Oracle Storage Connect plug-in Development Guide

General

Revision 1.2.8-BETA

SPECIFICATION, DEVELOPMENT AND DISTRIBUTION LICENSE AGREEMENT

"We," "us," and "our" refers to Oracle USA, Inc., for and on behalf of itself and its subsidiaries and affiliates under common control. "You" and "your" refers to the individual or entity that wishes to use materials from Oracle. "Programs" refers to the software product you wish to download and use and program documentation. "Specification" refers to the specification associated with the Programs that you wish to download and use. "Redistributables" refers to the materials associated with the program that are also identified in the Program documentation as redistributable. "Licensed Materials" refers to the Programs, Specification, and/or Redistributables. "License" refers to your rights to use the Licensed Materials under the terms of this agreement. This agreement is governed by the substantive and procedural laws of California. "Sample Code" refers to modules identified as sample code and which are licensed under the open source license included in their header files. You and Oracle agree to submit to the exclusive jurisdiction of, and venue in, the courts of San Francisco or Santa Clara counties in California in any dispute arising out of or relating to this agreement.

We are willing to license the Licensed Materials to you only upon the condition that you accept all of the terms contained in this agreement. Read the terms carefully and select the "Accept" button at the bottom of the page to confirm your acceptance. If you are not willing to be bound by these terms, select the "Do Not Accept" button and the registration process will not continue.

License Rights

We grant you a nonexclusive, nontransferable limited license to use the Programs and Specifications in unmodified form for purposes of developing your plug-in applications that interface with the Programs.

We grant you a nonexclusive, nontransferable right to copy and distribute the Redistributables in unmodified form with your plug-in application solely for the purpose of allowing your plug-in application to interface with the Program. Prior to distributing the Redistributables you shall require your end users to execute an agreement binding them to terms consistent with those contained in this section and the sections of this agreement entitled "License Rights," "Ownership and Restrictions," "Export," "Disclaimer of Warranties and Exclusive Remedies," "No Technical Support," "End of Agreement," and "Relationship Between the Parties." You must also include a provision specifying us as a third party beneficiary of the agreement. You are responsible for obtaining these agreements with your end users. We may audit your use of the Licensed Materials. Documentation is either shipped with the Programs, or documentation may be accessed online at http://otn.oracle.com/docs.

If you want to use the Licensed Materials for any purpose other than as expressly permitted under this agreement you must contact us to obtain the appropriate license.

Each Sample Code module is licensed under the terms and conditions of the open source license included with that module.

Ownership and Restrictions

We retain all ownership and intellectual property rights in the Licensed Materials. You may make a sufficient number of copies of the Licensed Materials for the licensed use and one copy of the Program for backup purposes.

You may not:

- use the Licensed Materials for any purpose other than as provided above;

- distribute the Redistributables unless accompanied with your plug-in applications;

- remove or modify any markings or any notice of our proprietary rights that may appear in any of the Licensed Materials;

- use the Licensed Materials to provide third party training on the content and/or functionality of the Programs, except for training your licensed users;

- assign this agreement or give the Licensed Materials, Program access or an interest in the Licensed Materials to any individual or entity except as provided under this agreement;

- cause or permit reverse engineering (unless required by law for interoperability), disassembly or decompilation of the Programs;

- disclose results of any Program benchmark tests without our prior consent; or,

- use any Oracle name, trademark or logo.

Indemnification

You agree to: (a) defend and indemnify us against all claims and damages caused by your distribution of the programs in breach of this agreements and/or failure to include the required contractual provisions in your end user agreement as stated above; (b) keep executed end user agreements and records of end user information including name, address, date of distribution and identity of programs distributed; (c) allow us to inspect your end user agreements and records upon request; and, (d) enforce the terms of your end user agreements so as to effect a timely cure of any end user breach, and to notify us of any breach of the terms.

Export

You agree that U.S. export control laws and other applicable export and import laws govern your use of the programs, including technical data; additional information can be found on Oracle's Global Trade Compliance web site located at http://www.oracle.com/products/export/index.html?content.html. You agree that neither the programs nor any direct product thereof will be exported, directly, or indirectly, in

violation of these laws, or will be used for any purpose prohibited by these laws including, without limitation, nuclear, chemical, or biological weapons proliferation.

Disclaimer of Warranty and Exclusive Remedies

THE LICENSED MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. WE FURTHER DISCLAIM ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT.

IN NO EVENT SHALL WE BE LIABLE FOR ANY INDIRECT, INCIDENTAL, SPECIAL, PUNITIVE OR CONSEQUENTIAL DAMAGES, OR DAMAGES FOR LOSS OF PROFITS, REVENUE, DATA OR DATA USE, INCURRED BY YOU OR ANY THIRD PARTY, WHETHER IN AN ACTION IN CONTRACT OR TORT, EVEN IF WE HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. OUR ENTIRE LIABILITY FOR DAMAGES HEREUNDER SHALL IN NO EVENT EXCEED ONE THOUSAND DOLLARS (U.S. $1,000).

No Technical Support

Our technical support organization will not provide technical support, phone support, or updates to you for the programs licensed under this agreement.

Restricted Rights

If you distribute a license to the United States government, the Licensed Materials, including documentation, shall be considered commercial computer software and you will place a legend, in addition to applicable copyright notices, on the documentation, and on the media label, substantially similar to the following:

NOTICE OF RESTRICTED RIGHTS

"Programs delivered subject to the DOD FAR Supplement are 'commercial computer software' and use, duplication, and disclosure of the programs, including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, programs delivered subject to the Federal Acquisition Regulations are 'restricted computer software' and use, duplication, and disclosure of the programs, including documentation, shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software-Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065."

End of Agreement

You may terminate this agreement by destroying all copies of the programs. We have the right to terminate your right to use the Licensed Materials if you fail to comply with any of the terms of this agreement, in which case you shall destroy all copies of the programs.

Relationship Between the Parties

The relationship between you and us is that of licensee/licensor. Neither party will represent that it has any authority to assume or create any obligation, express or implied, on behalf of the other party, nor to represent the other party as agent, employee, franchisee, or in any other capacity. Nothing in this agreement shall be construed to limit either party's right to independently develop or distribute software that is functionally similar to the other party's products, so long as proprietary information of the other party is not included in such software.

Entire Agreement

You agree that this agreement is the complete agreement for the Licensed Materials, and this agreement supersedes all prior or contemporaneous agreements or representations. If any term of this agreement is found to be invalid or unenforceable, the remaining provisions will remain effective.

Executive Summary

The Oracle Storage Connect framework provides a storage discovery and provisioning Application Programming Interface (API) that is intended to greatly enhance the ease with which storage can be managed and provisioned in an Oracle VM environment. The combined solutions will allow customers to provision and manage participating storage platforms through Oracle VM Manager and Oracle Enterprise Manager, simplifying virtual infrastructure management, and delivering faster virtual machine configuration and control. Public and private cloud infrastructures using virtualized compute and storage services will also benefit from accelerated provisioning and simpler, integrated management.

Structure of the documents

The development guide for the API is broken up into three different documents, namely the *General* (this document), *Storage Array* and the *File System* documents. The General document gives information regarding both types of plug-ins while the other two documents focuses on one specific type of plug-in as described in their respective titles.

