Hewlett Packard Enterprise

Cloud Service Automation

External Pricing Integration Guide

Software version: 4.80

Document release date: January 2017

Software release date: January 2017

Contents

Introduction

Service offerings published in the HPE Cloud Service Automation (CSA) catalog are offered to consumers for a price defined by the service provider. The service provider can either configure the price in CSA directly or link the offering to an external system providing the price.

This document describes basic concepts of externalized pricing implemented in CSA and also provides technical details for the key concepts such as the communication protocol used for the price information exchange, requirements for the external system, and limitations of the implemented solution.

Basic concepts

First, a few key points are enumerated and then described in detail:

- The service offerings are defined in CSA.
- The price for a service is defined in an external pricing system (EPS).
- Price information is requested from the EPS when CSA must show or process the price.
- There can be only one EPS instance configured for one CSA instance.
- The implementation is not tightly coupled with any specific EPS product.
- CSA defines the interface for interaction with an EPS. The EPS is expected to implement that interface, allowing CSA to interact with any EPS that implements this interface layer.

Interaction with EPS for price information exchange

Whenever CSA needs to process the price information, it makes a request to the EPS for the price of an offering. CSA always initiates the communication and the EPS responds immediately with price information or an error message. CSA sends the price request to the EPS for the following scenarios:

- When a service offering is configured, to validate the pricing information specified.
- When a service offering is displayed to a consumer, to inform about the price.
- When a service offering is being fulfilled, to store the price for further processing.

Single EPS configured in CSA

There can be only one EPS instance configured for one CSA instance. All service offerings created in this CSA instance can leverage only one EPS. Because externalized pricing is optional, some offerings can still be priced directly in CSA. However, all offerings with externalized pricing get the price from the single external pricing system.

CSA defines the interface

The pricing implementation is not coupled with any specific EPS product. Instead, CSA requires an EPS that is configured in CSA to implement the interface defined by CSA. In practice, an adapter must be implemented and put in front of the real pricing system, so that CSA and the real EPS can exchange the information.

Communication protocol

The communication protocol used for the information exchange between CSA and an EPS is defined by CSA. The EPS is responsible for providing a façade that exposes the required interface.

The communication is based on HTTP/S calls according to REST protocol. CSA sends a request for price to the EPS. The request contains details about the service offering and additional contextual information about the service requester (the consumer user). The request is processed by the EPS and a response is returned to CSA in a synchronous manner. The response contains information about the service price with details about the prices for selected and available service options and customization properties.

Let’s look at an example request payload.

Price request

Whenever CSA requires information about an offering price, a request is generated and sent to the EPS. Sample price request body is as shown below:

```
{
  "protocol-version": 1,
  "price-system-properties": {
    "price-list-id": "21321"
  },
  "user": "john",
  "organization": "ACME_INC",
  "catalog-name": "HARDWARE_TOOLS",
  "requested-date": "2010-01-01T12:00:00.100Z",
  "supported-periods": ["year", "month", "week", "day", "hour", "minute"],
  "base-price-key": "screwdriver-pack",
  "options": {
    "057f0ed07faa48b392a4135cbceb5fbd": {
      "price-key": "screwdriver-pack.chrome-plated",
      "selected": false
    },
    "13BBC2AEB89445D5AC51AE8BBD65FED4": {
      "price-key": "screwdriver-pack.numOfTools",
      "value": "5",
      "unit-measurement" : "Pieces",
      "selected": true
    }
  }
}
```

The price request contains these important elements:

- **price-system-properties** – A set of properties that the EPS may want to maintain. It originates from the EPS response. CSA ignores the values and just sends them back in subsequent requests that are related to the same service subscription.
- **user**, **organization**, **catalog-name** – Additional contextual information about the price request, made by the service requester.
- **supported-periods** – The EPS decides on the price currency and recurrence period, in the case of a recurring price. CSA only accepts recurrence periods that it supports.
- **base-price-key** - ID of the price in the EPS. CSA takes care of the information translation between CSA and the EPS. So, an offering called Screwdriver Pack in CSA may be translated to screwdriver-pack in the EPS terminology.
- **price-key** - ID of the price in the EPS. This price is related to the optional service customization such as different colors, better materials, or a bigger pack of screwdrivers.

