

Provider

Status

This is a Draft Document intended to evolve into a Proposed Specification. It does not specify an OSID specification of any kind.

Draft Documents are works in progress that may be updated, replaced, or obsoleted at any time. It is inappropriate to use Draft Documents as reference material or to cite them other than as a *work in progress*.

Abstract

The Provider OSID defines an interface for finding, installing and loading OSID implementations. A provider of this service can construct a repository of OSID implementations characterized by various properties such that the consumer can perform searches to discover implementations by OSID, by language binding, or other data defined in the interface or the provider's Properties set.

Provider provides a means for installing OSID implementations to make them available for execution. Installed OSID implementations may be updated if new versions become available or removed.

The Provider defines only those methods pertinent to a consumer to Provider. There are several issues an implementation Provider must address such as the structure of the installation data, version control, tracking installed Providers and local file management. These issues are considered to be within the scope of a Provider implementation and do not appear in this specification.

Table of Contents

1. Introduction.....	2
2. Note on Terminology	2
3. Design	3
3.1. Scope.....	3
3.2. Managers and Objects	3
3.3. Providers and OSIDs	4

3.4.	Providers and Versioning	5
3.5.	Providers and OSID Instantiation	5
3.6.	Provider Installations	5
3.7.	Concurrency	5
3.8.	Compatibility.....	5
4.	Definitions	6
4.1.	ProviderControlManager	6
4.2.	ProviderConsumerManager	7
4.3.	Provider	11
4.4.	ProviderIterator	14
5.	Examples and Scenarios	15
6.	Future Work.....	16
7.	Contributors	16
8.	References.....	16
9.	Copyright Statement.....	16

1. Introduction

The Provider service provides a means for managing OSID implementations. A means is required to locate, identify and install OSID implementations based on information pertaining to the desired functionality and/or the interoperability characteristics of the implementation. A Java-based Repository application may be interested in learning about new Java-based Repository implementations or a person might want to know if there is a Scheduling implementation that manages the Red Sox schedule that can be used from a PHP platform. These implementations might be stored on one or more networked servers that in turn might provide one or more means of download.

Once the desired implementation has been identified, there needs to be a method available for making that implementation available for execution on the local platform. This will generally involve installing some files from a download package.

As with all OSIDs, the intent of the Provider OSID is to provide a service that can be used in a variety of contexts. This might be from within a specific application, as a standalone OSID implementation management utility, or running within a server process. The design of the interface should be such that it usable across these contexts and properly divides the responsibilities between the OSID consumer and the OSID provider.

2. Note on Terminology

Generally, the term OSID consumer is used to describe the application or user of an OSID and OSID provider is used to identify the OSID implementation. To avoid confusion by using phrases such as *the Provider of the Providers*, this document uses *application* to mean OSID consumer and *implementation* to mean OSID provider.

3. Design

3.1. Scope

OSID implementations may be constructed for a variety of platforms and delivered through a variety of means. The design of the Provider interface is concerned with what information the application needs through the interface and this set is generally a subset of all the information required to successfully manage installations. An application should be able to perform the following functions:

- determine if a required Provider is installed
- determine if installed Providers have updates available
- determine the Providers required for the execution of a given Provider
- find additional Providers given search criteria
- test for compatibility with a given Provider
- identify Providers by a unique Id
- install a Provider
- uninstall a Provider
- instantiate the implementation described by the Provider

Application utilities may be developed that assist with Provider management that need to perform the following additional functions:

- post a Provider and its associated data
- update a posted Provider
- remove a posted Provider

Explicitly omitted from this interface design are issues associated with file management. It is expected that the concerns of file formats, download means, installation locations or any data with regard to the physical installation be addressed by the implementation of the Provider service.

3.2. Managers and Objects

The functionality of the Provider OSID can be organized into the following categories:

- methods pertaining to finding Providers
- methods pertaining to publishing Providers
- methods pertaining to installing Providers

An application might be interested in utilizing a subset of this list of functions and similarly an implementation might support a subset. Sorting these functions among separate managers simplifies the understanding of the interface specification and provides a means to describe method-level compliance at an interface level which lessens the reliance on UNIMPLEMENTED errors.