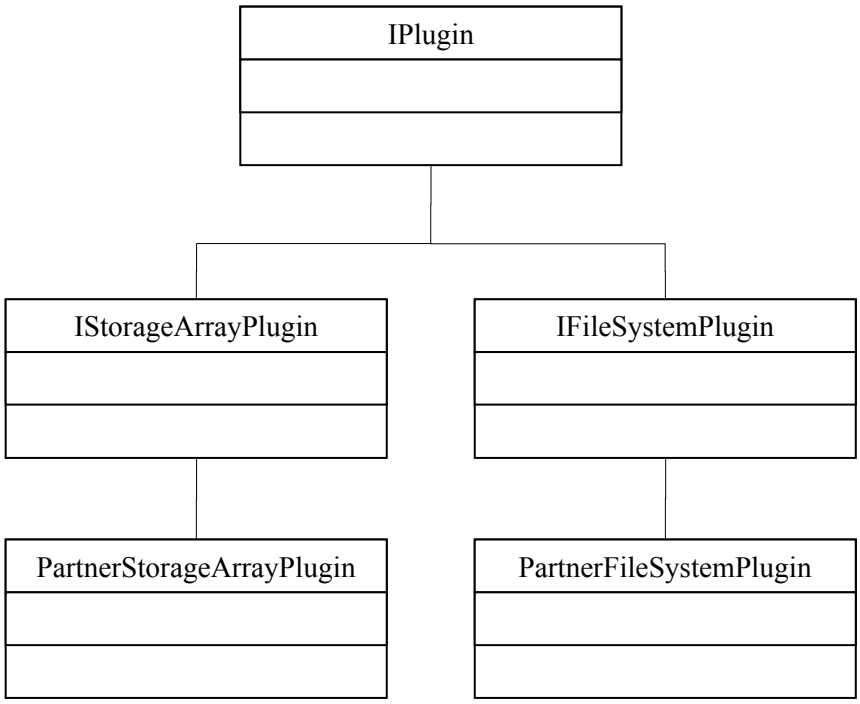
The Storage Connect API

The Oracle Storage Connect API (hereafter just API) is Python based and as such, the top level of all the plug-ins must be implemented in Python. There is no requirement that the full plug-in be implemented in Python. It is completely supported to have the top level Python call out to a utility or library implemented in a different language to perform the action requested via the plug-in method.

The architecture of the API is explained in more detail in the next section but essentially it consists of three predefined classes, all plug-ins must inherit from either the `IStorageArrayPlugin` or `IFileSystemPlugin` class. Since a plug-in can be executed on any dom0 or utility server at any time, all plug-ins must be stateless, a plug-in can under no circumstance keep state between calls, to this effect all class methods in the API have been decorated as `@staticmethod` and the top level class explicitly denies instantiating any class inheriting from it.

Architecture

The Storage Connect API consists of on base class and two specializing classes. A plug-in will inherit from one of the two specializing classes depending on the type of plug-in. The class diagram is shown below (for simplicity, all the attributes and methods are hidden):



Implementation

Developing a plug-in consists of sub-classing either the `IStorageArrayPlugin` (iSCSI or SAN) or `IFileSystemPlugin` (NFS) classes. The term SAN, as far as a plug-in is concerned, refers to any type of SCSI block device protocol except iSCSI, for example it can refer to SAS, Fibre Channel, shared SCSI etc. The plug-in must implement all the methods marked as **REQUIRED**, see the *Storage Array* and/or the *File System* documents for more information. Plug-ins should not create any files on the server to store information also temporary files can be requested via the `IPlugin` class only. Please note that the plug-in executes as the `root` user on the server, so it is essential to make sure that the plug-in does not open any security holes.

Packaging for deployment

All plug-ins are distributed as RPM packages (Appendix B contains a sample RPM `.spec` file that can be used for the plug-in with minimal changes). The top-level installation directory for the plug-in will be:

```
/opt/storage-connect/plugins/vendor/<plugin-module>
```

The plug-in should be packaged as a Python module, so it should include a `__init__.py` file at the plug-in module level. The `__init__.py` file must set the `osc_<type>_plugins` module variable(s) to the names of the plug-in classes, where `<type>` is either `ifs`, indicating the class

implements either the File System API or `isa`, indicating the class implements the Storage Array API, for example:

```
osc_isa_plugins=["oracle.gensa.OracleGenSCSI.GenericSCSI"]
```

This allows a single plug-in module to contain multiple Storage Connect plug-ins classes to enable sharing of common modules etc. between plug-ins from the same vendor. Also, note that a single plug-in module can contain both Storage Array plug-in classes and File System plug-in classes as shown in the example below:

```
osc_isa_plugins=["oracle.gensa.OracleGenSCSI.GenericSCSI",
                 "oracle.gensa.StorageServer.StorageServer"]
osc_ifs_plugins=["oracle.gensa.OCFS2.OCFS2",
                 "oracle.gensa.NFS.GenericNFS"]
```

If the plug-in depends on Java or some other external technology that the plug-in can use the RPM dependency so the dependent, RPMs will be installed when the plug-in is installed as well. Oracle will evaluate all the dependencies on a case-by-case basis to make sure the dependencies would be able to be accommodated inside the Oracle VM Server environment. If the plug-in depends on technology that cannot be accommodated in an Oracle VM Server environment, the plug-in can use a split implementation where only a very few calls is required to be able to run on the Oracle VM Server directly. By enabling the `split_plugin` option in the RPM spec file, two RPMs would be generated, one for the actual plug-in that would be able to install anywhere as well as a Meta RPM named `osc-requirer-vendor-plugin`. The Meta plug-in would only be able to install on a system where all the dependencies can be met. Also as part of the split, the plug-in should call the utility function `checkRequired(…)` at the start of each API that requires access to the commands that is pulled in via the Meta RPM.

NOTE: ALL the plug-ins within the plug-in module MUST have the SAME version and it MUST exactly match the RPM version in the `.spec` file.

Licensing

There is no requirement for the plug-in to be release under the Open Source (GPL) or any other specific license, to this end the Oracle Storage Connect infrastructure is dual licensed under the Open Source (GPL) and the Oracle Specification, Development and Distribution licenses to allow for both Open Source (GPL) and propriety licensed Storage Connect plug-ins.

Support and Maintenance

If the plug-in were released under the Open Source (GPL) license, Oracle would be able to distribute (if Oracle so choose) and support the plug-in and feedback any changes (bug fixes) back to the vendor for inclusion in updates from the vendor. If Oracle support is unable to resolve the problem Oracle support will contact the plug-in vendor to assist in supporting the plug-in. Closed Source (non GPL) plug-in support will follow the normal Oracle Support flow as used today and the plug-in must be made available to customers by the vendor. The vendor, no matter what license the plug-in is released under, will maintain the plug-in code supplied by the vendor.