Complete documentation for the price request format can be found in

Appendix A: Protocol details on page 9.

Price response

The price request is processed by the EPS and price is calculated based on the selected options of the service offering and additional contextual information contained in the request. CSA expects that the EPS would return the price information in the response within an acceptable time. An example response should look like this:

```
{
  "protocol-version": 1,
  "price-system-properties": {
    "price-list-id": "21321"
  },
  "currency": "USD",
  "period": "month",
  "total-price": {
    "init-price": 20,
```

```
    "recurring-price": 0,
    "message": "Order now!"
  },
  "base-price": {
    "price-key": " screwdriver-pack",
    "init-price": 20,
    "recurring-price": 0,
    "message": "20% discount this week"
  },
  "options": {
    "057f0ed07faa48b392a4135cbceb5fbd": {
      "price-key": " screwdriver-pack.chrome-plated",
      "init-price": {
        "price": 5,
        "type": "flat"
      },
      "recurring-price": {
        "price": 0,
        "type": "flat"
      },
      "selected" : false,
    },
    "13BBC2AEB89445D5AC51AE8BBD65FED4": {
      "price-key": "screwdriver-pack.numOfTools",
      "init-price": {
        "price": 5,
        "unit-price": 1,
        "type": "per-unit"
      },
      "recurring-price": {
        "price": 0,
        "unit-price": 0,
        "type": "per-unit"
      },
      "selected" : true,
       "value": "5",
      "unit-measurement", "Pieces"
    }
  }
}
```

The price response contains these important elements:

- **price-system-properties** – A set of properties the EPS may want to maintain. CSA ignores the values and just sends them back in subsequent requests related to the same service offering. The properties can help the EPS to track price requests related to a single service order. For example, when the service is modified after some time and the price for the modified service is requested, EPS can link the price request to the price request processed during the initial order. This linkage can influence the modification price to determine whether to keep the price for a selected option fixed over time or adjustable which will depend on the contract between service provider and consumer.
- **currency**, **period** – The EPS decides on currency and recurrence period for recurring prices.
- **total-price** – Total price for the service is computed by EPS.
- **base-price** – Base price as well as option/property prices are prices that CSA wants to display to consumer users, so that they have better experience while customizing the service being ordered. Option/property prices are returned by EPS even for option/property that are not currently selected and those prices are not included into the total price.
- **init-price**, **recurring-price** – Every price entry may have two elements: initial price and recurring price
- **Price type** – Price type can be flat or per-unit. Flat is an absolute value. Per-unit means that the price depends on the number coming in as the property value (e.g. number of tools), available for Integer type Property.
- **message** – Any price entry can have text message associated. This message can explain or advertise the price for a promotion, for example or just say hello.

Complete documentation to the price response format can be found in Appendix A on page 9.

Errors from the EPS

If everything is set up correctly, CSA and the EPS will communicate using the request and response described above. In case of some misconfiguration or failure, the EPS will return an error message instead of price in the response. Let's look at an example error message in case a price key is not recognized by the EPS:

```
{
  "message" : "Price ID screwdriver-pack.numOfTools was not found. \ Organization: BULL_WAR, user: Fred,
catalog-name: LEATHER_GOODIES",
  "error-code" : "404",
  "missing-keys" : ["screwdriver-pack.numOfTools"]
}
```

The price error contains these important elements:

- **message** – A human readable description of the error.
- **error-code** – A machine readable error code.
- **missing-keys** – Details for a missing price keys in case of error 404.

Complete documentation to the price error format and error codes can be found in

Appendix A: Protocol details on page 9.