This separation is implemented by the definition of three sub-managers; `ProviderLookupManager`, `ProviderInstallationManager` and `ProviderPromulgationManager`. Each of these managers is accessed via a central `ProviderControlManager`. The `ProviderControlManager` is instantiated through the `OsidLoader`.

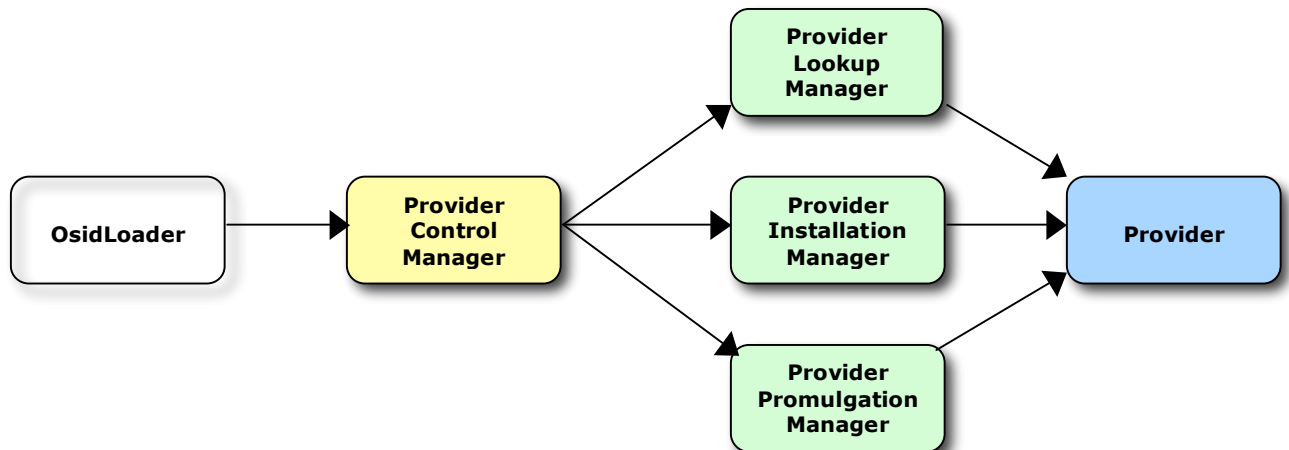


Figure 1: the ProviderControlManager used to access sub-managers

The Provider service manages Provider objects. Each Provider object represents an OSID implementation that may be examined in a remote repository of OSID implementations or installed locally for execution. The Provider interface does not specify the format of the implementation and reveals only enough information for the identification and management of the Provider from an application point of view. Additional information, such as versioning, platform, installed files and configurations may be required from a particular Provider implementation but is not exposed through the interface. A Provider contains Properties but the specification of these Properties is outside the scope of this specification (as it is with existing OSID specifications).

A ProviderIterator is defined to provide a means of returning multiple Provider objects.

3.3. Providers and OSIDs

A Provider represents a single OSID implementation. An OSID implementation may be composed of one or more files. Again, the structure of the installation and the method of acquisition are to be addressed within an implementation of Provider. A Provider implementation may download a zip file and uncompress a bunch of files while another may perform multiple downloads. Conceivably, a Provider may perform no permanent localized installation.

Sometimes, it is desirable to package multiple OSID implementations within a single file. Providers map to OSID implementations and consequently an installation of one Provider may result in an installation of multiple Providers. The implementation of Provider is responsible for managing the state of installed Providers in a local registry. This registry is not in the scope of the Provider

interface itself, but it may be desirable to develop a registry utility to simplify the task of coordinating among multiple Provider implementations.

3.4. Providers and Versioning

A Provider represents an OSID implementation that is coded to one or more versions of a language binding that in turn maps to a specific OSID specification version. The language-binding version (currently, this version is the same as the OSID specification version) is relevant in determining application to implementation compatibility.