Terminology

Throughout this document and specification, we will be referring to these terms. It is extremely important to use consistent and clear terminology throughout all the plug-in implementations, which is why we define these up front. We understand that these may not match the terminology used by each partner. It should be easy to map these to whatever terms are in use for the specific storage device. It is not necessary to use these terms when talking to customers etc., they are just used internally by the product and the plug-in

Storage Array	Any block based storage device, be it Fibre Channel (SAN), Ethernet (iSCSI), or direct connect (SCSI/SAS/DAS) based.
Storage Element	This term refers to any type of storage that the Storage Array support, for example: LUNs, Snapshots, Clones etc.
NAS	Any network file system based storage (NFS).
Storage Server	Refer to either a Storage Array or NAS depending on the type of plug-in.
Snapshot	Frozen thin provisioned **read only** point in time copy of a LUN (or file). This implies that a Snapshot remains a child of the original LUN (or file).
Clone	Thin provisioned **read write** copy of a LUN (or file). There is an implication that the possibility exist to determine from which LUN this is a *clone*.
Snap Clone	Thin provisioned **read write** copy of a previously created *Snapshot*. This implies that the parent child relationship for the Snapshot stays intact.
Split Clone	This refers to a deep copy of a previously created *clone*. This implies that the LUN is now a top level object and have no parents (it can become a parent when a Snapshot operation is triggered though)
Volume Group	LUNs, file systems etc. are carved from this *optional* grouping construct.
Access Group	Access groups are used to limit the server that can interact ('see') a specific LUN or mount a file system. For LUNS this would normally be a list of iSCSI initiators and for Fibre Channel a list of HBA port WWNs. For NFS this is normally the list of hosts the file system is exported to.

Some best practices for developing the plug-in

Even though there is no hard and fast rules on developing a Storage Connect plug-in there are still some guidelines to keep in mind when developing the plug-in.

- Use of temporary files
  The plug-in should use as few as possible temporary files. It is preferable if the plug-in does not use any temporary files. In "Appendix A" a few examples are listed in ways to still call out to other tools etc. without requiring temporary files. If a temporary file is required (maybe if an external tool require a file for output), the temporary file must be obtained via the `IPlugin` class, please see "Using a temporary file in a plug-in".
- Calling an external script
  It is preferable to implement what the external script does directly inside the Python plug-in classes if possible. Calling out to a script that essentially just calls an executable and run *sed* on its output or something similar, should be avoided. This is not intended to indicate that the plug-

in are not allowed calling out to a script. The point here is just when calling out to a script; the script should add real value and not just be a wrapper script for another executable.

- Python indentation
  Python is very sensitive to indentation. For this reason, we strongly suggest to **only** use spaces for indentation and not tab characters.

Logging

The plug-in is expected to log any problems using the standard Python logging facility. The plug-ins is supplied with a logging class which they can reference using the `IPlugin.logger` class variable. This is a preconfigured `logging.getLogger` instance. Plug-ins should NOT change any of the logging options for the instance (they are pre-configured) and should never print anything to standard output or try to read anything from standard input.

Caching

The plug-in can (and is strongly suggested) to cache the raw data from the Storage Server. In general the connection and authentication to the Storage Server is slow and very heavy. The idea is to obtain as much information as possible in a single connection and store all of it in the cache. On subsequent calls, the information can most likely be satisfied using the cache. The caching mechanism is implemented using two plugin manager calls, `IPlugin.cache.set()` and `IPlugin.cache.get()`.

IPlugin.cache.set()

Parameters:

Name	Type	Optional	Description
`ss_uuid`	UUID	No	The UUID of the Storage Server this specific key is associated with.
`cache_key`	String	No	Name (key) of cache. This can be any valid string value.
`cache_value`	String	No	This is the data to be cached. The value must be a string but it is completely valid to use `cPickle.dumps()` (ONLY protocol `0` are allowed to be used when pickling up the data) and `cPickle.loads()` to serialize/de-serialize Python objects.
`expire_time`	Integer	Yes	Expire time in seconds for this particular cache entry. If not supplied, the default (30 seconds) will be used.

Return value:

N/A.

Use this method to store any data that might be useful in the future to speed up any subsequent plugin calls. To clear the cache for this specific key just set the `cache_value` to `None`.

IPlugin.cache.get()

Parameters:

Name	Type	Optional	Description
`ss_uuid`	UUID	No	The UUID of the Storage Server this specific key is associated with.
`cache_key`	String	No	Name (key) of cached value being queried.

Return value:

String containing the previously stored cache data or `None` if the cache expired or does not exist.

This method is used to retrieve the previously cached data. If the cached data already expired or does not exist, it returns `None`.

IPlugin.cache.extend()

Parameters:

Name	Type	Optional	Description
`ss_uuid`	UUID	No	The UUID of the Storage Server this specific key is associated with.
`cache_key`	String	No	Name (key) of the cached value being extended.
`extend_time`	Integer	No	Time in seconds to extend the cached value's expire time.

Return value:

N/A.

If for some reason the plugin would like to make available longer, it can use this call to add time (in seconds) to the expiration timer for the cache.

IPlugin.cache.clear()

Parameters:

Name	Type	Optional	Description
`ss_uuid`	UUID	No	The UUID of the Storage Server being removed from the cache.

Return value:

N/A.

This will clear ALL the cache key and values associated with the Storage Server supplied.

Exceptions

Methods do not have failure returns. All error conditions have to be signaled via raising an exception. There are a number of exceptions classes pre-defined, plug-ins are not allowed to define more, if there is a need for a specific class that is not pre-defined, please let us know and so we can consider adding it to the official specification. Below is the list of currently defined exception classes (Note: the plug-in should never raise the `IPluginException` class directly).

IPluginException

This is the base class for all the other `IPlugin` exception classes.

NoSuchOperationEx

If the plug-in or the Storage Server does not support the specific operation, the plug-in should raise this exception.

OperationFailedEx

This is the exception of last resort, if no other exception class fit the specific error situation the plug-in is trying to convey, this exception can be raised using a descriptive message.

MissingKeyEx

If the plug-in encounter a situation where it except a certain key to be present in one for the records passed in and it is not, it should raise this exception. Note this exception takes both a message and the key name that is missing.

ValueFormatEx

If a value for a key in one of the records is invalid (this is not the same as the key being missing), the plug-in should raise this exception. Note this exception also takes a message as well as the name of the key of which the data is invalid.

StorageArrayLicenseEx

If the plug-in is being requested to invoke a licensed feature on the Storage Server and the license is not installed on the Storage Server, the plug-in should raise this exception. For example if the plug-in is requested to validate a new Storage Server record with the storage type being set to iSCSI but iSCSI is not licensed on the Storage Server, the plug-in would raise the exception with the message indicating that the Storage Server is not licensed for iSCSI access.

TargetDiscoveryEx

For an iSCSI Storage Server, if the target discovery fails, the plug-in should raise this exception with the error encountered discovering the targets.

LoginFailedEx

For an iSCSI Storage Server, if the logging into the targets fail, the plug-in should raise this exception with the error encountered trying to login to the targets.

LogoutFailedEx

For an iSCSI Storage Server, if the logging out of the targets fails, the plug-in should raise this exception with the error encountered trying to logout of the targets.

RefreshFailedEx

If the plug-in is unable to refresh the SCSI bus for the specific Storage Server, the plug-in should raise this exception with the message indicating what the error the plug-in encountered.