External pricing system

The EPS is an abstraction of a third-party pricing system. CSA does not claim to support any real EPS product out of the box. An adapter layer must be implemented and put in front of the real system, so that CSA can use the above described communication protocol to interact with the external system.

Sample EPS

The sample EPS is a basic implementation of the EPS abstraction that is shipped along with CSA product. It fulfills the interface defined by CSA and is capable of communication using the well-defined protocol. It can be used to test external pricing in CSA. The sample EPS is not production-ready and its use in real deployments should be avoided. It is not officially supported.

The sample EPS can be found in the CSA home folder:

%CSA_HOME%\extras\external-pricing

It includes the following:

- **eps.war** – Sample EPS web application
- **eps-protocol.jar** – A library that provides the Java interface for communication between CSA and an EPS. Not required for running the sample EPS server.

The installation of the sample EPS is as simple as deploying it into a servlet container. You can use the same JBoss that CSA is running on, for example. Just create a new eps.war folder and unpack the contents of eps.war archive into it, such as:

%CSA_HOME%\jboss-as\standalone\deployments\eps.war

The pricing data is stored in a JSON file:

eps.war\WEB-INF\classes\data\price-list.json

The data can be changed while the server is running and the changes will take effect immediately. Complete documentation for the sample EPS can be found in Appendix B: Sample EPS details on page 13.

Configuring the EPS integration

Whatever EPS implementation is used, the connection must be set up in the CSA configuration file. Note that this configuration is CSA-wide. There can be only one EPS instance configured for one CSA instance. To configure the connection to the EPS, edit the following file:

%CSA_HOME%\jboss-as\standalone\deployments\csa.war\WEB-INF\classes\csa.properties

Add the following lines:

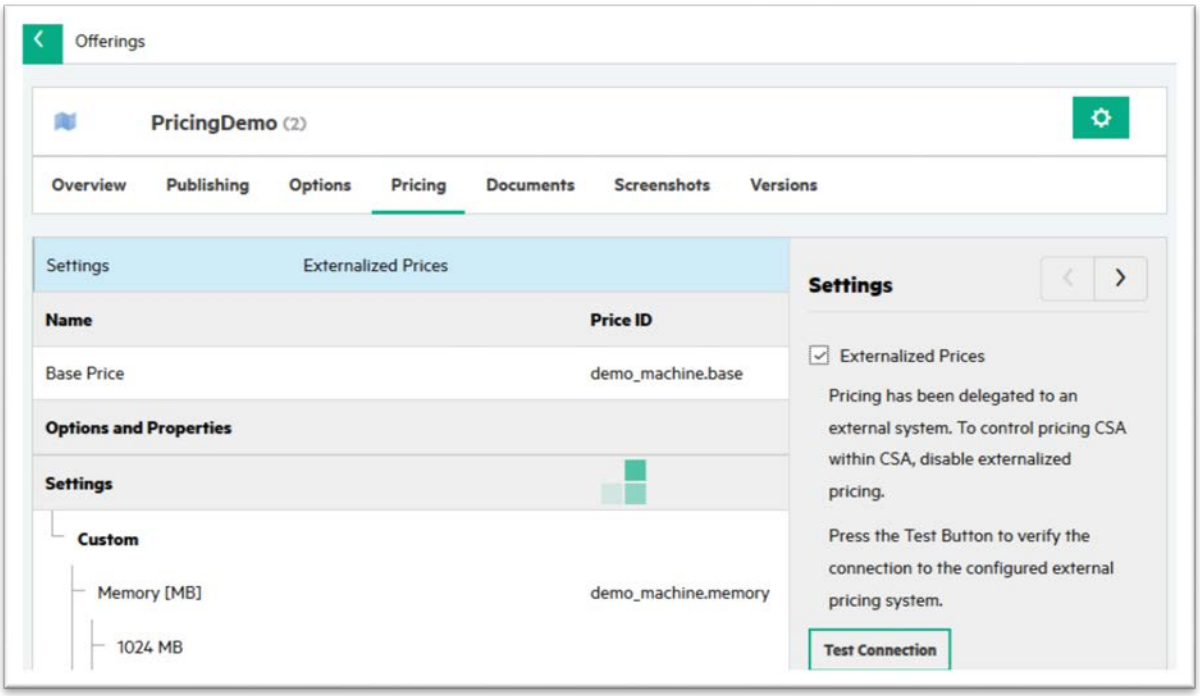
```
## External Pricing System (EPS) configuration
# Turn on/off EPS configuration
external.pricing.active=true

# URL to the EPS REST endpoint
external.pricing.url=https://localhost:8444/eps/api/pricing/quote

# EPS transport username
external.pricing.username=pricing
# EPS transport user password
external.pricing.password=ENC(zsnj3WHl7F6hwkJEDEX/7g==)

# Connection timeout to EPS (in milliseconds); if not configured default (2000) is used
external.pricing.timeout=2000
```