A Provider also represents a single version of an OSID implementation. A new Id should be designated for each available implementation update. The application is never in a complete position to assess logical compatibility issues across versions and across other dependant applications and as such a supplier may opt to not honor an update request and require a new installation of an OSID implementation under a different package name. Conversely, a Provider implementation may opt to map a Provider Id request to a more current version of the Provider identified by a different Id.

This gives the supplier some room to determine compatibility issues on a case-by-case basis that may not be describable in an interface or list of Properties. It also leaves room for an application or other utility to uninstall a new version and re-install an older version if a problem arises. While the version of an implementation is not exposed in the interface since such a version number is specific to a given implementation, means for identifying newer and previous versions based on Id are specified.

3.5. Providers and OSID Instantiation

Included in the Provider interface is a `getManager()` method to instantiate an OSID implementation through the given Provider Id in lieu of an application call to `OsidLoader.getManager()`. This serves to keep implementation location concerns in the domain of the Provider implementation.

3.6. Provider Installations

The Provider interface provides a means for updating, but it cannot be assumed that every update in a version sequence will be installed. Provider suppliers should assume that each update is a complete installation.

3.7. Concurrency

A Provider may provision for a means to support multiple versions of an implementation through the `getManager()` method. As each version is identified by a unique Id, a Provider may wish to organize installations based on an internal version identifier and load the implementation identified by its unique Id.

3.8. Compatibility

Application-implementation compatibility depends on a number of interoperability factors including implemented methods, supported Types, and Properties. Since these factors will vary on an OSID by OSID basis, they are omitted from the interface specification and should be addressed inside the Provider Properties.

4. Definitions

4.1. ProviderControlManager

<i>Method Summary</i>	provider.ProviderControlManager	
Implements	OsidManager	
supportsProviderLookup		
Description	Tests for the availability of a lookup manager from this provider.	
Return	boolean	true if a lookup manager is available, false otherwise
Compliance	mandatory	This method must be implemented in all providers.
getProviderLookupManager		
Description	Gets the lookup manager for this service.	
Return	provider.ProviderLocatorManager	the lookup manager for this service
Errors	OPERATION_FAILED	unable to complete request
	UNIMPLEMENTED	a lookup manager is not available for this provider
Compliance	optional	This method must be implemented if supportsProviderLookup() is true.
supportsProviderInstallation		
Description	Tests for the availability of an installation manager from this provider.	
Return	boolean	true if an installation manager is available, false otherwise
Compliance	mandatory	This method must be implemented in all providers.
getProviderInstallationManager		
Description	Gets the installation manager for this service.	
Return	provider.ProviderInstallationManager	the installation manager for this service
Errors	OPERATION_FAILED	unable to complete request
	UNIMPLEMENTED	an installation manager is not available for this provider
Compliance	optional	This method must be implemented if supportsProviderInstallation() is true.
supportsProviderPromulgation		
Description	Tests for the availability of a promulgation manager from this provider.	
Return	boolean	true if a promulgation manager is available, false otherwise
Compliance	mandatory	This method must be implemented in all providers.
getProviderPromulgationManager		
Description	Gets the promulgation manager for this service.	
Return	provider.ProviderPromulgationManager	the promulgation manager for this service
Errors	OPERATION_FAILED	unable to complete request
	UNIMPLEMENTED	a promulgation manager is not available for this provider
Compliance	optional	This method must be implemented if supportsProviderPromulgation() is true.