ListFailedEx

If the plug-in run into errors trying to list (using the `list` method) or update the Storage Element records (using the `updateSERecords` method), the plug-in should raise this exception.

CreateSnapFailedEx

If the plug-in is unable to create a snapshot for the Storage Element, the plug-in should raise this exception with the message indicating the specific problem.

ListSnapFailedEx

This exception is intended to be raised by the plug-in is unable to retrieve a list of the existing snapshots on the Storage Server.

SnapRestoreNotSafeEx

When the plug-in is requested to roll back a snapshot for a Storage Element and it cannot be safely done without affecting other Storage Elements etc., the plug-in must raise this exception and not attempt rolling back the snapshot.

CloneFailedEx

If the plug-in encounter an error when requested to create a clone (either via the `clone` or `cloneFromSnap` methods), the plug-in should raise this exception.

InvalidStorageArrayEx

The Storage Server record specified failed validation. This exception is only allowed to be raised by the `validate` method.

InvalidValueEx

If the plug-in detect an invalid value in any for the records (including the `extra_info` fields), the plug-in should raise this exception.

StorageElementBusyEx

If the Storage Server does not support the operation being performed while the Storage Element is online or mapped, the plug-in should raise this exception.

PermissionDeniedEx

If the plug-in is not authorized to perfume the specific action requested, it should raise this exception with the message indicating the exact permission error returned by the Storage Server.

OperationPreReqNotMetEx

The plug-in raise this exception if a pre-requisite is not met for the requested operation.

StorageNameRequiredEx

This exception can only be raised in the `validate` call by the plug-in if the plug-in requires the `storage_name` to be set in the Storage Server record.

InvalidFSTypeEx

The File System type is not valid for the particular plug-in. This exception is only allowed to be raised by the `validate` method.

FileSystemBusyEx

If the Storage Server does not support the operation being performed while the File System is online or mounted, the plug-in should raise this exception.

FileSystemAlreadyMountedEx

If the File System is already mounted, the plug-in should raise this exception on any subsequent mount requests.

Enum definitions

The API defines a pre-defined enumeration type that are used by the ability `dict` to indicate what specific operations the plug-in and the Storage Server can support.

ABILITY_TYPES enumeration

The ABILITY_TYPES enumeration contains the following identifiers:

Identifier	Description
`UNSUPPORTED`	If the plug-in (or the Storage Server) do not support a particular feature the value for the ability key should be set to this.
`OFFLINE`	If the plug-in (or the Storage Server) can only perform the specific operation while the target (or source) storage entity needs to be offline.
`ONLINE`	If the plug-in (or the Storage Server) can be performed while the storage entity is online and being actively used.
`NO`	If the plug-in (or the Storage Server) does not have a particular feature the value for the ability key should be set to this. NOTE: Only certain keys allow for YES and NO values, they are highlighted in the ability `dict` in the respective documents.
`YES`	If the plug-in (or the Storage Server) does have a particular feature the value for the ability key should be set to this. NOTE: Only certain keys allow for YES and NO values, they are highlighted in the ability `dict` in the respective documents.
`INVALID`	This value should never be used by a plug-in; it is just a placeholder value.

BACKING_DEVICE_TYPES enumeration

The BACKING_DEVICE_TYPES enumeration contains the following identifiers:

Identifier	Description
UNSUPPORTED	This is to indicate that the plugin cannot create a file system.
DEVICE_SINGLE	The File System can only exist / be created on a single block device.
DEVICE_MULTI	The File System can exist / be created on a many block devices.
PLUGIN_SINGLE	The File System is not block based and can only exist / be created on a single backing device. The list of available backing devices will be obtained via the getFileSystemBackingDevices() call.
PLUGIN_MULTI	The File System is not block based and can exist / be created on a multiple backing devices. The list of available backing devices will be obtained via the getFileSystemBackingDevices() call.
INVALID	This value should never be used by a plug-in; it is just a placeholder value.

SE_STATES enumeration

The SE_STATES enumeration contains the following identifiers:

Identifier	Description
UNKNOWN	The Storage Element is in an Unknown state; this effectively indicates an error condition and the Storage Element would be marked as such.
OFFLINE	The Storage Element is in an Offline state and cannot be accessed for normal I/O operations.
BUSY_CREATE	The Storage Element is still being created and is not available for any operations yet.
BUSY_CLONE	The Storage Element is still being cloned, this state is only applicable to the destination of a clone operation.
BUSY_COPY	The Storage Element is still being deep copied (being split from its peer or parent).
BUSY_CHILDREN	The Storage Element is locked busy since it have active children (i.e. it cannot be delete before all the children are either split or released from the parent).
ERROR	The Storage Element is in an error condition and cannot be used or operated on.
ONLINE	The Storage Element is an Online state. This is the **normal** operating state.

FS_STATES enumeration

The FS_STATES enumeration contains the following identifiers:

Identifier	Description
UNKNOWN	The File System is in an Unknown state; this effectively indicates an error condition and the File System would be marked as such.

Identifier	Description
`UNMOUNTED`	The File System is available but not mounted.
`MOUNTED`	The File System is currently mounted. This is the **normal** operating state.
`MOUNTED_READONLY`	The File System is currently mounted in READ ONLY mode.
`ERROR`	The File System is in an error condition and cannot be used or operated on.

Record definitions

The API makes use of a few predefined records (they are implemented using Python `dict` objects), namely the Storage Server record (`ss_record`), the Storage Element record (`se_record`), the File System record (`fs_record`), and the File record (`file_record`). Note that both the `IStorageArrayPlugin` and `IFileSystemPlugin` plug-ins use the Storage Server record.

Storage Server Record

The `ss_record` is used by both `IStorageArrayPlugin` and `IFileSystemPlugin` plug-ins and have the following known fields:

Attribute name	Type	Allow Update	Description	Optional
`name`	String	Only if Empty	User-friendly name of the Storage Server as shown in the Oracle VM Manager. This will be supplied by the user when creating the Storage Server object.	No
`uuid`	UUID	Only if Empty	UUID generated by the Oracle VM Manager when the new Storage Server object is created.	No
`storage_server_id`	String	Yes	Opaque Storage Server unique identifier. This is for use solely by the plug-in to uniquely identify the Storage Server.	Yes
`storage_type`	String	No	For a Storage Server plug-in it will be set to either `IStorageArrayPlugin.SANStorage` or `IStorageArrayPlugin.iSCSIStorage`. For a File System plug-in it will be set to either `IFileSystemPlugin.NetworkFileSystem` or `IFileSystemPlugin.BlockBasedFileSystem`.	No
`access_host`	String	No	If the Storage Server is network based the user will enter the hostname or IP address used to access the storage.	Yes
`access_port`	String	Only if Empty	If the Storage Server is network based this may optionally contain the port number to be used when accessing the storage.	Yes
`username`	String	No	Username to use when logging into the storage server.	Yes

