Specification: SoftwareManagement

General

The Software Management set of services contains services for on-board software management of a spacecraft. The following services are contained in the Software Management set:
The Memory Management service is a low-level service intended to be used to load, dump and check memory addresses.
The Software Image service is intended to manage Software Images and Software Patches by allowing the generation of new Software Images from a patch.
The Package Management service allows the management of software packages on modern operating systems using the concept of packages.
The Apps Launcher service allows the management of runnable applications in a system.
The Heartbeat service publishes a periodic beat to the consumers that are listening.

This section details the Software Management services. The area and structures are defined in terms of the MO Message Abstraction Layer (MAL), so it is possible to deploy them over any supported protocol and message transport.

Service: MemoryManagement

General

The Memory Management service provides the ability to load, dump and check binary data blocks in known memory addresses from/to a certain memory device.

Table 0-1: MemoryManagement Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
SoftwareManagement	MemoryManagement	7	1	1
Interaction Pattern	**Operation Identifier**	**Operation Number**	**Support in Replay**	**Capability Set**
SUBMIT	loadMemory	1	No	1
PROGRESS	dumpMemory	2	No	2
SUBMIT	abortMemoryDump	3	No	
INVOKE	checkMemory	4	No	3

High Level Requirements

a) The Memory Management service shall provide:
   a. the capability for loading an on-board's memory device;
   b. the capability for dumping data from an on-board's memory device and aborting it;
   c. the capability for checking an on-board's memory device.

b)

Functional Requirements

OPERATION: loadMemory

General

The loadMemory operation allows a consumer to submit a block of data to a provider for loading the memory of the selected device. The operation shall use the Activity Tracking to report on the progress of the activity.

Operation Identifier	loadMemory	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Type
IN	SUBMIT	deviceName : (MAL::Identifier) base : (MAL::UInteger) dataBlocks : (List<DataBlock>)

Structures

a) The deviceName field shall contain the name of the destination device. The interpretation of the value is implementation-specific.
b) The base field shall contain the base reference of the memory within the memory block.
c) The dataBlocks field shall contain the memory block data to be loaded.

Errors

The operation may return one of the following errors:

ERROR: UNKNOWN_DEVICE

The submitted device is unknown.

Error	Error #	ExtraInfo Type
UNKNOWN_DEVICE	711	Not Used

ERROR: INVALID_ADDRESS

The submitted base is invalid.

Error	Error #	ExtraInfo Type
INVALID_ADDRESS	712	Not Used

OPERATION: dumpMemory

General

The dumpMemory operation allows a consumer to dump memory areas of a selected device from a provider. The dumping process can be aborted by using the identifier provided in the acknowledge phase.

Operation Identifier	dumpMemory	
Interaction Pattern	PROGRESS	
Pattern Sequence	Message	Body Type
IN	PROGRESS	deviceName : (MAL::Identifier)

		base : (MAL::UInteger) memoryAreas : (List<MemoryArea>)
OUT	ACK	requestId : (MAL::Long)
OUT	UPDATE	dataBlock : (DataBlock)
OUT	RESPONSE	

Structures

a) The deviceName field shall contain the name of the destination device. The interpretation of the value is implementation-specific.
b) The base field shall contain the base reference of the memory within the memory block.
c) The memoryAreas field shall contain the memory areas to be dumped.
d) The requestId field shall contain the id of the request.
e) The dataBlock field shall contain the memory block data being dumped.

Errors

The operation may return one of the following errors:

ERROR: UNKNOWN_DEVICE

The submitted device is unknown.

Error	Error #	ExtraInfo Type
UNKNOWN_DEVICE	711	Not Used

ERROR: INVALID_ADDRESS

The submitted base is invalid.

Error	Error #	ExtraInfo Type
INVALID_ADDRESS	712	Not Used

OPERATION: abortMemoryDump

General

The abortMemoryDump operation allows a consumer to abort on-going dumping processes.

Operation Identifier	abortMemoryDump	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Type
IN	SUBMIT	requestIds : (List<MAL::Long>)

Structures

a) The requestIds field shall contain the request ids to be aborted.

Errors

The operation does not return any errors.

OPERATION: checkMemory

General

The checkMemory operation allows a consumer to check memory areas of a selected device. The response contains the checksum for the selected checksum algorithm.