4.2. ProviderLookupManager

<i>Method Summary</i>	provider.ProviderLookupManager		
	getProvider		
Description	Gets the Provider object specified by its Id.		
Parameters	id.Id	providerId	the Id of the Provider
Return	provider.Provider the Provider object identified by the given Id.		
Errors	NOT_FOUND		no Provider with given Id was found
	NULL_ARGUMENT		a null argument provided
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in accessing Providers
Compliance	mandatory		This method must be implemented in all providers.
	getProviders		
Description	Gets all the Provider objects known to this provider.		
Return	provider.ProviderIterator		an iterator containing the Provider objects
Errors	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in accessing Providers
Compliance	mandatory		This method must be implemented in all providers.
	getProvidersByOsid		
Description	Gets all the Provider objects matching a given OSID.		
Parameters	string	osid	name of the osid
	string	binding	name of the language binding
	string	version	osid version of the binding release
Return	provider.ProviderIterator		an iterator containing the Provider objects
Errors	NULL_ARGUMENT		a null argument provided
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in accessing Providers
Compliance	mandatory		This method must be implemented in all providers.
	getSearchTypes		
Description	Gets all the query types supported by this provider.		
Return	type.TypeIterator		an iterator containing the search types
Compliance	mandatory		This method must be implemented in all providers.
	supportsSearchType		
Description	Tests if this provider supports the given query type.		
Parameters	type.Type	searchType	the search type
Return	boolean		true if this provider supports the given searchType, false otherwise
Compliance	mandatory		This method must be implemented in all providers.
	getProvidersBySearch		
Description	Gets all the Provider objects known to this provider.		
Parameters	java.io.Serializable	searchCriteria	the query
	type.Type	searchType	the type of search
Return	ProviderIterator		an iterator containing the Provider objects
Errors	INVALID_ARGUMENT		searchCriteria is not of searchType
	NULL_ARGUMENT		a null argument provided
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in accessing Providers
	UNKNOWN_TYPE		searchType is unknown
Compliance	mandatory		This method must be implemented in all providers.

4.3. ProviderInstallationManager

<i>Method Summary</i>	provider.ProviderInstallationManager		
	installProvider		
Description	Installs a Provider.		
Parameters	id.id	providerId	the Id of the Provider to install
Errors	NOT_FOUND		Provider not found
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in accessing Providers
Compliance	mandatory		This method must be implemented in all providers.
	updateProvider		
Description	Updates a Provider.		
Parameters	id.id	providerId	the Id of the Provider to update
Errors	NOT_FOUND		Provider not found or not installed.
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in accessing Providers
Compliance	mandatory		This method must be implemented in all providers.
	removeProvider		
Description	Removes a Provider.		
Parameters	id.id	providerId	the Id of the Provider to remove
Errors	NOT_FOUND		Provider not found
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in accessing Providers
Compliance	mandatory		This method must be implemented in all providers.
	getInstalledProviders		
Description	Gets the list of Providers that are installed.		
Return	provider.ProviderIterator	an iterator containing the Providers installed	
Errors	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in accessing Providers
Compliance	mandatory		This method must be implemented in all providers.
	getProvidersNeedingUpdate		
Description	Gets the list of Providers that are installed and need to be updated.		
Return	provider.ProviderIterator	an iterator containing the Providers that need to be updated	
Errors	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in accessing Providers
Compliance	mandatory		This method must be implemented in all providers.

getManager			
Description	Gets the OsidManager corresponding to the given Id.		
Parameters	id.Id	providerId	the Id of the Provider
	OsidContext	context	the context for the OsidLoader
	Properties	properties	the additional properties for OsidLoader
Return	OsidManager	the OsidManager	
Errors	CONFIGURATION_ERROR		an error in implementation configuration
	NOT_FOUND		cannot find this implementation
	NULL_ARGUMENT		a null argument provided
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in accessing Providers
	VERSION_ERROR		requested version not available
Compliance	mandatory	This method must be implemented in all providers.	