The value of external.pricing.url must point to an endpoint capable of communication according to the well-defined protocol. HTTP basic authentication is supported and so the username and password is used to authenticate to that endpoint.

Service offering with externalized pricing

Once the EPS is running and configured in CSA, the externalized pricing can be used for service offerings. Note that all service offerings share the single EPS instance when it comes to externalized prices.

The externalized pricing can be turned on or off for each service offering individually. Go to **Settings** and select the checkbox to enable externalized prices, as shown below:



Once externalized pricing is enabled for an offering, **Price ID** for the base price and for the service options and properties need to be entered. Make sure the same **Price IDs** are also defined in the EPS. When the service offering is ready and is published, CSA communicates with the EPS and validates that all used IDs can be recognized by the EPS.

When setting price ID for a list property (both static and dynamic), note that there is only one ID for the property independent on the number of potential values the property can have. Different values can still have different prices, since the property value is sent to the EPS along with the price ID.

Published offerings can be consumed by users through Marketplace Portal in similar fashion as before. The difference is that the prices are loaded from the EPS this time.

Limitations

The externalized pricing works in CSA with the following limitations:

Changing the EPS requires CSA restart

The EPS is configured in a configuration file. Any change made to the configuration file requires CSA restart to take effect.

Price is cached at publish time

Price for a service is loaded from the EPS when a consumer user views the service checkout page. However, when viewing the list of services on the catalog browse page, cached prices are shown instead. It would be expensive to fetch prices for all listed services from the EPS. The sorting by price is even more challenging. To work around that, prices are cached into CSA database at the service publishing time. The price is based on the used catalog and default option selection. The price is frozen in time, any price adjustment made to the EPS is not propagated.

Price multiplication by quantity is not supported

In CSA, you can promote an integer property to be a quantity multiplier so that the quantity multiplier value multiplies the price. Currently, there is no way to choose a quantity multiplier when externalized pricing is used. The EPS calculates the price on its own and there is no way to send a property that would multiply the price.

Appendix A: Protocol details

Request format

```
{
  "protocol-version": 1,
  "price-system-properties": {
    "price-list-id": "21321"
  },
  "user": "john",
  "organization": "ACME_INC",
  "catalog-name": "HARDWARE_TOOLS",
  "requested-date": "2010-01-01T12:00:00.100Z",
  "supported-periods": ["year", "month", "week", "day", "hour", "minute"],
  "base-price-key": "screwdriver-pack",
  "options": {
    "13BBC2AEB89445D5AC51AE8BBD65FED4": {
      "price-key": "screwdriver-pack.numOfTools",
      "value": "5",
      "unit-measurement" : "Pieces",
      "selected": true
    }
  }
}
```