Operation Identifier	checkMemory	
Interaction Pattern	INVOKE	
Pattern Sequence	Message	Body Type
IN	INVOKE	deviceName : (MAL::Identifier) base : (MAL::UInteger) checkAreas : (List<MemoryArea>) checksumAlgorithm : (MAL::String)
OUT	ACK	
OUT	RESPONSE	checksums : (MAL::Blob)

Structures

a) The deviceName field shall contain the name of the destination device. The interpretation of the value is implementation-specific.
b) The memoryAreas field shall contain the memory areas to be dumped.
c) The checkAreas field shall contain the memory areas to be checked.
d) The checksumAlgorithm field shall contain the checksum algorithm to be used.
e) The checksums field shall contain the generated checksums for the selected memory areas.

Errors

The operation may return one of the following errors:

ERROR: UNSUPPORTED_CHECKSUM

The requested checksum algorithm is not supported.

Error	Error #	ExtraInfo Type
UNSUPPORTED_CHECKSUM	1	Not Used

ERROR: UNKNOWN_DEVICE

The submitted device is unknown.

Error	Error #	ExtraInfo Type
UNKNOWN_DEVICE	711	Not Used

ERROR: INVALID_ADDRESS

The submitted base is invalid.

Error	Error #	ExtraInfo Type
INVALID_ADDRESS	712	Not Used

Service: SoftwareImage

General

The Software Image service provides the ability to deploy, list, generate, delete, clone and check the integrity of Software Images.
This service can be used to manage Software Images on-board of a spacecraft or to manage Software Images in virtual machines that might run on-board of a spacecraft.
The service supports the generation of a new Image from a previous baseline Image with an additional delta. This allows patching software Images without the need to transfer the complete new Image.

Table 0-2: SoftwareImage Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
SoftwareManagement	SoftwareImage	7	2	1
Interaction Pattern	Operation Identifier	Operation Number	Support in Replay	Capability Set
SUBMIT	deployImage	1	No	1
INVOKE	checkImageIntegrity	2	No	2
INVOKE	cloneImage	3	No	3
REQUEST	listImage	4	Yes	4
INVOKE	patchImage	5	No	5
REQUEST	addImage	6	No	
SUBMIT	deleteImage	7	No	
REQUEST	listPatch	8	Yes	6
REQUEST	addPatch	9	No	7
SUBMIT	deletePatch	10	No	

High Level Requirements

a) The Software Image service shall provide:
   a. to be done;
   b. to be done;
   c. to be done.
b) All on-board application software shall be updatable by the Ground at any time during the mission.
c) On-board application software images shall be stored on-board in non-volatile memory.
d) It shall be possible to add or remove patches to on-board software images stored on-board without the need to reload the entire original image.
e) In presence of any single failure, the remaining non-volatile memory used to store on-board software images shall have enough space to store at least two different versions of the same software.
f) Ground shall be provided with the capability to define in non-volatile memory for each on-board processor the main and redundant location of the software image to be used at the next processor reset / reconfiguration.
g) The process of loading a software image from non-volatile memory to RAM upon boot shall be robust to data corruptions in various parts of the stored image.

h)

Functional Requirements

COM usage

Table 0-3: SoftwareImage Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
SoftwareImage	1	ImageDefinition		2
Patch	2	PatchDefinition	1	

COM Object Relationships

The Figure below shows the COM object relationships for this service:

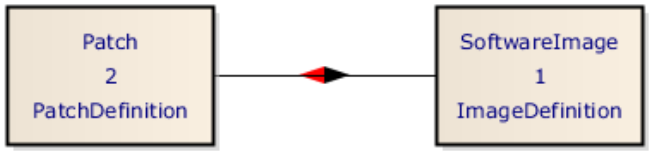

Figure 0-1: SoftwareImage Service COM object relationships

OPERATION: deployImage

General

Operation Identifier	deployImage	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Type
IN	SUBMIT	imageInstId : (MAL::Long)

Structures

Errors

The operation does not return any errors.

OPERATION: checkImageIntegrity

General

The checkImageIntegrity operation allows a consumer to check the integrity of a software image on the provider. The checksum shall be generated from the checksum algorithm selected by the consumer.

