File-Based Application Signaling

The OCAP implementation supports an alternate form of application signalling through Java properties files. This format is always used to describe resident host device manufacturer applications (hostapps). For development and testing activities, this format can also be used to represent inband application signalling (AIT) and out-of-band MSO application signalling (XAIT).

The SignallingManager implementation determines whether the stack will use normal network-based (DavicSignallingMgr) or .properties-based (TestSignallingMgr) application signalling. The DavicSignallingMgr implementation (com.vidiom.impl.manager.signalling package), which supports true OCAP application signalling, is the default. You can modify the signalling implementation used by the stack by modifying the following manager definition in the final.properties file:

```
#OCAP.mgrmgr.manager.Signalling=org.cablelabs.impl.manager.signalling.TestSignallingMgr
OCAP.mgrmgr.manager.Signalling=org.cablelabs.impl.manager.signalling.DavicSignallingMgr
```

Just uncomment the manager that you want to use and comment out the other. ***NOTE: File-based signaling is always used for resident application signaling! The signaling manager only describes handling of in-band (AIT) and out-of-band (XAIT) signaling***

Types Of Signaling

There are three forms of file-based signaling. They are:

- AIT
- XAIT
- Hostapps

All signaling files must reside somewhere in the JVM system classpath as defined in the mpeenv.ini file.

AIT

The AIT form is used to describe in-band application signalling. All AIT files must be named as such:

```
ait-<[1-9a-f][0-9a-f]*>.properties

example:   ait-44d.properties
```

There may be one or more AIT files present in the system classpath. As long as a particular service is selected, the implementation will watch for changes to its associated ait-XXXX.properties file (useful for testing signaling version changes). The ait form is comprised of the following information:

- version
- transport protocols
- applications
- external authorization

XAIT

The XAIT form is used to describe MSO abstract services and their associated unbound applications. There can only be one XAIT file in the system classpath and it must be named as such:

```
xait.properties
```

The implementation will watch for changes to the xait.properties file (useful for testing signaling version changes). The XAIT form is comprised of the following information:

- version
- transport protocols
- services
- applications
- privileged certificate hash(es)

Hostapps

The hostapp form is used to describe resident host device applications. The implementation will read a hostapp.properties file at bootup and populate the services database based upon its contents. There can only be one hostapps file in the system class back and it must be named as such:

```
hostapp.properties
```

The hostapp.properties can contain the exact same information as the xait.properties.

Fields

Below of the detailed descriptions of the various fields that can appear in signaling files. The symbol **<i>** indicates that multiple entries are supported (always starting at 0).

Version

A version definition is required to document the version of the application signalling. Signalling updates are only acknowledged with a change of version. If no definition is provided, then the following implicit entry is assumed.

```
version=0
```

Transport Protocols

Transport protocols describe the methods for delivering application content to the stack.

```
transport.<i>=<type>
transport.<i>.<field>=<value>
```

With the following supported types:

- oc
- ip
- ic
- local

field	Description	Type
remote	Boolean; indicates remote protocol	oc, ip
service	Service id, only if remote=true	oc, ip
component	Principal component tag for OC	oc
alignment	Alignment indicator	ip
url.<i>	URL	ip
url	URL	ic

The default maximum of 32 transport protocols and or 32 URLs per IP protocol can be overridden with:

```
maxtps=<max transport protocols>
maxurls=<max URLS per IP protocol>
```

By default, if no transport protocols are defined, an implicit local protocol is defined:

```
transport.255=local
```

The local transport is an RI-specific mechanism that allows you to place apps in your local filesystem instead on OCAP-defined transports. Note that the local transport always has an index of 255, regardless of the index used in the definition. I.e., the app.<i>.transport must reference 255 to get the local transport. At this time, you can not use the local transport in combination with other transports. If you want to use the local transport, you must remove all other transport definitions from your signaling file.