Attribute name	Type	Allow Update	Description	Optional
`passwd`	String	No	Password to use when logging into the storage server.	Yes
`chap`	Bool	No	This will be set if the Storage Server require CHAP authentication.	Yes
`admin_host`	String	No	Hostname / IP address used to administrate the Storage Server.	No
`admin_user`	String	No	Administrator username.	No
`admin_passwd`	String	No	Administrator password.	No
`netdevs`	List	No	Network device(s) that should be used when accessing the Storage Server. This is used in conjunction with the `access_host`. This is to enable forcing iSCSI I/O traffic over a specific network device.	Yes
`status`	String	Yes	Current status of the Storage Server.	No
`total_sz`	Integer	Yes	Total size of the Storage Server (in bytes). (If the Storage Server does not support this, return this as -1 which would translate to 'Not Applicable')	Yes
`used_sz`	Integer	Yes	Actual space already used on the Storage Server (in bytes). (If the Storage Server does not support this, return this as -1 which would translate to 'Not Applicable')	Yes
`free_sz`	Integer	Yes	Available space on the Storage Server (in bytes). (If the Storage Server does not support this, return this as -1 which would translate to 'Not Applicable')	Yes
`alloc_sz`	Integer	Yes	Size of already allocated space on the Storage Server (in bytes). The difference between this and the `used_space` key is that `used_space` does not include any reserved space, for example, if the Storage Server supports sparse allocation and/or de-duplication used space would only return the actual space used and this key would return space that will be consumed when the sparse holes are filled in. (If the Storage Server does not support this, return this as -1 which would translate to 'Not Applicable')	Yes
`access_grps`	List	No	List of `access_grp` `dicts` for the Storage Server.	No

Attribute name	Type	Allow Update	Description	Optional
`vol_groups`	List	Yes	List of `vol_group dicts` known by the Storage Server. This list (if any) is displayed to the user to allow selecting one when a new Storage Element is created.	Yes
`storage_name`	String	No	Actual name of the Storage Server controller as entered or selected by the user when creating the Storage Server. This will uniquely identify the correct controller to operate on if the Storage Server manager supports multiple controllers at the same time.	Yes
`storage_id`	List	Yes	Storage Server identifier (for instance the target name for iSCSI).	Yes
`storage_desc`	String	Yes	Storage Server description.	No
`extra_info`	String	No	Open form opaque information that passed to/from the plug-in for use internally by the plug-in. The Oracle VM product does not look or try to interpret any of the information in this field.	Yes

access_grp dict definition

The `access_grp dict` is defined as follows:

Attribute name	Type	Allow Update	Description	Optional
`grp_name`	String	Yes	This is the name of the access group as defined on the Storage Server. If the Storage Server do not support named access groups the plug-in can ignore the name, if the plug-in have set correct flags in the plug-in ability.	No
`grp_entries`	List	Yes	This is the list of access entries for the access group. For example, for Fibre Channel this would be a list WWNs (using the 0x0000000000000000 format for the port name), for iSCSI a list of initiators and for file systems, this would be a list of the host names.	No
`grp_modes`	List	Yes	An optional list of modes that are associated with the access group, for example in the case of a NFS file system it can contain the export options for the file system export.	Yes

vol_group dict definition

The `vol_group dict` is defined as follows:

Attribute name	Type	Allow Update	Description	Optional
`vol_name`	String	Yes	This is the name of the volume group as defined on the Storage Server.	No
`vol_total_sz`	Integer	Yes	The volume group's total size (in bytes).	No
`vol_used_sz`	Integer	Yes	Actual space already used in the volume group (in bytes).	No
`vol_free_sz`	Integer	Yes	Amount of free space available in the volume group (in bytes).	No
`vol_alloc_sz`	Integer	Yes	Space already allocated in the volume group (in bytes). The difference between this and the `used_space` key is that `used_space` does not include any reserved space, for example, if the Storage Server supports sparse allocation and/or de-duplication used space would only return the actual space used and this key would return space that will be consumed when the sparse holes are filled in.	Yes
`vol_desc`	String	Yes	Description for the volume group, for example `"RAID 5"`. Note: It is up to the plug-in to decide what to fill in here, there is no requirement as to what the field should contain, just something that the user would find useful when selecting which volume group to use when creating a new Storage Element.	Yes

qos_vals dict definition

The `qos_vals dict` is defined as follows:

Attribute name	Type	Allow Update	Description	Optional
`priority`	Integer	Yes	This is priority of the QoS value, zero (0) being the highest priority. It is not required for the values to follow numerically.	No
`value`	String	Yes	This is the Quality-of-Service name as known by the Storage Server.	No
`description`	String	Yes	This is the description of this specific Quality-of-Service.	Yes

Storage Element Record

The `se_record` is only used by `IStorageArrayPlugin` plug-ins and have the following known fields:

Attribute name	Type	Allow Update	Description	Optional
`se_type`	String	Yes	This is required to be always set. This can currently be one of the following values: `IStorageArrayPlugin.LUNType`, `IStorageArrayPlugin.GhostLUNType`, `IStorageArrayPlugin.SnapType`, `IStorageArrayPlugin.SnapCloneType`.	No
`ss_uuid`	UUID	Only if Empty	This is the Storage Server record's UUID. This allows for the persisting the association between any given Storage Element record to the owning Storage Server record.	No
`name`	String	Only if Empty	User-friendly name for the Storage Element as shown in Manager. This can be overwritten by the user in the Oracle VM Manager but the plug-in have to set this to something reasonable.	No
`uuid`	UUID	Yes	The UUID for the LUN (if the storage server supports UUIDs on a LUN basis).	No
`page83_id`	String	Yes	This should be set to the SCSI page 83 unique ID. If possible, the plug-in should return this, it is sometimes impossible to determine this for an un-presented LUN, in which case this key can be omitted from the return list.	No
`id`	String	Yes	Identifier for use internally by the plug-in, for instance the Snap ID or LUN ID.	Yes
`vendor`	String	Yes	The vendor string as reported by the SCSI INQ command.	Yes
`product_id`	String	Yes	The product string as reported by the SCSI INQ command.	Yes
`path`	List	Yes	This is a list of the system device paths for the LUN.	Depends
`array_path`	String	Yes	Path for the Storage Element on the Storage Server for use internally by the plug-in.	Yes
`size`	Integer	Yes	Size in bytes, of the Storage Element, normally returned but supplied for resizing operations.	No
`status`	String	Yes	Status of the Storage Element.	No

Attribute name	Type	Allow Update	Description	Optional
`vol_group_name`	String	Yes	The volume group name (if any) the Storage Element is part of on the Storage Server.	Yes
`access_grp_names`	List	Yes	List of access group names this Storage Element should be, or is presented to.	No
`qos`	String	Yes	The Quality-of-Service value for the Storage Element. NOTE: This is might not reflect the QoS of the Storage Element at all times since this can be set directly on the Storage Server.	Yes
`state`	SE_STATES_enum	Yes	This is the current state of the Storage Element (see SE_STATES enum).	No
`async_handle`	String	Yes	Temporary field that can be set for use by the `getAsyncProgress()` call. Although the value must be a string, it is completely valid to use `cPickle.dumps()` and `cPickle.loads()` to serialize/de-serialize Python objects and store them in the string. The plugin should store any data that will be required later by the `getAsyncProgress()` call to locate the previously started asynchronous operation on the Storage Server. This handle will **not** be persisted! It is for transient use while the asynchronous operation is active to retrieve status and progress on the asynchronous operation only!	Yes
`async_progress`	Integer	Yes	Temporary field that should be set by the `getAsyncProgress()` call only. Valid values are `-1`, `0` to `100` and `None`. A `-1` value indicates the operation is still in progress but plugin is unable to determine the percentage complete. A value between `0` and `100` indicates the percentage complete for the operation. A `None` value indicates that the operation is completed. The `async_progress` value will **not** be persisted! It is for transient use while the asynchronous operation is active to communicate the progress of the asynchronous operation only!	Yes
`extra_info`	String	No	Open form opaque information that passed to/from the plug-in for use internally by the plug-in. The Oracle VM product does not look or try to interpret any of the information in this field.	Yes