Operation Identifier	checkImageIntegrity	
Interaction Pattern	INVOKE	
Pattern Sequence	Message	Body Type
IN	INVOKE	imageInstIds : (List<MAL::Long>) checksumAlgorithm : (MAL::String)
OUT	ACK	
OUT	RESPONSE	checksums : (List<MAL::Blob>)

Structures

a) The imageInstIds shall contain the object instance identifier of the image to be checked.

Errors

The operation may return the following error:

ERROR: UNSUPPORTED_CHECKSUM

The requested checksum algorithm is not supported.

Error	Error #	ExtraInfo Type
UNSUPPORTED_CHECKSUM	1	Not Used

OPERATION: cloneImage

General

The cloneImage operation allows a consumer to clone a software image on the provider.

Operation Identifier	cloneImage	
Interaction Pattern	INVOKE	
Pattern Sequence	Message	Body Type
IN	INVOKE	imageInstIds : (List<MAL::Long>) newNames : (List<MAL::Identifier>)
OUT	ACK	
OUT	RESPONSE	cloneImageInstIds : (List<MAL::Long>)

Structures

Errors

The operation does not return any errors.

OPERATION: listImage

General

The listImage operation allows a consumer to request the object instance identifiers of the SoftwareImage objects for the existing Images available on the provider.

Operation Identifier	listImage	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Type
IN	REQUEST	imageNames : (List<MAL::Identifier>)

OUT	RESPONSE	imageInstId : (List<MAL::Long>)

Structures

Errors

The operation does not return any errors.

OPERATION: patchImage

General

The patchImage operation allows a consumer to generate a new software image on the provider from a previous baseline Image. The operation can use the Activity Tracking service to report the progress of the generation.

Operation Identifier	patchImage	
Interaction Pattern	INVOKE	
Pattern Sequence	Message	Body Type
IN	INVOKE	baselineImage : (MAL::Long) patchObjId : (MAL::Long)
OUT	ACK	estimateDuration : (MAL::Duration)
OUT	RESPONSE	newImageInstId : (MAL::Long)

Structures

Errors

The operation does not return any errors.

OPERATION: addImage

General

Operation Identifier	addImage	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Type
IN	REQUEST	files : (List<MAL::File>) imageDefs : (List<ImageDefinition>)
OUT	RESPONSE	imageObjIds : (List<MAL::Long>)

Structures

Errors

The operation does not return any errors.

OPERATION: deleteImage

General

The deleteImage operation allows a consumer to delete software images from the provider.

Operation Identifier	deleteImage	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Type
IN	SUBMIT	ImageInstIds : (List<MAL::Long>)

Structures

Errors

The operation does not return any errors.

OPERATION: listPatch

General

Operation Identifier	listPatch	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Type
IN	REQUEST	patchName : (List<MAL::Identifier>)
OUT	RESPONSE	patchObjIds : (List<MAL::Long>)

Structures

Errors

The operation does not return any errors.

OPERATION: addPatch

General

Operation Identifier	addPatch	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Type
IN	REQUEST	relatedImageObjIds : (List<MAL::Long>) files : (List<MAL::File>) patchDefs : (PatchDefinition)
OUT	RESPONSE	patchObjIds : (List<MAL::Long>)

Structures

Errors

The operation does not return any errors.

OPERATION: deletePatch

General

Operation Identifier	deletePatch	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Type
IN	SUBMIT	patchObjIds : (List<MAL::Long>)

Structures

Errors

The operation does not return any errors.

Service: PackageManagement

General

The Package Management service provides the ability to install, uninstall, upgrade, list, check packages. The service shall optionally support verification of packages, verification of digital signatures, upgrade software, manage dependencies.
Packages can be organized in different categories in order to facilitate the management of software.