Element	Description	
protocol-version	Version of protocol to be used. Currently, we introduce the first version of protocol. In case of incompatible changes needed, a new version will be created.	
price-system-properties	Optional section. Dedicated to the EPS, so that it can store custom values. First request coming from CSA does not contain this element. When it is received in the response from the EPS, CSA returns the values in the subsequent request.	
user	Username of the user who requests to see the price of a service.	
organization	Name of the organization the user belongs to.	
catalog-name	Name of the CSA catalog used for the offering.	
requested-date	Date for which the pricing information is requested.	
supported-periods	List of recurring periods supported by CSA. The EPS can use any of the listed recurring periods when returning the pricing information.	
base-price-key	ID of the base price record in the EPS.	
options	Service options/properties can be priced as well. List of elements with price IDs. The EPS is requested to provide the prices for these elements.	
	Nested Elements	
	<id>	CSA specific identifier necessary for backward price mapping.
	price-key	ID of the price record in the EPS.
	selected	Identifies if price for this option/property should be computed into total price (true) or not (false). Even if it is set to false, the option/property price alone is returned in response.

	value	Optional. Used for measurable properties (e.g. CPU count, RAM size, etc.) and list items. The value influence the price.
	unit-measurement	Measurement unit used with the property value. Defined in CSA and passed in the request to the EPS. E.g. GB, MB, etc.

Response format

```
{
  "protocol-version": 1,
  "price-system-properties": {
    "price-list-id": "21321"
  },
  "currency": "USD",
  "period": "month",
  "total-price": {
    "init-price": 20,
    "recurring-price": 0,
    "message": "Order now!"
  },
  "base-price": {
    "price-key": " screwdriver-pack",
    "init-price": 20,
    "recurring-price": 0,
    "message": "20% discount this week"
  },
  "options": {
    "057f0ed07faa48b392a4135cbceb5fbd": {
      "price-key": " screwdriver-pack.chrome-plated",
      "init-price": {
        "price": 5,
        "type": "flat"
      },
      "recurring-price": {
        "price": 0,
        "type": "flat"
      },
      "selected" : false,
    },
    "13BBC2AEB89445D5AC51AE8BBD65FED4": {
      "price-key": "screwdriver-pack.numOfTools",
      "init-price": {
        "price": 5,
        "unit-price": 1,
        "type": "per-unit"
      },
      "recurring-price": {
        "price": 0,
        "unit-price": 0,
        "type": "per-unit"
      },
      "selected" : true,
       "value": "5",
      "unit-measurement", "Pieces"
    }
  }
}
```

Element	Description		
protocol-version	Version of protocol to be used. Currently, we introduce the first version of protocol. In case of incompatible changes needed, a new version will be created.		
message	Optional message coming from the EPS. A message can be put on top level or within every price object: base-price, total-price, init-price, and recurring-price.		
price-system-properties	Optional section. Dedicated to the EPS, so that it can store custom values. First request coming from CSA does not contain this element. When it is received in the response from the EPS, CSA returns the values in the subsequent request.		
	First request doesn't contain price-system-properties		
	First response contains this section with information the EPS want to store.		
	Subsequent requests keeps previously stored data.		
currency	Currency used for all the prices.		
period	Time period used for recurring prices.		
base-price	Object containing information about the service base price.		
	Nested Elements		
	price-key	ID of the price record in the EPS.	
	init-price	Initial price value.	
	recurring-price	Recurring price value. Related to the period.	
total-price	Total price for the service calculated by the EPS based on selected options, property values and requester's context.		
	Nested Elements		
	init-price	Initial price value.	
	recurring-price	Recurring price value. Related to the period.	
options	Service options/properties can be priced as well. List of elements with price information. The EPS provides the prices for these elements.		
	Nested Elements		
	<id>	CSA specific identifier necessary for backward price mapping.	
	price-key	ID of the price record in the EPS.	
	init-price	Initial price value.	
		Nested Elements	
		price	Price.
		type	Model used to compute a price for an integer property value. Can be flat or per-unit. Flat price is the same irrespective of the property value. Per-unit price depends on the property value.
		unit-price	In case of per-unit pricing model, unit-price is the price for a single unit.