File System Record

The `fs_record` is only used by `IFileSystemPlugin` plug-ins and have the following known fields:

Attribute name	Type	Allow Update	Description	Optional
`access_path`	String	Yes	Access path for the file system, only used by networked file systems for example `myserver1:/my/export`.	Yes
`ss_uuid`	UUID	Only if Empty	This is the Storage Server record's UUID. This allows for the persisting the association between any given File System record to the owning Storage Server record.	No
`name`	String	Only if Empty	User-friendly name for the file system as shown in the Oracle VM Manager. The plug-in should set this to something reasonable if empty.	No
`mount_options`	List	Yes	Default options that would be used when mounting a file system.	No
`export_opts`	List	Yes	Options the file system is being exported with on the Storage Server (for example no_root_squash etc.).	No
`array_path`	String	Yes	Path on the file system on the storage server. For use internally by the plug-in.	Yes
`uuid`	UUID	Yes	The UUID of the file system.	No
`size`	Integer	Yes	Size of the file system, normally returned but will be supplied for resizing operations.	No
`free_sz`	Integer	Yes	Free or available size on the file system.	No
`status`	String	Yes	Status of the file system.	No
`state`	FS_STATES_enum	Yes	This is the current state of the File System (see FS_STATES enum).	No
`access_grp_names`	List	Yes	List of access group names the file system should be, or is exported (presented) to.	Yes
`qos`	String	Yes	The Quality-of-Service value for the File System. NOTE: This is might not reflect the QoS of the File System at all times since this can be set directly on the Storage Server.	Yes

Attribute name	Type	Allow Update	Description	Optional
`backing_device`	String / List	No	Depending on the `backing_device_type` ability for the plugin and Storage Server, this would contain a single device (e.g. `/dev/sdc`), a list of devices (e.g. `/dev/sdc`, `/dev/sdd`) or the "`value`" key of the `backing_device` record.	Yes
`async_handle`	String	Yes	Temporary field that can be set for use by the `getAsyncProgress()` call. Although the value must be a string, it is completely valid to use `cPickle.dumps()` and `cPickle.loads()` to serialize/de-serialize Python objects and store them in the string. The plugin should store any data that will be required later by the `getAsyncProgress()` call to locate the previously started asynchronous operation on the Storage Server. This handle will **not** be persisted! It is for transient use while the asynchronous operation is active to retrieve status and progress on the asynchronous operation only!	Yes
`async_progress`	Integer	Yes	Temporary field that should be set by the `getAsyncProgress()` call only. Valid values are `-1`, `0` to `100` and `None`. A `-1` value indicates the operation is still in progress but plugin is unable to determine the percentage complete. A value between `0` and `100` indicates the percentage complete for the operation. A `None` value indicates that the operation is completed. The `async_progress` value will **not** be persisted! It is for transient use while the asynchronous operation is active to communicate the progress of the asynchronous operation only!	Yes
`extra_info`	String	No	Open form opaque information that passed to/from the plug-in for use internally by the plug-in. The Oracle VM product does not look or try to interpret any of the information in this field.	Yes

File Record

The `file_record` is only used by `IFileSystemPlugin` plug-ins and have the following known fields:

Attribute name	Type	Allow Update	Description	Optional
`fr_type`	String	Yes	This is required to be always set. This can currently be one of the following values: `IFileSystemPlugin.FileType`, `IFileSystemPlugin.DirType`, `IFileSystemPlugin.SymLinkType`, `IFileSystemPlugin.CharDevType`, `IFileSystemPlugin.BlkDevType`, `IFileSystemPlugin.FifoType`, `IFileSystemPlugin.SockFileType`, `IFileSystemPlugin.SnapType`, and `IFileSystemPlugin.SnapCloneType`.	No
`file_path`	String	Yes	Path to the file or snapshot to operate on (this can be a relative path under the file system mount path).	No
`ondisk_sz`	Integer	Yes	Size in bytes the file is physically using on disk (i.e. if the file is sparse this size would exclude all the non-allocated holes in the file).	Yes
`file_sz`	Integer	Yes	Size in bytes of the file (i.e. if the file is sparse this size would include all the non allocated holes in the file, in other words this is the apparent size of the file and not the physical use size).	No
`fs_uuid`	UUID	Only if Empty	This is the File System record UUID. This allows for the persisting the association between any given File record to the owning File System record.	No
`name_pattern`	String	Yes	When doing listing files in the file system, this would optionally contain the regular expression to use for filtering out all unwanted files from the listing.	Yes
`shared_sz`	Integer	Yes	This is the amount of bytes the file is currently sharing with another file.	Yes
`snap_name`	String	Yes	Name of the snapshot.	Yes
`dest_path`	String	Yes	Destination path of a symbolic link (this can be a relative path under the file system mount path).	Yes

Attribute name	Type	Allow Update	Description	Optional
async_handle	String	Yes	Temporary field that can be set for use by the `getAsyncProgress()` call. Although the value must be a string, it is completely valid to use `cPickle.dumps()` and `cPickle.loads()` to serialize/de-serialize Python objects and store them in the string. The plugin should store any data that will be required later by the `getAsyncProgress()` call to locate the previously started asynchronous operation on the Storage Server. This handle will **not** be persisted! It is for transient use while the asynchronous operation is active to retrieve status and progress on the asynchronous operation only!	Yes
async_progress	Integer	Yes	Temporary field that should be set by the `getAsyncProgress()` call only. Valid values are `-1`, `0` to `100` and `None`. A `-1` value indicates the operation is still in progress but plugin is unable to determine the percentage complete. A value between `0` and `100` indicates the percentage complete for the operation. A `None` value indicates that the operation is completed. The `async_progress` value will **not** be persisted! It is for transient use while the asynchronous operation is active to communicate the progress of the asynchronous operation only!	Yes
`extra_info`	String	No	Open form opaque information that passed to/from the plug-in for use internally by the plug-in. The Oracle VM product does not look or try to interpret any of the information in this field.	Yes

Mount Record

The `mount_record` is only used by `IFileSystemPlugin` plug-ins and have the following known fields:

Attribute name	Type	Allow Update	Description	Optional
`uuid`	UUID	Only if Empty	This is the mount point record UUID. This allows for the unique identifying a specific mount point on different servers.	No
`fs_uuid`	UUID	Only if Empty	This is the File System record UUID. This allows for persisting the association between the Mount record and the owning File System record.	No
`mount_point`	String	Yes	The absolute path name where the file system is mounted.	No