Table 0-4: PackageManagement Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
SoftwareManagement	PackageManagement	7	3	1
Interaction Pattern	Operation Identifier	Operation Number	Support in Replay	Capability Set
REQUEST	findPackage	1	Yes	1
INVOKE	install	2	No	
INVOKE	uninstall	3	No	
INVOKE	upgrade	4	No	2
REQUEST	checkPackageIntegrity	5	No	3

High Level Requirements

a) The Package Management service shall provide:
   a. the capability for listing, installing and uninstalling packages;
   b. the capability for upgrading packages;
   c. the capability for checking the integrity of a package;
   d. the capability for listing the package categories available on the provider.

b)

Functional Requirements

COM usage

Table 0-5: PackageManagement Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
Package	1	PackageDefinition		
Installation	2	MAL::Identifier	1	

COM Event Service usage

Table 0-6: PackageManagement Service Events

Event Name	Object Number	Object Body Type	Related points to	Source points to

COM Object Relationships

The Figure below shows the COM object and event relationships for this service:

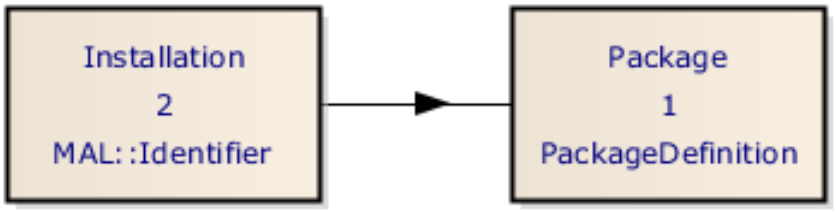

Figure 0-2: PackageManagement Service COM object and event relationships

COM Activity Service usage

Add the activity tracking for the progress of installation. Also for the Software Image service

OPERATION: findPackage

General

The findPackage operation allows a consumer to find the available packages on the provider.

Operation Identifier	findPackage	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Type
IN	REQUEST	names : (List<MAL::Identifier>)
OUT	RESPONSE	names : (List<MAL::Identifier>) installed : (List<MAL::Boolean>)

Structures

a) The names field contains the names of the packages.
b) The names field contains the names of the packages.
c) The installed field shall hold the status of the package.

Errors

The operation may return the following error:

ERROR: UNKNOWN

a) One or more of the requested packages is unknown.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::Integer>

OPERATION: install

General

The install operation allows a consumer to install the content of a package on the provider. The selected packages will be installed sequencially.

Operation Identifier	install	
Interaction Pattern	INVOKE	
Pattern Sequence	Message	Body Type
IN	INVOKE	names : (List<MAL::Identifier>)
OUT	ACK	integrity : (List<MAL::Boolean>)
OUT	RESPONSE	

Structures

a) The names field contains the names of the packages.
b) The integrity field contains the status of the package integrity.

Errors

The operation may return one of the following errors:

ERROR: UNKNOWN

a) One or more of the requested packages is unknown.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::Integer>

ERROR: INVALID

a) One or more of the requested packages is already installed.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::Integer>

OPERATION: uninstall

General

The uninstall operation allows a consumer to uninstall the content of a package on the provider. The selected packages will be uninstalled sequencially.

Operation Identifier	uninstall	
Interaction Pattern	INVOKE	
Pattern Sequence	Message	Body Type
IN	INVOKE	names : (List<MAL::Identifier>) keepConfigurations : (List<MAL::Boolean>)
OUT	ACK	
OUT	RESPONSE	

Structures

a) The names field contains the names of the packages.
b) The keepConfigurations field selects if the configuration of the executable files shall be kept.

Errors

The operation may return one of the following errors:

ERROR: UNKNOWN

a) One or more of the requested packages is unknown.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::Integer>

ERROR: INVALID

a) One or more of the requested packages is not installed.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::Integer>

OPERATION: upgrade

General

The upgrade operation allows a consumer to upgrade the content of a package on the provider. The selected packages will be upgraded sequencially.

Operation Identifier	upgrade
Interaction Pattern	INVOKE

Pattern Sequence	Message	Body Type
IN	INVOKE	names : (List<MAL::Identifier>)
OUT	ACK	
OUT	RESPONSE	

Structures

a) The names field contains the names of the packages.

Errors

The operation may return one of the following errors:

ERROR: UNKNOWN

a) One or more of the requested packages is unknown.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::Integer>

ERROR: INVALID

a) One or more of the requested packages is not installed.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::Integer>

OPERATION: checkPackageIntegrity

General

The checkPackageIntegrity operation allows a consumer to check the integrity of a certain package on the provider.