4.4. ProviderPromulgationManager

<i>Method Summary</i>	provider.ProviderPromulgationManager		
	createProvider		
Description	Creates a new Provider.		
Parameters	string	displayName	the display name of the Provider
	string	description	the description of the new Provider
	string	osid	the name of the OSID of this Provider
	string	binding	the OSID binding of this Provider
	string	version	the version of the OSID binding
	string	license	the rights statement associated with the usage or acquisition of this Provider
	string	publisher	the publisher or distributor of this Provider
	string	creator	the creator of this Provider
	date	releaseDate	the date of this Provider release
	Provider[]	dependencies	the Providers this provider depends on
	Properties	properties	properties associated with this Provider
string	path	location of the installation	
Return	provider.Provider	the Provider object created	
Errors	INVALID_ARGUMENT	a problem with a parameter	
	NOT_FOUND	installation path not found	
	NULL_ARGUMENT	a null argument provided	
	OPERATION_FAILED	unable to complete request	
	PERMISSION_DENIED	authorization failure in creating Provider	
Compliance	mandatory	This method must be implemented in all providers.	
	updateProvider		
Description	Persists the changes to the given Provider.		
Parameters	Provider	provider	the Provider to update
Errors	NULL_ARGUMENT	a null argument provided	
	OPERATION_FAILED	unable to complete request	
	PERMISSION_DENIED	authorization failure in updating Provider	
Compliance	mandatory	This method must be implemented in all providers.	
	deleteProvider		
Description	Deletes the Provider identified by the given Id.		
Parameters	Id	providerId	the Id of the Provider to delete
Errors	NOT_FOUND	Provider identified by this Id was not found	
	NULL_ARGUMENT	a null argument provided	
	OPERATION_FAILED	unable to complete request	
	PERMISSION_DENIED	authorization failure in deleting Provider	
Compliance	mandatory	This method must be implemented in all providers.	

4.5. Provider

<i>Method Summary</i>	provider.Provider		
	getId		
Description	Gets the Id associated with this Provider.		
Return	Id	the Id	
Compliance	mandatory	This method must be implemented in all providers.	
	getDisplayName		
Description	Gets the display name associated with this Provider.		
Return	string	the display name	
Compliance	mandatory	This method must be implemented in all providers.	
	setDisplayName		
Description	Sets the display name associated with this Provider.		
Parameters	string	displayName	the new display name
Errors	INVALID_ARGUMENT		an invalid display name provided
	NULL_ARGUMENT		a null argument provided
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in updating display name
Compliance	optional	This method must be implemented if supportsUpdate() is true.	
	getDescription		
Description	Gets the description associated with this Provider.		
Return	string	the description	
Compliance	mandatory	This method must be implemented in all providers.	
	setDescription		
Description	Sets the description associated with this Provider.		
Parameters	string	description	the new description
Errors	INVALID_ARGUMENT		an invalid description provided
	NULL_ARGUMENT		a null argument provided
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in updating description
Compliance	optional	This method must be implemented if supportsUpdate() is true.	
	getOsidName		
Description	Gets the OSID name associated with this Provider.		
Return	string	the name of the OSID	
Compliance	mandatory	This method must be implemented in all providers.	
	getOsidBinding		
Description	Gets the OSID binding associated with this Provider.		
Return	string	the name of the OSID binding	
Compliance	mandatory	This method must be implemented in all providers.	
	getOsidBindingVersion		
Description	Gets the version of the OSID binding associated with this Provider.		
Return	string	the version of the OSID binding	
Compliance	mandatory	This method must be implemented in all providers.	

setOsId			
Description	Sets the OSID data associated with this Provider.		
Parameters	string	osid	the new OSID name
	string	binding	the OSID binding
	string	version	the version of the OSID binding
Errors	INVALID_ARGUMENT		an invalid description provided
	NULL_ARGUMENT		a null argument provided
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in updating data
Compliance	optional	This method must be implemented if supportsUpdate() is true.	
getLicense			
Description	Gets the license associated with this Provider.		
Return	string	the license	
Compliance	mandatory	This method must be implemented in all providers.	
setLicense			
Description	Sets the license associated with this Provider.		
Parameters	string	license	the new license
Errors	INVALID_ARGUMENT		an invalid license provided
	NULL_ARGUMENT		a null argument provided
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in updating license
Compliance	optional	This method must be implemented if supportsUpdate() is true.	
getPublisher			
Description	Gets the publisher associated with this Provider.		
Return	string	the publisher	
Compliance	mandatory	This method must be implemented in all providers.	
setPublisher			
Description	Sets the publisher associated with this Provider.		
Parameters	string	publisher	the new publisher
Errors	INVALID_ARGUMENT		an invalid publisher provided
	NULL_ARGUMENT		a null argument provided
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in updating publisher
Compliance	optional	This method must be implemented if supportsUpdate() is true.	
getCreator			
Description	Gets the creator associated with this Provider.		
Return	string	the creator	
Compliance	mandatory	This method must be implemented in all providers.	
setCreator			
Description	Sets the creator associated with this Provider.		
Parameters	string	creator	the new creator
Errors	INVALID_ARGUMENT		an invalid creator provided
	NULL_ARGUMENT		a null argument provided
	OPERATION_FAILED		unable to complete request
	PERMISSION_DENIED		authorization failure in updating creator
Compliance	optional	This method must be implemented if supportsUpdate() is true.	