	recurring-price	Recurring price value. Related to the period.	
		Nested Elements	
		price	Price.
		type	Model used to compute a price for an integer property value. Can be flat or per-unit. Flat price is the same irrespective of the property value. Per-unit price depends on the property value.
		unit-price	In case of per-unit pricing model, unit-price is the price for a single unit.
	value	The value received in the request is sent back in the response. The value can influence the price.	
	selected	The value received in the request is sent back in the response. Identifies if price for this option/property should be computed into total price (true) or not (false).	
	unit-measurement	Units used for integer property values. E.g. GB, CPU, Node	

Error codes

CSA expects to receive standard error codes from the EPS when something goes wrong. These codes include the following:

Error #		Description
401	Unauthorized	Incorrect credentials
404	Not Found	Requested price for a price ID could not be found
500	Internal Error	EPS internal error
503	Not Available	Indicates that the EPS is not running

Appendix B: Sample EPS details

The sample EPS is a web application exposing an API on the following URL. HTTP basic authentication is required.

Endpoint	https://<host>:<port>/eps/api/pricing/quote
Username	pricing
Password	csa

The pricing data for the sample system are loaded from a JSON file located inside the WAR file:

`eps.war\WEB-INF\classes\data\price-list.json`

Inside the WAR file, an example JSON file can be found. The file structure is as follows:

organizations	List of organization. Each organization has its own price lists.			
	Nested Elements			
	organization-name	Name of the Organization. Note that CSA sends the internal name of the organization in the price request. An internal name is usually in UPPERCASE and using underscore_instead_of_space.		
	price-lists	Each organization has its own price lists. Each price list contains detailed information about prices.		
		Nested Elements		
		price-list-id	Price list ID.	
		user	A price list can be defined for a specific user. It is optional and probably not very common.	
		valid-from	Timestamp of the price list validity start.	
			Format: 2015-12-31T11:59:59.999Z	
		valid-to	Timestamp of the price list validity end.	
			Format: 2015-12-31T11:59:59.999Z	
		currency	Currency used with prices defined within this price list.	
		period	Time period used with recurring prices defined within this price list. CSA supports these values: *year, month, week, day, hour, minute*	
		prices	List of prices per price keys.	
			Nested Elements	
			price-key	ID of the price record in EPS.
			init-price-type	Model used to compute an initial price for an integer property value. Can be flat or per-unit. Flat price is same independently on the property value. Per-unit price depends on the property value.

			recurring-price-type	Model used to compute a recurring price for an integer property value. Can be flat or per-unit. Flat price is same independently on the property value. Per-unit price depends on the property value.	
			init-price	Initial price.	
			recurring-price	Recurring price.	
			value-prices	List properties in CSA have usually multiple values. Each value can have a different price assigned. Instead of defining a single *init-price* and *recurring-price*, list of prices can be defined for one *price-key*. The prices are stored under the property value as an additional key.	
				Nested Elements	
				<value>	Property value. Servers as an additional key for the price assigned to the property value.
				init-price	Initial price.
				recurring-price	Recurring price.

Send documentation feedback

If you have comments about this document, you can send them to clouddocs@hpe.com.

Legal notices

Warranty

The only warranties for Hewlett Packard Enterprise products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Hewlett Packard Enterprise shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Restricted rights legend

Confidential computer software. Valid license from Hewlett Packard Enterprise required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright notice

Trademark notices

Adobe® is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

RED HAT READY™ Logo and RED HAT CERTIFIED PARTNER™ Logo are trademarks of Red Hat, Inc.

The OpenStack word mark and the Square O Design, together or apart, are trademarks or registered trademarks of OpenStack Foundation in the United States and other countries, and are used with the OpenStack Foundation's permission.

Documentation updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to the following URL and sign-in or register: https://softwaresupport.hp.com.

Select Manuals from the Dashboard menu to view all available documentation. Use the search and filter functions to find documentation, whitepapers, and other information sources.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your Hewlett Packard Enterprise sales representative for details.

Support

Visit the Hewlett Packard Enterprise Software Support Online web site at https://softwaresupport.hp.com.