When using the local transport, your application base_directory should be modified to indicate the absolute path to the application base directory in your local file system. On Windows, only use "/" in your pathnames, not "\". Also, you must transform the drive letter designation into something more POSIX-like. <drive_letter>: should become /<drive_letter>. For example:

```
C:\MyProjects\OCAPApps\MyApp

becomes:

/C/MyProjects/OCAPApps/MyApp
```

Services

Abstract service specifications (only valid in xait.properties or hostapp.properties forms) have the following format:

```
svc.<i>.<field>=<value>
```

field	Description	Required/Default Value
service_id	Service id (hostapps: 0x010000-0x01FFFF) (xait: 0x020000-0xFFFFFF)	required
auto_select	Boolean; whether service is autoselect	false
name	Service name	required

If an application definition in a hostapp.properties does not include an app.<i>.service field, then an implicit definition corresponding to the following is included:

```
svc.x.service_id=0x012345
svc.x.auto_select=true
svc.x.name=Default Service
```

Applications

Application specifications have the following form:

```
app.<i>.<field>=<value>
```

field	Description	Required/Default Value	Form
application_identifier	App ID (48-bit hex)	required	all
application_control_code	One of PRESENT, AUTOSTART, KILL, REMOTE	required	all
application_name	English language name	null	all
application_name.<i>	Language-specific name expressed as <lang>,<name>	null	all
service_bound	Boolean; indicates if app is service bound	true	all
version	Integer application version, higher number indicates newer version	0	XAIT, Hostapps
visibility	One of INVISIBLE, VISIBLE, VISIBLE-TO-APPS-ONLY	required	all
priority	Application priority (255=Mon App, 100-254=Unbound, 1-200=Bound)	required	all
icon_locator	Path relative to base_directory for icons	null	all
icon_flags	Icon flags as appropriate for AppIcon	none	all
initial_class_name	Fully qualified name of initial xlet class	required	all
base_directory	Base directory for application	required	all
classpath_extension	Semi-colon separated list of classpath extension directories	null	all
args.<i>	Xlet arguments	none	all

transport	Comma-separated list of transport protocol indices	local	all
service	Abstract service identifier	required for xait; 0x012345 default for hostapp	XAIT, Hostapps
storage_priority	Application storage priority	0	XAIT, Hostapps
launch_order	Application launch order	0	XAIT, Hostapps
app_profiles.<j>.profile	Supported profile (hex or int)	Not required, default 0x102	all
app_profiles.<j>. version_major	Supported profile (hex or int)	Not required, default 0x1	all
app_profiles.<j>. version_minor	Supported profile (hex or int)	Not required, default 0x1	all
app_profiles.<j>. version_micro	Supported profile (hex or int)	Not required, default 0x1	all
api.<i>	Desired "registered api"	none	XAIT
address_label.<i>	Integer; Application adressing labels	none	AIT, XAIT
application_mode	One of LEGACY, NORMAL, CROSSENVIRONMENT, BACKGROUND, PAUSED	LEGACY	AIT, XAIT

Note: registration of an app profile requires profile, major, minor and micro to be defined. Multiple profiles may be registered (starting with index j=0).

External Authorization

Any number of external authorization specifications can be included (the default maximum of 32 can be overridden via a maxauth definition). The specification includes an application identifier and a priority. Application identifiers with an AID of 0xFFFE and 0xFFFF have the same wildcard meaning as for the AIT. The format is as follows:

```
authorized.<i>=<appid>:<priority>
```

Privileged Certificates

Any number of "privileged certificate hashes" may be included (the default maximum of 32 can be overridden via a maxprivcert definition). The privileged certificate hashes may be specified as a single long hexadecimal string representing multiples of 20 bytes or as individual entries.

```
privcertbytes=[<hex>{40}]
or:
privcertbytes.<i>=<hex>{40}

example --  the following are equivalent:
privcertbytes=0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF

privcertbytes.0=0123456789ABCDEF0123456789ABCDEF01234567
privcertbytes.1=89ABCDEF0123456789ABCDEF0123456789ABCDEF
```

Attribute Mapping Descriptor

For addressable X/AIT, any number of attribute mapping descriptors can be specified (the default maximum of 32 can be overridden using a *maxattr* definition). Attribute mapping descriptors are specified using the following form:

```
attribute.<i>.id=<attribute ID>
attribute.<i>.name=<attribute name>
```

Addressing Descriptor

For addressable X/AIT, Any number of addressing descriptors can be specified (the default maximum of 32 can be overridden using a *maxaddr* definition). Addressing descriptors are specified using the following form:

```
addressing.<i>.<field>=<value>
```

The following fields are supported:

field	Description	Required /Default
group	Integer; Addressing group ID	required
label	Integer; Application addressing label	required
priority	Integer; Addressing descriptor priority	required
expression.<j>	See next section	required

