR-17]	The ~~parameterExtensionUri~~`parameter_extension_uri` MAY be used to reference the ~~Extension~~extension which documents how the parameter is handled.
[PR-20]	All HTTP responses that return representation of a resource SHOULD use strong *Etag*

response header field indicating the current value of the entity tag for the resource.

[PR-23]	Consumers MAY use the HTTP PUT method to replace the representation of a resource, in its entirety, with a new representation that adds, omits or replaces the values for some of the attributes.
[PR-24]	Alternatively, Consumers MAY use the HTTP PATCH [HTTP PATCH] method and the "application/json-patch+json" media type ~~[JSON Patch~~[RFC6902] to add, delete, or replace specific attributes.
[PR-25]	If a resource attribute is present on a resource and if an HTTP PUT request omits that attribute, it SHOULD be treated by the Provider as a request to delete the attribute.
[PR-28]	Providers MAY support the 'move', 'copy, and 'test' operations.
[PR-33]	The JSON object MAY contain additional name-value pairs that are not defined in this specification.
~~[PR-38]~~	~~A Provider MAY have additional state values that it allows.~~
~~[PR-39]~~	~~The JSON object MAY contain additional name-value pairs that are not defined in this specification.~~
[PR-40]	~~Therefore, requests~~Requests sent from Consumers across unsecured networks SHOULD use the HTTPS protocol.
[PR-42]	TLS 1.2 [RFC5246] is RECOMMENDED.
[PR-43]	When TLS is implemented, the following cipher suites are RECOMMENDED to ensure a minimum level of security and interoperability between implementations: • TLS_RSA_WITH_AES_128_CBC_SHA (mandatory for TLS 1.1/1.2)
[PR-44]	When TLS is implemented, the following cipher suites are RECOMMENDED to ensure a minimum level of security and interoperability between implementations: • TLS_RSA_WITH_AES_256_CBC_SHA256 (addresses 112-bit security strength requirements)
[PR-45]~~[EX-01]~~	~~Using Requirements and Capabilities is RECOMMENDED instead of Extensions, if possible.~~For each Supported Format, Consumers MAY request any resource from the Provider in that format.
[PR-46	The JSON object MAY contain additional name-value pairs that are not defined in this specification.
[EX-02]	Extensions MAY be added by registering the new functionality in the ~~Extensions~~*extensions resource*.
[EX-07]	The use of your registered ICAAN Internet domain name followed by a colon (":") character as a prefix to all your entity names is RECOMMENDED to comply with these requirements.
[EX-10]	New attributes MAY be added to an existing resource using an Extension if the Unique Name Requirement in ~~0~~7.1 is met.
[RMR-09]	Any href attributes of ServiceSpecifications SHOULD refer to a Service resource.
[RMR-10]	All href attributes in the *plan resource* SHOULD be set to a consumer accessible URL. If the original Plan file referred to a local file, the URL indicates where the Provider stored the content.
[MO-01]	A sub-type MAY further restrict the constraints of an attribute inherited from its super-type(s).