Operation Identifier	checkPackageIntegrity	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Type
IN	REQUEST	names : (List<MAL::Identifier>)
OUT	RESPONSE	validCRCs : (List<MAL::Boolean>) publicKeys : (List<MAL::String>)

Structures

a) The names field contains the names of the packages.

Errors

The operation may return the following error:

ERROR: UNKNOWN

a) One or more of the requested packages is unknown.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type

UNKNOWN	Defined in MAL	List<MAL::Integer>

Service: ProcessManagement

General

The Processes service provides the ability to monitor and manage processes running on-board.

Table 0-7: ProcessManagement Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
SoftwareManagement	ProcessManagement	7	4	1
Interaction Pattern	**Operation Identifier**	**Operation Number**	**Support in Replay**	**Capability Set**
PUBLISH-SUBSCRIBE	monitorProcess	1	No	1
SUBMIT	setRate	2	No	
SUBMIT	startProcess	3	No	2
SUBMIT	endProcess	4	No	3
REQUEST	getProcessSummary	5	No	4

High Level Requirements

Functional Requirements

OPERATION: monitorProcess

General

Operation Identifier	monitorProcess	
Interaction Pattern	PUBLISH-SUBSCRIBE	
Pattern Sequence	Message	Body Type
OUT	PUBLISH/NOTIFY	processInfo : (ProcessInformation)

Structures

a) Add that the PID and the user name shall be passed on the UpdateHeader field.

Errors

The operation does not return any errors.

OPERATION: setRate

General

Operation Identifier	setRate	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Type
IN	SUBMIT	rate : (MAL::Duration)

Structures

a) Setting it to zero stops the publishing of the stats.

Errors

The operation does not return any errors.

OPERATION: startProcess

General

Operation Identifier	startProcess	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Type
IN	SUBMIT	

Structures

Errors

The operation does not return any errors.

OPERATION: endProcess

General

Operation Identifier	endProcess	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Type
IN	SUBMIT	

Structures

Errors

The operation does not return any errors.

OPERATION: getProcessSummary

General

Operation Identifier	getProcessSummary	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Type
IN	REQUEST	processesInfo : (ProcessInformation)
OUT	RESPONSE	

Structures

Errors

The operation does not return any errors.

Service: AppsLauncher

General

The Apps Launcher service provides the ability to monitor the execution, run, stop, kill and list applications software on-board of a spacecraft. The apps can be organized in categories.
The service is independent from any particular Operating System or platform.

Table 0-8: AppsLauncher Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
SoftwareManagement	AppsLauncher	7	5	1
Interaction Pattern	**Operation Identifier**	**Operation Number**	**Support in Replay**	**Capability Set**
PUBLISH-SUBSCRIBE	monitorExecution	1	No	1
SUBMIT	runApp	2	No	2
SUBMIT	killApp	3	No	
PROGRESS	stopApp	4	No	3
REQUEST	listApp	5	Yes	4

High Level Requirements

a) The Apps Launcher service shall provide:
   a. the capability for periodic monitoring of the applications output;
   b. the capability for running and killing applications;
   c. the capability for stopping applications;
   d. the capability for listing the object instance identifiers for the available apps.

b)

Functional Requirements

COM usage

a) An App COM object represents an on-board application. The COM object body shall hold the details of the application.
    a. The App COM object source link should point to the package from where the app was installed.

Table 0-9: AppsLauncher Service Object Types

Object Name	Object Number	Object Body Type	Related points to	Source points to
App	1	AppDetails		PackageManagement::2

COM Event Service usage

a) A StopApp COM event represents a request to stop a certain application. The COM event object body shall hold the name of the provider to be stopped.
    a. The StopApp COM event related link shall point to the App COM object.
    b. The StopApp COM event source link should point to the object that caused it to be created, most likely a COM OperationActivity object.
    c. The StopApp COM event shall be generated by the stopApp operation.

b) A Stopping COM event represents an acknowledgement that the application is stopping its execution.
    a. The Stopping COM event source link shall point to the StopApp COM event that requested the application to stop or to null if there was no request.
    b. The Stopping COM event shall be generated by the application when it is going to stop its execution.

c) A Stopped COM event represents an acknowledgement that the application is going to completely stop its execution.
    a. The Stopping COM event source link shall point to the StopApp COM event that requested the application to stop or to null if there was no request.
    b. The Stopping COM event shall be generated by the application when it is completely stopping its execution.