Attribute name	Type	Allow Update	Description	Optional
`options`	List	Yes	The actual mount options in use on this particular mount point.	
`share_path`	String	Yes	The path appended to the file system exported path that is mounted. For example if the export path is `nfsserver:/export/vmrepos` and `share_path` is set to `poola` then the full path on the mount would be `nfsserver:/export/vmrepos/poola`	Yes
`status`	String	Yes	Display friendly status for this mount point. (Mounted/Read-Only Mounted etc.)	No
`extra_info`	String	No	Open form opaque information that passed to/from the plug-in for use internally by the plug-in. The Oracle VM product does not look or try to interpret any of the information in this field.	Yes

Backing Device

The `backing_device dict` is defined as follows:

Attribute name	Type	Allow Update	Description	Optional
`name`	String	Yes	This is the name of the volume group as defined on the Storage Server.	No
`total_sz`	Integer	Yes	The volume group's total size (in bytes).	No
`free_sz`	Integer	Yes	Amount of free space available in the volume group (in bytes).	No
`value`	String	Yes	This is the value the Storage Server requires to identify / use this specific backing device.	No
`description`	String	Yes	User friendly description for the File System backing device. If not supplied `name` will be used only.	Yes

Appendix A

This following (Appendix A) is Sample Code and is licensed under the following terms and conditions:

NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Calling an external executable without using a temporary file

As can be seen below, instead of sending the output to a temporary file and then reading it back in, the output is send directly to the standard output pipe of the command which is then read directly into the `p_out` string. The returned string is parsed directly with the help of the `StringIO` class. This technique can be used for almost all external executable files as long as the executable support writing the output to a pipe and the amount of output is not too big (around 50K it is probably better to use a file)

```
# This is the SG_IO INQUIRY CDB
get_sg_inq = "12 00 00 00 60 00"

# Use sg_raw to get the LUN info from the device
p_cmd = ("/usr/bin/sg_raw "
         "--binary "
         "--nosense "
         "--request=96 "
         "--timeout=15 "
         "%s " +
         get_sg_inq) % dev_name
p_cmd = p_cmd.split(" ")
sgraw_p = subprocess.Popen(p_cmd,
                           stdout = subprocess.PIPE,
                           stderr = subprocess.PIPE)
(p_out, p_err) = sgraw_p.communicate()

# Succesfull?
if sgraw_p.returncode != 0:
    # Nope, throw an IO Error since we are doing raw scsi
    raise IOError("Unable to issue SG_IO INQ CDB: %s" % p_err)

# Use the StringIO class to read the string as file
sgraw_rdr = StringIO(p_out)

# Start of the expected vendor name
sgraw_rdr.seek(8)

# Read the vendor name
vendor  = sgraw_rdr.read(8)
product = sgraw_rdr.read(16)
version = sgraw_rdr.read(4)

# Update the dict with all the values from the INQ CDB
lun_info["vendor"]  = vendor.strip()
lun_info["product"] = product.strip()
lun_info["version"] = version.strip()
```

Below is a sample of calling an executable that returns the output in XML format and parsing the output inline from the resulting standard output string.

```
# Obtain the size
p_cmd = "sainq %s %s %s getsz %s" % (admin_host,
                                     admin_user,
                                     admin_passwd,
                                     dev_name)
p_cmd = p_cmd.split(" ")
sainq_p = subprocess.Popen(p_cmd,
                           stdout = subprocess.PIPE,
                           stderr = subprocess.PIPE)
(p_out, p_err) = sainq_p.communicate()

# Succesfull?
if sainq_p.returncode != 0:
    raise IOError("Unable to obtain the size: %s" % p_err)

# Add the XML header tags
xml_str = "<Head>\n%s\n</Head>" % p_out

# Create the XML element tree
xml_root = ElementTree.ElementTree(ElementTree.XML(str)).getroot()

# Get the size
for xml_obj in xml_root.findall("object"):
    size = xml_obj.findtext("size")
```

Using a temporary file in a plug-in

```
# Obtain a named temporary file name
temp_file = IPlugin.getNamedTemporaryFile()

# Get the fully qualified name of the temporary file
temp_file_name = temp_file.name

# Use the file ...

# Close the file when done
temp_file.close()

# Make sure it is removed from the system
if os.path.exists(temp_file_name):
      os.remove(temp_file_name)
```

Appendix B

RPM spec file

```
#
# This file osc-oracle-sample.spec is Sample Code and is licensed under the
# following terms and conditions:
#
# Copyright (C) 2008, 2010 Oracle and/or its affiliates. All rights reserved.
#
# Redistribution and use in source and binary forms, with or without
# modification, are permitted provided that the following conditions are met:
#
# 1.  Redistributions of source code must retain the above copyright notice
#     this list of conditions and the following disclaimer.
#
# 2.  The end-user documentation included with the redistribution, if any,
#     must include the following acknowledgment:
#
# "This product includes software developed by Oracle."
#
# Alternately, this acknowledgment may appear in the software itself, if
# and wherever such third-party acknowledgments normally appear.
#
# 3.  The name "Oracle" may not be used to endorse or promote products
#     derived from this software.
#
# 4.  Products derived from this software may not be called "Oracle" nor
#     may "Oracle" or "Ora" appear in their names.
#
# 5.  THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED
#     WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
#     MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
#     IN NO EVENT SHALL ORACLE OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT,
#     INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
#     (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
#     SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
#     HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
#     STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
#     IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
#     POSSIBILITY OF SUCH DAMAGE.
#

#
# Oracle Storage Connect plugin sample .spec file
#

################################################################################
#                                                                              #
# These defines need to be set to reflect the specific plugin                  #
#                                                                              #
################################################################################

%define     vendor_name         vendor
%define     plugin_module       sample
%define     plugin_version      1.0.0
```

```
%define     plugin_release      4
%define     plugin_description  Oracle Storage Connect Sample plugin rpm.
%define     plugin_requires     None

Vendor:     Vendor Name
Summary:    Oracle Storage Connect Sample Plugin
URL:        http://vendor.com/

License:    GPLv2 / Whichever license the plugin should be released under
BuildArch:  noarch

################################################################################
#                                                                              #
# If the plug-in is a split implementation (because some dependency)           #
# set split_plugin to 'yes' and split_requires to any RPM dependencies.        #
#                                                                              #
# Note: When split_plugin is set, two RPMs will be generated, the normal       #
#       osc-vendor-plugin-... one as well as the Meta package named            #
#       osc-requirer-vendor-plugin-...                                         #
#                                                                              #
################################################################################

%define     split_plugin        no
%define     split_requires      None

################################################################################
#                                                                              #
#  The following defines are generic and should not need to be changed         #
#                                                                              #
################################################################################

%define     dist                .el5
%define     build_dir           %{_builddir}/%{name}-%{version}
%define     plugin_dir          /opt/storage-connect/plugins
%define     vendor_dir          %{plugin_dir}/%{vendor_name}
%define     install_dir         %{vendor_dir}/%{plugin_module}

Name:       osc-%{vendor_name}-%{plugin_module}
Version:    %{plugin_version}
Release:    %{plugin_release}%{dist}
Group:      Storage/StorageConnect
Source0:    %{name}-%{version}.tar.bz2
BuildRoot:  %{_tmppath}/%{name}-%{version}
Requires:   osc-plugin-manager >= 1.2.0

%if  %(test X%{plugin_requires} != XNone && echo 1 || echo 0)

Requires:   %{plugin_requires}

%endif


%if %(test X%{split_plugin} == Xyes && echo 1 || echo 0)

%package -n osc-requirer-%{vendor_name}-%{plugin_module}
Vendor:     %{vendor}
Summary:    Meta package for %{name}
URL:        %{url}
License:    %{license}
```