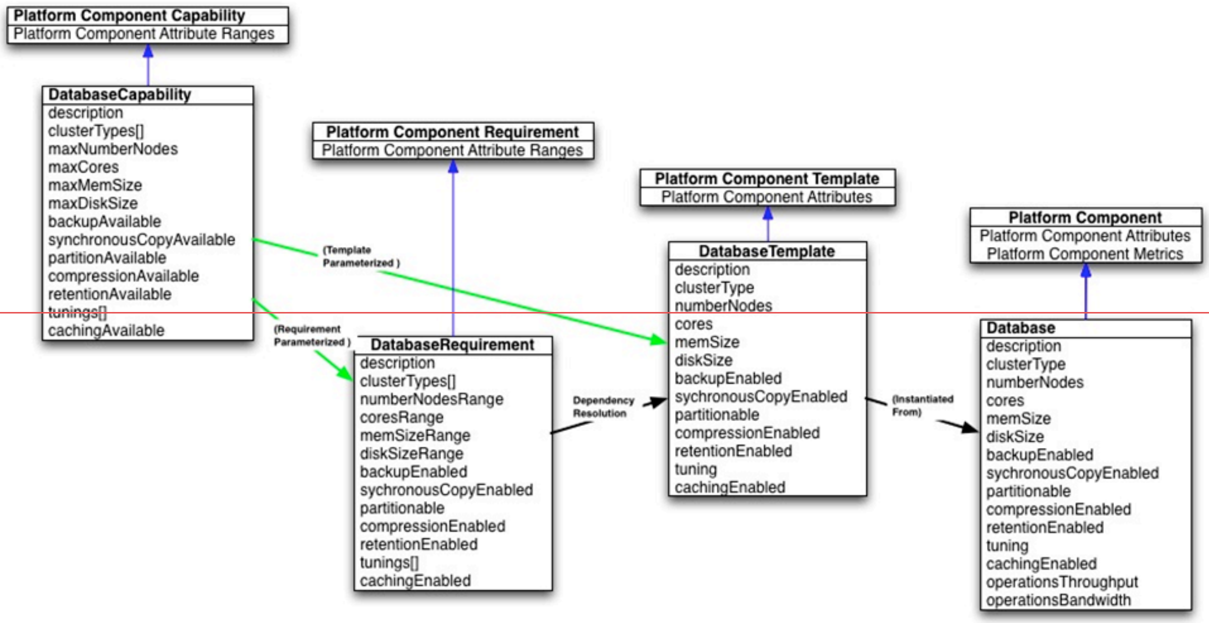
[MO-07]	If a type inherits only from the *camp_resource* type then this attribute MAY be absent.

~~Appendix D. Example Database Platform Component (Non-Normative)~~

~~One important Platform Component that can be provided by many platforms is a database. The following sections illustrate how database components could be provided by extending the model defined in this specification. The material in these sections is non-normative.~~

~~D.1 Model~~

~~A Database Platform Component provides four sub-classed resources as shown in the below diagram:~~



~~*Figure D-1: Database Platform Component Model*~~

~~D.2 DatabaseCapability~~

~~For an Application Administrator, a Database Capability represents the definition of a database platform component and its range of capabilities. This resource has the following, general representation:~~

```
~~{~~
~~    "uri": URI,~~
~~    "name": String,~~
~~    "type": "databaseCapability",~~
~~    "description": String,~~
~~    "representationSkew": String, ?~~
~~    "tags": [ String, + ], ?~~
~~    "clusterTypes": [~~
~~        String, +~~
~~    ],~~
~~    "maxNumberNodes": String,~~
~~    "maxCores": String,~~
~~    "maxMemSize": String,~~
~~    "maxDiskSize": String,~~
~~    "backupAvailable": Boolean,~~
~~    "synchronousCopyAvailable": Boolean,~~
~~    "partitionAvailable": Boolean,~~
~~    "compressionAvailable": Boolean,~~
~~    "retentionAvailable": Boolean,~~
~~    "tunings": [~~
~~        String, +~~
~~    ],~~
~~    "cachingAvailable": Boolean,~~
~~}~~
```

~~Each type of DatabasePlatformComponent implements this class and populates the attributes in the list below.~~

~~D.2.1 clusterTypes~~

~~**Type:** String[]~~

~~**Required**: true~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~An array of supported cluster types. Values include: “active” and “passive”.~~

~~D.2.2 maxNumberNodes~~

~~**Type:** String~~

~~**Required**: true~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~Expresses the maximum number of supported nodes for scaling purposes.~~

~~D.2.3 maxCores~~

~~**Type:** String~~

~~**Required**: true~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~Expresses the maximum number of supported cores for scaling purposes.~~

~~D.2.4 maxMemSize~~

~~**Type:** String~~

~~**Required**: true~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~Expresses the maximum size of supported memory for an instance.~~

~~D.2.5 maxDiskSize~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~Expresses the limit to the size of a disk for an instance.~~

~~D.2.6 backupAvailable~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if backup can be enabled.~~

~~D.2.7 synchronousCopyAvailable~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if synchronous copy can be enabled~~

~~D.2.8 partitionAvailable~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if partitioning can be enabled~~

~~D.2.9 compressionAvailable~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if compression can be enabled~~

~~D.2.10 retentionAvailable~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if retention can be enabled~~