Table 0-10: AppsLauncher Service Events

Event Name	Object Number	Object Body Type	Related points to	Source points to
StopApp	2	MAL::Identifier	1	COM::ActivityTracking::6
Stopping	3	No body		2
Stopped	4	No body		2

COM Object Relationships

The Figure below shows the COM object and event relationships for this service:

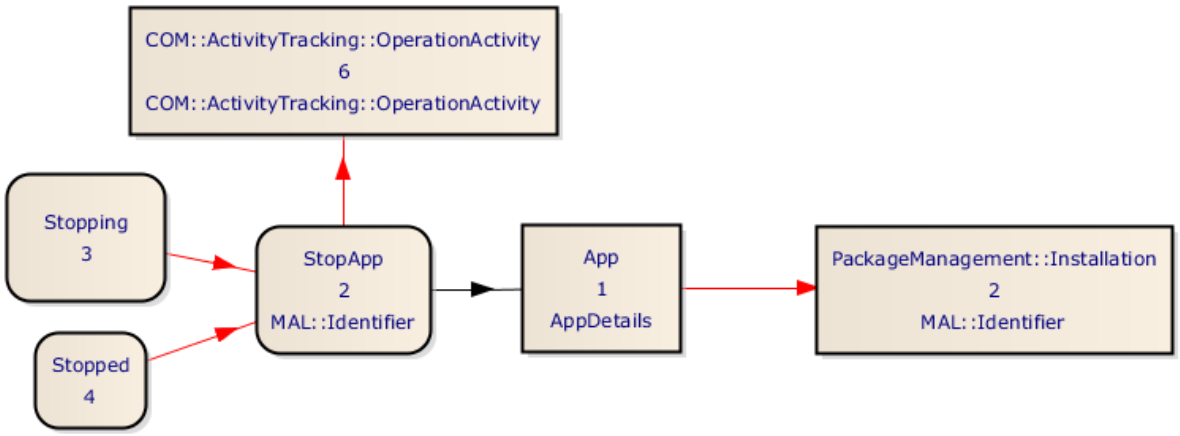

Figure 0-3: AppsLauncher Service COM object and event relationships

OPERATION: monitorExecution

General

The monitorExecution operation allows a consumer to subscribe for the application execution output stream.

Operation Identifier	monitorExecution	
Interaction Pattern	PUBLISH-SUBSCRIBE	
Pattern Sequence	Message	Body Type
OUT	PUBLISH/NOTIFY	outputStream : (MAL::String)

Structures

a) The outputStream field shall contain a stream of characters corresponding to the output stream of the application.
b) The MAL EntityKey.firstSubKey shall contain the App name.
c) The MAL EntityKey.secondSubKey shall contain the AppDetails object instance identifier.
d) The MAL EntityKey.thirdSubKey shall be NULL.
e) The MAL EntityKey.fourthSubKey shall be NULL.
f) The timestamp of the update shall be the on-board time when the update was published.
g) The publish message shall include the ObjectId of the source link of the update.
h) If no source link is needed then the ObjectId shall be replaced with a NULL.

Errors

The operation does not return any errors.

OPERATION: runApp

General

The runApp operation allows a consumer to run an application on the provider.

An object instance identifier is returned for further monitoring of the application execution in the monitorExecution operation.

Operation Identifier	runApp	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Type
IN	SUBMIT	appInstIds : (List<MAL::Long>)

Structures

a) The appInstIds field contains the list of apps to run.

Errors

The operation may return one of the following errors:

ERROR: UNKNOWN

a) One or more of the requested apps to run is unknown.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

ERROR: INVALID

a) One or more of the requested apps is already running.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

ERROR: INTERNAL

a) The process of the app could not be started.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
INTERNAL	Defined in MAL	List<MAL::UInteger>

OPERATION: killApp

General

The killApp operation allows a consumer to kill the execution of an application on the provider in case an application becomes unresponsive.

Operation Identifier	killApp	
Interaction Pattern	SUBMIT	
Pattern Sequence	Message	Body Type
IN	SUBMIT	appInstIds : (List<MAL::Long>)

Structures

a) The appInstIds field contains the list of apps to be killed.