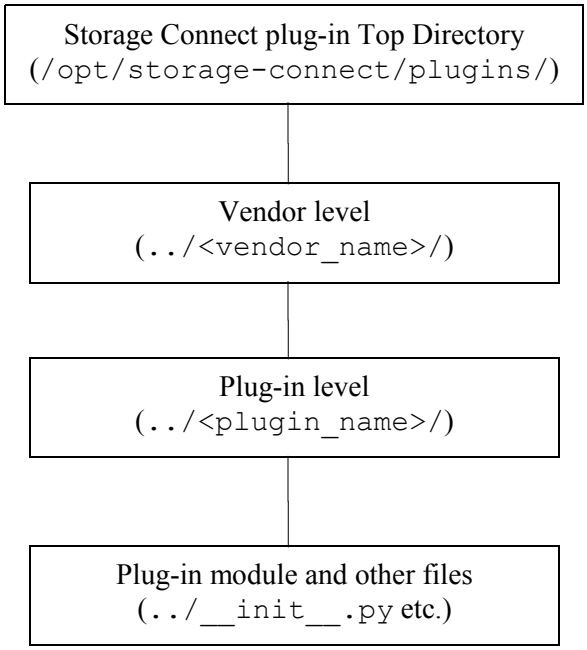
```
Version:    %{plugin_version}
Release:    %{plugin_release}%{dist}
Group:      %{group}
Requires:   %{name} = %{plugin_version}-%{plugin_release}
Requires:   %{split_requires}

%description -n osc-requirer-%{vendor_name}-%{plugin_module}
Requirer package for %{name}

%files -n osc-requirer-%{vendor_name}-%{plugin_module}
%defattr(0755, root, root)

%endif

%description
%{plugin_description}

%prep
%setup -q

%build

%install
rm -fr %{buildroot}
install -d -m0755 %{buildroot}/%{install_dir}
mv %{build_dir}/* %{buildroot}/%{install_dir}/.

%clean
rm -fr %{buildroot}

%files
%defattr(0755, root, root, 0755)
%attr(0755, root, root) /*

%preun
if [ $1 -eq 0 ]
then
    OSC_BIN=/opt/storage-connect/bin
    if [ -x "$OSC_BIN/announce_erase" ]
    then
        $OSC_BIN/announce_erase "%{name}" \
                                "%{version}-%{release}" \
                                "%{install_dir}" \
                                "%{vendor_name}" \
                                "%{plugin_module}"
    fi
    if [ -f %{vendor_dir}/__init__.py ]
    then
        python %{vendor_dir}/__init__.py %{vendor_dir} >/dev/null 2>&1
        if [ "$?" -eq 0 ]
        then
            rm -f %{vendor_dir}/__init__.py
            rm -f %{vendor_dir}/__init__.pyc
            rm -f %{vendor_dir}/__init__.pyo
        fi
    fi
fi
exit 0
```

```
%postun
if [ $1 -ge 1 ]
then
    OSC_BIN=/opt/storage-connect/bin
    if [ -x "$OSC_BIN/announce_upgrade" ]
    then
        $OSC_BIN/announce_upgrade "%{name}" \
                                  "%{version}-%{release}" \
                                  "%{install_dir}" \
                                  "%{vendor_name}" \
                                  "%{plugin_module}"
     fi
elif [ $1 -eq 0 ]
then
    rmdir --ignore-fail-on-non-empty %{install_dir}
    rmdir --ignore-fail-on-non-empty %{vendor_dir}
fi
exit 0

%post
OSC_BIN=/opt/storage-connect/bin
if [ -x "$OSC_BIN/announce_install" ]
then
    $OSC_BIN/announce_install "%{name}" \
                              "%{version}-%{release}" \
                              "%{install_dir}" \
                              "%{vendor_name}" \
                              "%{plugin_module}"
fi
if [ ! -f %{vendor_dir}/__init__.py ]
then
    cat >%{vendor_dir}/__init__.py <<__EOF__
"""
@copyright: Copyright (C) 2008, 2010 Oracle and/or its affiliates. All rights
reserved.

@license:   This program is free software; you can redistribute it and/or
            modify it under the terms of the GNU General Public
            License as published by the Free Software Foundation, version 2.

            This program is distributed in the hope that it will be useful,
            but WITHOUT ANY WARRANTY; without even the implied warranty of
            MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
            General Public License for more details.

            You should have received a copy of the GNU General Public
            License along with this program; if not, write to the Free Software
            Foundation, Inc., 59 Temple Place - Suite 330, Boston,
            MA 021110-1307, USA

@summary:   This is just to enable the import of the plugins.
"""

#
# This is a major hack to get around RPM's limitation of not being able
# to specify the same file for multiple RPM installs
#
# Exit codes:
```

```
#   0 : No more dependencies
#   1 : ERROR condition
#   2 : Still have dependencies
#
if __name__ == '__main__':
    import sys
    import os
    if len(sys.argv) <= 1:
        print "ERROR: No vendor dir given!"
        sys.exit(1)

    try:
       vendor_dirs = os.listdir(sys.argv[1])
       ref_count = 0
       for vendor_dir in vendor_dirs:
           vendor_path = os.path.join(sys.argv[1], vendor_dir)
           if os.path.isdir(vendor_path):
               ref_count += 1

    except Exception, e:
        print "ERROR:", e
        sys.exit(1)

    if ref_count > 1:
        sys.exit(2)

    sys.exit(0)

__EOF__
fi
exit 0

%changelog
* Thu Jul  1 2010 Wiekus Beukes <wiekus.beukes@oracle.com> 1.0.0-4
- Added requirer RPM generation
- Updated the license

* Fri Jun  4 2010 Wiekus Beukes <wiekus.beukes@oracle.com> 1.0.0-3
- Fixed bug in the generated __init__.py

* Wed Mar  3 2010 Wiekus Beukes <wiekus.beukes@oracle.com> 1.0.0-2
- Removed ISA/IFS layer
- Added the vendor directory __init__.py logic

* Mon Nov 16 2009 Wiekus Beukes <wiekus.beukes@oracle.com> 1.0.0-1
- Initial packaging
```

Directory hierarchy

The directory hierarchy for a plug-in is as follows:



For example, the `gensa` plug-in the tar file listing is as follows:

```
osc-oracle-gensa-1.0.0/
osc-oracle-gensa-1.0.0/OracleGenSCSI.py
osc-oracle-gensa-1.0.0/GenStorageServer.py
osc-oracle-gensa-1.0.0/OracleGenNFS.py
osc-oracle-gensa-1.0.0/__init__.py
```