~~D.2.11 tunings~~

~~**Type:** String[]~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~An array of supported tuning types. Values include: “OLAP”, “DataMining”, and “Spatial”~~

~~D.2.12 cachingAvailable~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if caching can be enabled~~

~~D.3 DatabaseRequirement~~

~~For an Application Administrator, a Database Requirement represents an Application Component’s requirements on a database platform component and its required range of capabilities. This resource has the following, general representation:~~

```
{
  "uri": URI,
  "name": String,
  "type": "databaseRequirement",
  "description": String,
  "representationSkew": String, ?
  "tags": [ String, + ], ?
  "clusterTypes": [ String, + ],
  "numberNodesRange": String,
  "coreRange": String,
  "memSizeRange": String,
  "diskSizeRange": String,
  "backupEnabled": Boolean,
  "synchronousCopyEnabled": Boolean,
  "partitionEnabled": Boolean,
  "compressionEnabled": Boolean,
  "retentionEnabled": Boolean,
  "tunings": [ String, + ],
  "cachingEnabled": Boolean
}
```

~~Each type of DatabaseRequirement implements this class and populates the attributes in the list below.~~

~~D.3.1 clusterTypes~~

~~**Type:** String[]~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~An array of required cluster types. Values include: “active” and “passive”.~~

~~D.3.2 numberNodesRange~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~Expresses the range of the number of nodes for scaling purposes~~

~~D.3.3 coreRange~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~Expresses the required range of number of cores.~~

~~D.3.4 memSizeRange~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~Expresses the range of required memory sizes.~~

~~D.3.5 diskSizeRange~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~Expresses the range if disk sizes required.~~

~~D.3.6 backupEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~true if backup is required.~~

~~D.3.7 synchronousCopyEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if synchronous copy is required.~~

~~D.3.8 partitionEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~true if partitioning is required.~~

~~D.3.9 compressionEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~true if compression is required.~~

~~D.3.10 retentionEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~true if retention is required.~~

~~D.3.11 tunings~~

~~**Type:** String[]~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~An array of required tuning types. Values include: "OLAP", "DataMining" and "Spatial".~~

~~D.3.12 cachingEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: true**~~

~~true if caching is required.~~

~~D.4 DatabaseTemplate~~

~~A Database Template represents the desired configuration of a Database Platform Component with specific values for the component capabilities. The specified value for each component attribute shall be in the range defined in the corresponding Database Capability. Some PaaS offerings might only offer a fixed number of Database Template instances that cannot be modified (read-only, no new instances) indicating pre-tuned and configured pools of database resources from which these draw. Other PaaS offerings may allow newly created Database Template instances with values in any combination. This resource has the following, general representation:~~

```
~~{~~
~~    "uri": URI,~~
~~    "name": String,~~
~~    "type": "databaseTemplate",~~
~~    "description": String,~~
~~    "representationSkew": String, ?~~
~~    "tags": [ String, + ], ?~~
~~    "clusterType": String,~~
~~    "numberNodes": String,~~
~~    "cores": String,~~
~~    "memSize": String,~~
~~    "diskSize": String,~~
~~    "backupEnabled": Boolean,~~
~~    "synchronousCopyEnabled": Boolean,~~
~~    "partitionEnabled": Boolean,~~
~~    "compressionEnabled": Boolean,~~
~~    "retentionEnabled": Boolean,~~
~~    "tuning": String,~~
~~    "cachingEnabled": Boolean~~
~~}~~
```

~~Each type of DatabaseTemplate implements this class and populates the attributes in the list below.~~

~~D.4.1 clusterType~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~The desired cluster type. Values include: "active" and "passive"~~

~~D.4.2 numberNodes~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~The desired number of nodes for an instance~~

~~D.4.3 cores~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~The desired number of cores for an instance~~

~~D.4.4 memSize~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~The desired size of memory for an instance~~

~~D.4.5 diskSize~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~The desired size of a disk for an instance.~~

~~D.4.6 backupEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if backup is desired for an instance.~~

~~D.4.7 synchronousCopyEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if synchronous copy is desired for an instance.~~

~~D.4.8 partitionEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if partitioning is desired for an instance~~