getReleaseDate		
Description	Gets the release date of this Provider.	
Return	date	the release date
Compliance	mandatory	This method must be implemented in all providers.
getDependencies		
Description	Gets the Providers this Provider depends on.	
Return	ProviderIterator	An iterator containing the Provider objects
Compliance	mandatory	This method must be implemented in all providers.
addDependency		
Description	Adds a dependency to this Provider.	
Parameters	Id	providerId
Errors	NOT_FOUND	no Provider found identified by the given Id
	NULL_ARGUMENT	a null argument provided
	OPERATION_FAILED	unable to complete request
	PERMISSION_DENIED	authorization failure in adding dependency
Compliance	optional	This method must be implemented if supportsUpdate() is true.
removeDependency		
Description	Removes a dependency from this Provider.	
Parameters	Id	providerId
Errors	NOT_FOUND	no Provider found identified by the given Id
	NULL_ARGUMENT	a null argument provided
	OPERATION_FAILED	unable to complete request
	PERMISSION_DENIED	portsUpdate() is true.
Compliance	optional	This method must be implemented if supportsUpdate() is true.
isInstalled		
Description	Tests if this Provider has been installed.	
Return	boolean	true if this Provider is installed, false otherwise
Compliance	mandatory	This method must be implemented in all providers.
needsUpdate		
Description	Tests if this Provider is installed and is out of date.	
Return	boolean	true if this Provider is installed and needs to be updated, false otherwise
Compliance	mandatory	This method must be implemented in all providers.
supportsUpdate		
Description	Tests if this Provider supports update methods.	
Return	boolean	true if this Provider supports update methods, false otherwise
Compliance	mandatory	This method must be implemented in all providers.
getPreviousVersion		
Description	Gets the previous version of this Provider.	
Return	Provider	The previous version of this Provider, or null if there is not a previous version.
Compliance	mandatory	This method must be implemented in all providers.

PROPERTIES!

4.6. ProviderIterator

<i>Method Summary</i>	provider.ProviderIterator	
	hasNextProvider	
Description	Tests if there are more Provider objects in this iterator.	
Return	boolean	true if this there are more Provider objects in this iterator, false otherwise
Compliance	mandatory	This method must be implemented in all providers.
	getNextProvider	
Description	Gets the next Provider object in this iterator.	
Return	provider.Provider	the next Provider object
Compliance	mandatory	This method must be implemented in all providers.