Addressing Expression

Each addressing descriptor must contain at least one evaluation expression. At most 32 expressions can be specified for each addressing descriptor. Expressions are specified using the following form:

```
addressing.<i>.expression.<j>=<logicalExpression>

<logicalExpression> := <logical_comp> | <comparison> | <sec_comparison>
<logical_comp> := TRUE | NOT | AND | OR
<sec_comparison> := S <comparison>
<comparison> := attributeID <comparator> attributeValue
<comparator> := < | <= | > | >= | ==
```

NOTE: attributeID is the ID from the corresponding attribute mapping descriptor. Addressing expressions should be specified in the order that they should be pushed on to the operation stack (See OCAP-1.1.1 11.2.2.5.1 Addressing Descriptor)

Maximum Number of Entries

In general, the maximum number of any type of entry is 32 (actually, the maximum index for any type of entry). This can be overridden with an additional property.

property	description	default
maxauth	Number of authorization entries	32
maxtp	Number of transport entries	32
maxurl	Number of urls in an IP transport	32
maxapps	Number of app entries	32
privcert	Number of privcertbytes entries	32
maxattr	Numberof attribute mapping descriptors	32
maxaddr	Number of addressing descriptors	32

Example

Following is an example ait-44d.properties file: **NOTE: This Wiki is putting '[' ']' around the interaction channel protocol URLs. The actual URLs SHOULD NOT be surrounded by '[' ']'**

```
version=1

transport.0=oc
transport.0.component=6

transport.1=ic
transport.1.url=[http://www.ajax.com/appserver]

app.1.application_identifier=0xcafed00d4d01
app.1.application_control_code=AUTOSTART
app.1.visibility=VISIBLE
app.1.priority=200
app.1.application_name.0=eng,Launcher
app.1.service_bound=false
app.1.initial_class_name=com.ajax.xlet.launcher.LauncherXlet
app.1.base_directory=/app/InbandLauncher
app.1.transport=0
app.1.args.0=showMenu

app.2.application_identifier=0xcafed00d4d02
app.2.application_control_code=PRESENT
app.2.visibility=VISIBLE
app.2.priority=180
app.2.application_name.0=eng,Checkers
app.2.initial_class_name=com.ajax.xlet.games.checkers.CheckersXlet
app.2.base_directory=/app/Checkers
app.2.classpath_extension=/lib/ui;/lib/games
app.2.transport=1
app.2.args.0=player=red
```

Following is an example of xait.properties:

```
version=4

transport.0=oc
transport.0.remote=true
transport.0.service=0x3e8
transport.0.component=10

transport.1=oc
transport.1.remote=true
transport.1.service=0x41a
transport.1.component=11

svc.0.service=0x2CAFE
svc.0.auto_select=true
svc.0.name=IMA service

svc.1.service=0x3CAFE
svc.1.auto_select=false
svc.1.name=Games Service

app.0.application_identifier=0x000000016220
app.0.application_control_code=AUTOSTART
app.0.visibility=VISIBLE
app.0.priority=255
app.0.version=0
app.0.service=0x2CAFE
app.0.application_name=IMA
app.0.transport=0
app.0.base_directory=/
app.0.initial_class_name=com.xyz.ima.MonApp

app.1.application_identifier=0x000000016221
app.1.application_control_code=PRESENT
app.1.visibility=VISIBLE
app.1.priority=200
app.1.service=0x2CAFE
app.1.application_name=IMA Helper
app.1.transport=0
app.1.base_directory=/
app.1.initial_class_name=com.xyz.ima.Helper

app.2.application_identifier=0x000000016330
app.2.application_control_code=AUTOSTART
app.2.visibility=VISIBLE
app.2.priority=200
app.2.version=0
app.2.service=0x3CAFE
app.2.application_name=Games Menu
app.2.transport=1
app.2.base_directory=/menu
app.2.initial_class_name=com.xyz.games.MenuXlet

app.3.application_identifier=0xcafed00d4d02
app.3.application_control_code=PRESENT
app.3.visibility=VISIBLE
app.3.priority=199
app.3.application_name=Checkers
app.3.transport=1
app.3.base_directory=/games/Checkers
app.3.initial_class_name=com.ajax.xlet.games.checkers.CheckersXlet
```