~~D.4.9 compressionEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if compression is desired for an instance~~

~~D.4.10 retentionEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if retention is desired for an instance~~

~~D.4.11 tuning~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~The desired tuning types. Values include: “OLAP”, “DataMining” and “Spatial”~~

~~D.4.12 cachingEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~true if caching is desired for an instance~~

~~D.5 Database~~

~~A Database represents the runtime instance of a Database Template and its configuration of component attributes as well as metrics associated with those attributes. Each Application’s use of a Database is represented by an instance of this resource. This resource has the following, general representation:~~

```
{
  "uri": URI,
  "name": String,
  "type": "database",
  "description": String,
  "representationSkew": String, ?
  "tags": [ String, + ], ?
  "externalManagementResource": String,
  "clusterType": String,
  "numberNodes": String,
  "cores": String,
  "memSize": String,
  "diskSize": String,
  "backupEnabled": Boolean,
  "synchronousCopyEnabled": Boolean,
  "partitionEnabled": Boolean,
  "compressionEnabled": Boolean,
  "retentionEnabled": Boolean,
  "tuning": String,
  "cachingEnabled": Boolean,
  "operationsThroughput": String,
  "operationsBandwidth": String
}
```

~~Each type of Database implements this class and populates the attributes in the list below.~~

~~D.5.1 externalManagementResource~~

~~**Type:** URI~~

~~**Required: false**~~

~~**Mutable: false**~~

~~A URI to an external management interface to manage this platform component (such as an IaaS API to manage the virtual machines that make up this database). This is platform dependent and requires external documentation to understand its meaning.~~

~~D.5.2 clusterType~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: false**~~

~~The actual cluster type. Values include: “active” and “passive”~~

~~D.5.3 numberNodes~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: false**~~

~~The actual number of nodes for an instance.~~

~~D.5.4 cores~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: false**~~

~~The actual number of cores for an instance.~~

~~D.5.5 memSize~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: false**~~

~~The actual size of memory for an instance.~~

~~D.5.6 diskSize~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: false**~~

~~The actual size of a disk for an instance.~~

~~D.5.7 backupEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: false**~~

~~true if backup is enabled for this instance.~~

~~D.5.8 synchronousCopyEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: false**~~

~~true if synchronous copy is enabled for this instance.~~

~~D.5.9 partitionEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: false**~~

~~true if partitioning is enabled for this instance.~~

~~D.5.10 compressionEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: false**~~

~~true if compression is enabled for this instance.~~

~~D.5.11 retentionEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: false**~~

~~true if retention is enabled for this instance.~~

~~D.5.12 tuning~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: false**~~

~~The actual tuning type of the database instance. Values include: “OLAP”, “DataMining” and “Spatial”~~

~~D.5.13 cachingEnabled~~

~~**Type:** Boolean~~

~~**Required: true**~~

~~**Mutable: false**~~

~~true if caching is enabled for this instance.~~

~~D.5.14 operationsThroughput~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~The billable operations per second.~~