Errors

The operation may return one of the following errors:

ERROR: UNKNOWN

a) One or more of the requested apps to be killed is unknown.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

ERROR: INVALID

a) One or more of the requested apps is not running.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
INVALID	Defined in COM	List<MAL::UInteger>

OPERATION: stopApp

General

The stopApp operation allows a consumer to stop the execution of an application on the provider.

Operation Identifier	stopApp	
Interaction Pattern	PROGRESS	
Pattern Sequence	Message	Body Type
IN	PROGRESS	appInstIds : (List<MAL::Long>)
OUT	ACK	
OUT	UPDATE	appClosing : (MAL::Long)
OUT	RESPONSE	

Structures

a) The appInstIds field contains the list of apps to stop.
b) The appClosing field shall contain the object instance identifier of an app. This update shall be sent after the app acknowledges the reception of the command to stop.

Errors

The operation does not return any errors.

OPERATION: listApp

General

The listApp operation allows a consumer to request the object instance identifiers of the Apps objects and running status for an app name or for a certain app category.

Operation Identifier	listApp	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Type
IN	REQUEST	appNames : (List<MAL::Identifier>) category : (MAL::Identifier)

OUT	RESPONSE	appInstIds : (List<MAL::Long>) running : (List<MAL::Boolean>)

Structures

a) The appNames field contains a list of application names.
b) The category field contains the category identifier to filter on.
c) The appInstIds field contains a list of apps.
d) The running field contains a list of boolean values with the information about thte running status of requested apps.
e) The returned lists shall maintain the same order as the submitted list unless the wildcard value was included in the appNames field request.

Errors

The operation may return the following error:

ERROR: UNKNOWN

a) One or more of the requested apps to run is unknown.
b) A list of the indexes of the error values shall be contained in the extra information field.

Error	Error #	ExtraInfo Type
UNKNOWN	Defined in MAL	List<MAL::UInteger>

Service: Heartbeat

General

The Heartbeat service provides the ability to periodically publish a heartbeat message. Additionally it is possible to get the period of the beat.

Table 0-11: Heartbeat Service Operations

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
SoftwareManagement	Heartbeat	7	6	1
Interaction Pattern	**Operation Identifier**	**Operation Number**	**Support in Replay**	**Capability Set**
PUBLISH-SUBSCRIBE	beat	1	No	1
REQUEST	getPeriod	2	Yes	2

High Level Requirements

a) The Heartbeat service shall provide:
    a. the capability for periodic monitoring of the beat;
    b. the capability for retrieving the period of the beat.
b)

Functional Requirements

OPERATION: beat

General

The beat operation allows a provider to periodically send a heartbeat to the consumers.

Operation Identifier	beat	
Interaction Pattern	PUBLISH-SUBSCRIBE	
Pattern Sequence	Message	Body Type
OUT	PUBLISH/NOTIFY	

Structures

Errors

The operation does not return any errors.

OPERATION: getPeriod

General

The getPeriod operation allows a consumer to get the period of the provider's heartbeat.

Operation Identifier	getPeriod	
Interaction Pattern	REQUEST	
Pattern Sequence	Message	Body Type
IN	REQUEST	
OUT	RESPONSE	period : (MAL::Duration)

Structures

a) The period field shall hold period of the heartbeat message.

Errors

The operation does not return any errors.

Data types

Area data types: SoftwareManagement

ENUMERATION: Checksum

The Checksum enumeration holds common checksums.

Name	Checksum	
Short Form Part	1	
Enumeration Value	Numerical Value	Comment
MD5	1	The MD5 checksum.
SHA1	2	The SHA1 checksum.
SHA256	3	The SHA256 checksum.
CRC	4	The CRC checksum.

Service data types: MemoryManagement

Composite: DataBlock

The DataBlock structure holds a block of data.

Name	DataBlock		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
address	MAL::UInteger	No	The memory address.
data	MAL::Blob	No	The data of the block.
checksum	MAL::Blob	Yes	A checksum for the data field. The checksum algorithm is implementation-specific.

Composite: MemoryArea

The MemoryArea structure holds the description of a memory area.