5. Examples and Scenarios

```

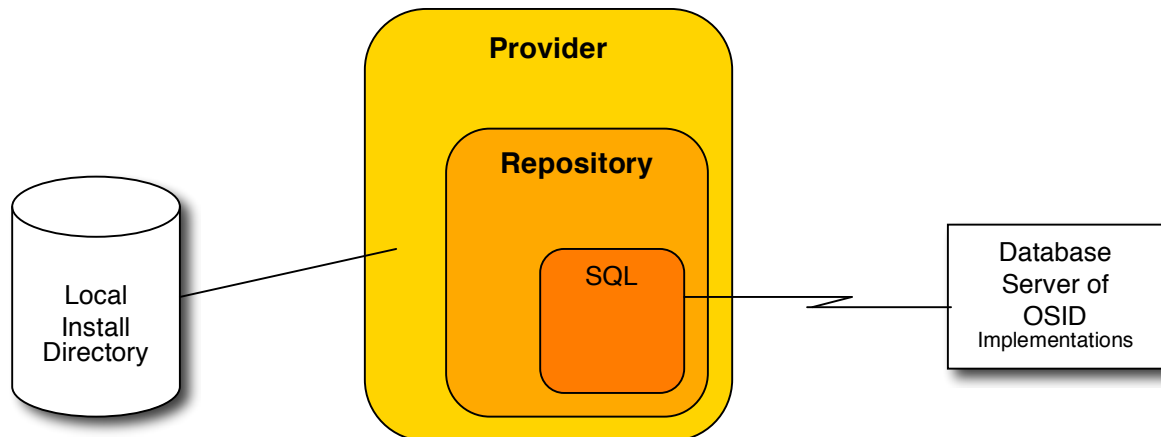
ProviderControlManager pm = OsidLoader.getManager("org.osid.provider", impl,
                                                new OsidContext(), null);
ProviderLookupManager plm = pm.getLookupManager();
ProviderInstallationManager pim = pm.getInstallationManager();

ProviderIterator pi = plm.getProvidersByOsid("repository", "java", "2.0");

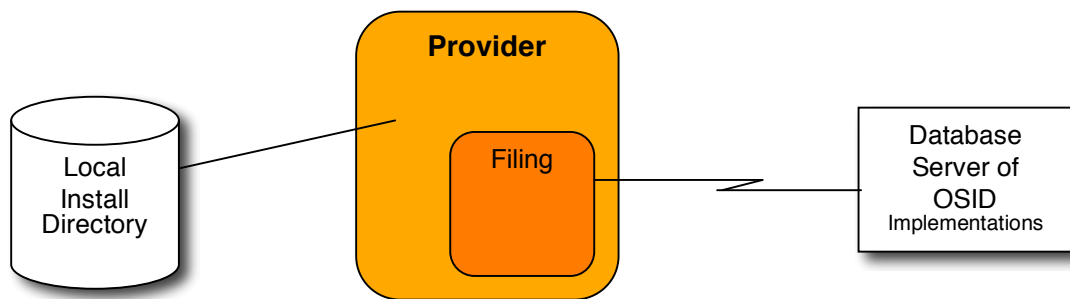
while (pi.hasNext()) {
    Provider p = pi.getNextProvider();
    pim.install(p.getId());
}

```

Example 1: installing OSID Repository implementations



Example 2: a design of a Provider implementation



Example 3: another design of a Provider implementation

6. Future Work

With the specification of an interface for Provider comes the assumption that multiple implementations of Provider may be developed that behave in different ways. In an ideal world there would be a single OSID implementation repository and a consistent means of managing locally installed libraries and files. This could be said of any service but nonetheless we have OSIDs.

Implementing functions such as a single local registry of installed implementations, robust file management and integrity, configuration management, and rich version control, while not seen as concerns for applications of Provider itself, are desirable and may contribute significantly to the complexity of a Provider implementation. What components here are deserving of a service interface remain to be determined but an implementer of Provider would benefit from the existence of modular or singleton utilities that may be used to layer a Provider implementation upon.

This design assigns interoperability characteristics such as supported methods and Types to the Provider Properties. A set of Properties can be defined for each OSID that describes these characteristics and make them available for examination or searching.

7. Contributors

Jeff Kahn
Verbana Consulting

Scott Thorne
Massachusetts Institute of Technology

8. References

O.K.I. Open Service Interface Definitions, Massachusetts Institute of Technology, 2003.

9. Copyright Statement

Copyright (C) Massachusetts Institute of Technology (2006). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Open Knowledge Initiative (O.K.I.), the Massachusetts Institute of Technology (MIT), or other organizations, except as required to translate it into languages other than English.

This document and the information contained herein is provided on an "AS IS" basis and the Massachusetts Institute of Technology and the Open Knowledge Initiative DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED,

INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION
HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF
MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.