~~D.5.15 operationsBandwidth~~

~~**Type:** String~~

~~**Required: true**~~

~~**Mutable: true**~~

~~**Consumer-mutable: false**~~

~~The billable MegaBytes per second.~~

~~Appendix E.~~Appendix D. Revision History

Revision	Date	Editor	Changes Made
1	2012-12-04	Anish Karmarkar	Applied OASIS template to version 1.0
2	2012-12-18	Anish Karmarkar	Included resolutions for issues 7, 12, 13, 15, 24, 33.
3	2013-02-05	Anish Karmarkar	Included resolutions for issues 2, 6, 10, 14, 25, 35
4	2013-02-12	Adrian Otto	Included resolutions for 19, 38
5	2013-02-13	Adrian Otto	Included resolutions for 1, 49
6	2013-02-27	Adrian Otto	Included resolutions for 36, 48, 53
7	2013-02-27	Adrian Otto	Included resolutions for 34, 52
8	2013-03-13	Adrian Otto	Included resolution for 40
9	2013-04-29	Anish Karmarkar	Included resolutions for 50, 57
10	2013-05-01	Adrian Otto	Included resolution for 30, 60
11	2013-06-05	Adrian Otto	Included resolution for 58
12	2013-06-13	Gilbert Pilz, Tom Rutt	Included resolutions for issues 17, 67 and 68
13	2013-06-27	Gilbert Pilz	Included resolution for 3.
14	2013-07-01	Tom Rutt	Updated Figures for 3, kept revision marks from 12
15	2013-07-08	Gilbert Pilz	Included resolution for 4.
16	2013-07-10	Gilbert Pilz, Tom Rutt	Included resolution for 9. Miscellaneous, non-material changes and cleanups.
17	2013-07-11	Gilbert Pilz, Tom Rutt	Included resolution for 65. Miscellaneous, non-material changes and cleanups.
18	2013-07-19	Gilbert Pilz, Tom Rutt	Includes resolution for 55. Miscellaneous, non-material changes and cleanups.
19	2013-07-25	Gilbert Pilz	Included resolutions for 45, 56, 61, 75, 76, and 78. Miscellaneous, non-material changes and cleanups.
20	2013-07-26	Gilbert Pilz, Adrian Otto	Added names to Appendix A, “Acknowledgements”. Tagged normative statements that were missed in WD19. Homogenized captions for figures and tables. Homogenized and corrected cross-references. Miscellaneous editorial cleanups.
21	2013-08-14	Gilbert Pilz	Include resolution for 81. Fix the omission of the resolution for 60.

22	2013-08-21	Gilbert Pilz	Include resolution for 71. Fixed a couple of cross references.
23	2013-09-11	Gilbert Pilz	Fix bug where the resolution to issue 54 was not incorporated as the TC directed. Fix references to use the tag [RFC6902] to be consistent with other references to RFCs. Miscellaneous editorial cleanups.
24	2013-09-18	Gilbert Pilz	Include resolutions for 51 and 112. Miscellaneous editorial cleanups.
25	2013-10-10	Anish Karmarkar, Gilbert Pilz, Tom Rutt	Include resolutions for 18, 31, and 111. Numerous reformats to fix effect of Word bug. Miscellaneous editorial cleanups.
26	2013-10-23	Anish Karmarkar	Include resolutions for 89, 92, 93, 94, 99, 119, 120, 122, 123, 125, 127, 131, 136, 137, 138, and 145. Clean up of Appendix C and conformance item tags is not yet completely done. Update of figures not yet done.
27	2013-10-25	Anish Karmarkar	Added Tom's figures. Fixed appendix C. Added tags for new normative statements. Some ed. cleanup.
28	2013-10-31	Gilbert Pilz	Include resolutions for 85, 115, and 135. Miscellaneous editorial cleanups.
29	2013-11-06	Gilbert Pilz	Include resolutions for 83 and 149. Miscellaneous editorial cleanups.
30	2013-11-14	Gilbert Pilz	Include resolutions for 74, 80, and 86. Miscellaneous editorial cleanups.
31	2013-11-21	Gilbert Pilz, Tom Rutt	Include resolutions for 72, 73, 98, 105, 109, 110, 113, 116, 117, 118, and 144. Miscellaneous editorial cleanups.
32	2013-12-03	Adrian Otto, Anish Karmarkar, Tom Rutt, Gilbert Pilz	Include resolution for 151. Miscellaneous editorial cleanups.
33	2013-12-09	Adrian Otto	Editorial correction of 8 instances of RequirementType in section 4 to requirement_type in accordance with the resolution to issue 151.
34	2013-12-11	Gilbert Pilz	Include resolution for 157. Miscellaneous editorial cleanups.
35	2014-01-09	Gilbert Pilz	Include resolution for 44 and 150. Miscellaneous editorial cleanups.
36	2014-01-16	Gilbert Pilz	Include resolution for 155. Clean up broken references.
37	2014-01-23	Gilbert Pilz, Tom Rutt	Include resolutions for 156, 158, 160, and 161.
38	2014-01-29	Gilbert Pilz	Include resolution for 63. Add references to type definitions package and TA document on

			cover page. Update acknowledgements. Miscellaneous editorial fixes.
39	2014-02-03	Gilbert Pilz	Accepted all changes to clean up candidate for CD04.