Name	MemoryArea		
Extends	MAL::Composite		
Short Form Part	2		
Field	Type	Nullable	Comment
address	MAL::UInteger	No	The address of the Memory.
length	MAL::UInteger	No	The length of the block area.

Service data types: SoftwareImage

Composite: ImageDefinition

The ImageDefinition structure holds the definition information of an image.

Name	ImageDefinition		
Extends	MAL::Composite		
Short Form Part	3		
Field	Type	Nullable	Comment
name	MAL::Identifier	Yes	The name of the image.
description	MAL::String	Yes	The description of the image.

Composite: PatchDefinition

The PatchDefinition structure holds the definition information of a patch.

Name	PatchDefinition		
Extends	MAL::Composite		
Short Form Part	5		
Field	Type	Nullable	Comment
name	MAL::Identifier	Yes	The name of the patch.
description	MAL::String	Yes	The description of the patch.

Service data types: PackageManagement

Composite: PackageDefinition

The PackageDefinition structure holds the definition information of a package.

Name	PackageDefinition		
Extends	MAL::Composite		
Short Form Part	7		
Field	Type	Nullable	Comment
name	MAL::Identifier	No	The name of the package.
description	MAL::String	Yes	The description of the package.
category	MAL::Identifier	No	The category of the package.
path	MAL::File	No	The path of the file where the package is located.
publisher	MAL::String	Yes	The publisher of the package. It can be, for example, a company or group of people.

Composite: InstallationDetails

The InstallationDetails structure holds the installation details of a certain package.

Name	InstallationDetails		
Extends	MAL::Composite		
Short Form Part	8		
Field	Type	Nullable	Comment
installed	MAL::Boolean	Yes	The status information about whether a package is/was installed or not.
folderLocation	MAL::String	Yes	To be done...

Service data types: ProcessManagement

Composite: ProcessInformation

The ProcessInformation structure holds the information of a certain process.

Name	ProcessInformation		
Extends	MAL::Composite		
Short Form Part	6		
Field	Type	Nullable	Comment
cpuPercentage	MAL::Float	Yes	The percentage of CPU time.
memoryPercentage	MAL::Float	Yes	The percentage of Memory used.
priority	MAL::Integer	Yes	The priority of the process.
virt	MAL::Float	Yes	The virtual memory used.
res	MAL::Float	Yes	The physical memory used.
shr	MAL::Float	Yes	The shared memory used.
status	ProcessState	Yes	The state of the process.
timeCPU	MAL::Duration	Yes	Total CPU Time.
command	MAL::String	Yes	Command name.

ENUMERATION: ProcessState

The ProcessState enumeration holds a set of process state codes.

Name	ProcessState	
Short Form Part	9	
Enumeration Value	Numerical Value	Comment
Running	1	The running or runnable state.
UninterruptibleSleep	2	The uninterruptible sleep state.
InterruptibleSleep	3	The interruptible sleep state.
Zombie	4	The zombie/defunct stated. Terminated but not reap by its parent.
Stopped	5	The stopped state. Either by a job control signal or because it is being traced.

Service data types: AppsLauncher

Composite: AppDetails

The AppDetails structure holds the details of an instance of an app.

Name	AppDetails		
Extends	MAL::Composite		
Short Form Part	1		
Field	Type	Nullable	Comment
name	MAL::Identifier	No	The name of the app. Must not be empty or wildcard value.
description	MAL::String	No	The description of the app.
version	MAL::String	No	The version of the app.
category	MAL::Identifier	No	The category of the app.
runAtStartup	MAL::Boolean	No	Controls whether the app runs at startup.
running	MAL::Boolean	No	The current running state of the app.
extraInfo	MAL::String	Yes	Additional information that might be implementation-specific.
copyright	MAL::String	Yes	The copyright of the app.
iconPath	MAL::String	Yes	The icon location path of the app. It can be either to a remote link or to a local file.

Error codes

The following table lists the errors defined in this specification:

Table 0-1: SoftwareManagement Error Codes

Error	Error #	Comment
UNSUPPORTED_CHECKSUM	1	The checksum is not supported.
UNKNOWN_DEVICE	711	The device is unknown.
INVALID_ADDRESS	712	The selected address is invalid.

DELETION_FAILED	21	The software image cannot be deleted