# OpenCable APPLICATION PLATFORM – STATUS AND ROADMAP

Allen R. Schmitt-Gordon, Ph.D.
Frank Sandoval
Cable Television Laboratories, Inc.

## ABSTRACT

*The OpenCable Application Platform (OCAP™) is a software middleware layer that resides functionally on top of the operating system of an OpenCable™ terminal. It provides an interface and enables application portability. A fundamental requirement is that applications written for OCAP be capable of running on any OpenCable hardware, without recompilation.*

*Two profiles of OCAP have been identified. OCAP 1.0 is a minimal platform that supports procedural applications with an Execution Engine (EE). OCAP 2.0 is a super set of OCAP 1.0, and includes support for declarative content with the inclusion of a Presentation Engine (PE), that supports HTML, XML, ECMAScript, and a bridge between the PE and the EE. The bridge enables PE applications to obtain privileges and directly perform EE operations.*

*OCAP 1.0 has been publically released, OCAP 2.0 is scheduled for around 1Q02. CableLabs plans to draft a family of OCAP specifications, each being backward compatible and defining different feature sets.*

*The OCAP specifications are based upon the DVB MHP specifications with modifications for the North American Cable environment that includes a full time return channel. OCAP 1.0 corresponds to MHP 1.0.2, and OCAP 2.0 will correspond to MHP 1.1.*

## INTRODUCTION

OCAP is part of a concerted effort, called OpenCable, by North American cable operators to provide the next generation digital device, encourage supplier competition, and create a retail hardware platform. A cable receiver provided at retail must provide portability of content and applications across networks and platforms, and be geared towards the full range of interactive services. Current devices are network specific and operate proprietary software that is not portable across platforms or networks. With OCAP, applications are written to a middleware API so that a single application can be deployed on any OpenCable host device. Such applications might include:

- Electronic Program Guide (EPG)
- Impulse Pay Per View (IPPV)
- Video On Demand (VOD)
- Interactive sports, game shows
- E-mail, Chat, Instant messaging
- Games
- Web Browser: Shopping, Home banking
- Personal Video Recorder (PVR)

OCAP provides applications an abstracted view of the receiver and network, hiding vendor and network specific characteristics that would tie an application to a given system. OCAP is operating system and hardware agnostic, so that applications can be run on a variety of CPUs and operating systems. The OCAP middleware also simplifies content development by encapsulating common operations within the API. Another essential requirement is that the middleware be secure and robust. Stability in the cable terminal or receiver is imperative as resets are not acceptible.

## Background

The OCAP specifications include Application Programming Interfaces (API) as well as definitions regarding platform behaviors. The APIs define the syntax of the platform, while normative text define semantic guarantees.

The architecture of the OCAP 2.0 middleware comprises two parts: a Presentation Engine (PE) and an Execution Engine (EE). The PE is generally composed of an HTML engine and ECMAScript. The EE includes a Java virtual machine. This architecture is shown in Figure Figure 1. It shows that native applications are supported as well as applications written to the middleware via the OCAP interface.

In order to expedite development of the OCAP specifications, it was necessary to utilize existing standards and architectures as much as possible.

A key decision was to base the OCAP EE on standard Java APIs. In particular, the OCAP EE comprises pertinent portions of Sun's Java Virtual Machine (JVM) and JavaTV API specifications. The Sun technology is licensed by CableLabs and is available to all implementers of OCAP royalty-free. CableLabs will also incorporate the Sun Technology Compatibility Kit as part of the OCAP compliance test suite.

Another key decision was to adopt the DVB project's Multimedia Home Platform (MHP) specification. The DVB Project is a European consortium of manufacturers, content developers, broadcasters, governmental agencies, and system operators. MHP is designed to apply to a wide set of networks and devices, including cable networks. CableLab's recognized that DVB was trying to solve a similar set of problems. OCAP 1.0 is a delta from MHP 1.0.2, that is, the OCAP 1.0 document identifies conformance with MHP 1.0.2 section by section, and includes material to identify differences and extensions.

## **OCAP Roadmap**

### Rationale for Middleware

The current software model used by the cable industry is similar to the more general pattern in which applications are developed for a specific operating system (OS) and compilied for a specific device. This model does not lend itself to application portability where a variety of devices can be attached to several different network infrastructures. Currently, applications such as EPG, VOD, mail, etc. are compiled to the application programming interface determined by the operating system and associated hardware.

By providing a middleware layer that abstracts the functionality of the OS, hardware device and network interfaces, applications can be written that will run on any conformant platform. Java was chosen in order to avoid recompilation of source code as would be the case with languages such as C or C++.

### OCAP 1.0

The primary architectural component of OCAP 1.0 is the Execution Engine (EE). It is composed of a Java Virtual Machine and a set of Java packages. Application portability is achieved through the "write once, run anywhere" nature of Java. Java source code is compiled into bytecode, and the JVM interprets the bytecode in real time on the target machine. Once the JVM is ported to a target machine, any EE application will run. The Java packages that comprise the EE include basic support functionality from Sun, such as net, io, util, and lang packages, and JavaTV and Java Media Framework. The EE also includes Java APIs identified in DAVIC, HAVi, and MHP 1.0.2. In addition, there are OCAP specific APIs.

This collection of APIs allows access to system resources, such as the cable tuner and input events, and access to network resources, such as System Information, in-band and out-of-band resources, and IP flows.

Because of the interpreted nature of Java, application security is easily maintained. Applications by default are given limited access to system resources. An application can gain extended access by requesting resources via a permissions file, and by being authenticated by the operator.

OCAP 1.0 specific APIs have been designed to address the business requirements of the cable industry. Unbound applications can exist outside of the context of a given service, or channel. Also, an optional monitor application can be created by the system operator. The monitor application allows to operator to control the execution lifecycle of other applications, and perform some simple resource management functions, in order to prevent harm to the cable network.

## OCAP 2.0

OCAP 2.0 refers to OCAP 1.0 for core functionality and adds features found in MHP 1.1. A Presentation Engine and Bridge are included to support so-called declarative applications. The PE renders content such as graphics, text, animations and audio based on formatting rules in the PE itself and formatting instructions contained in markup language. The PE enables the use of tools that have been widely used for internet content.

OCAP 2.0 primary components consist of XHTML 4, XML 1.0, CSS 2, DOM 2, and ECMAScript.

In order to extend the functionality of the PE without adding the burden of requiring extensive plugins, a bridge is defined between the two environments. The bridge allows PE applications, through ECMAScript calls, to access functionality defined in the EE. Conversely, EE applications, through the DOM interface, can interact with concurrently running PE applications.

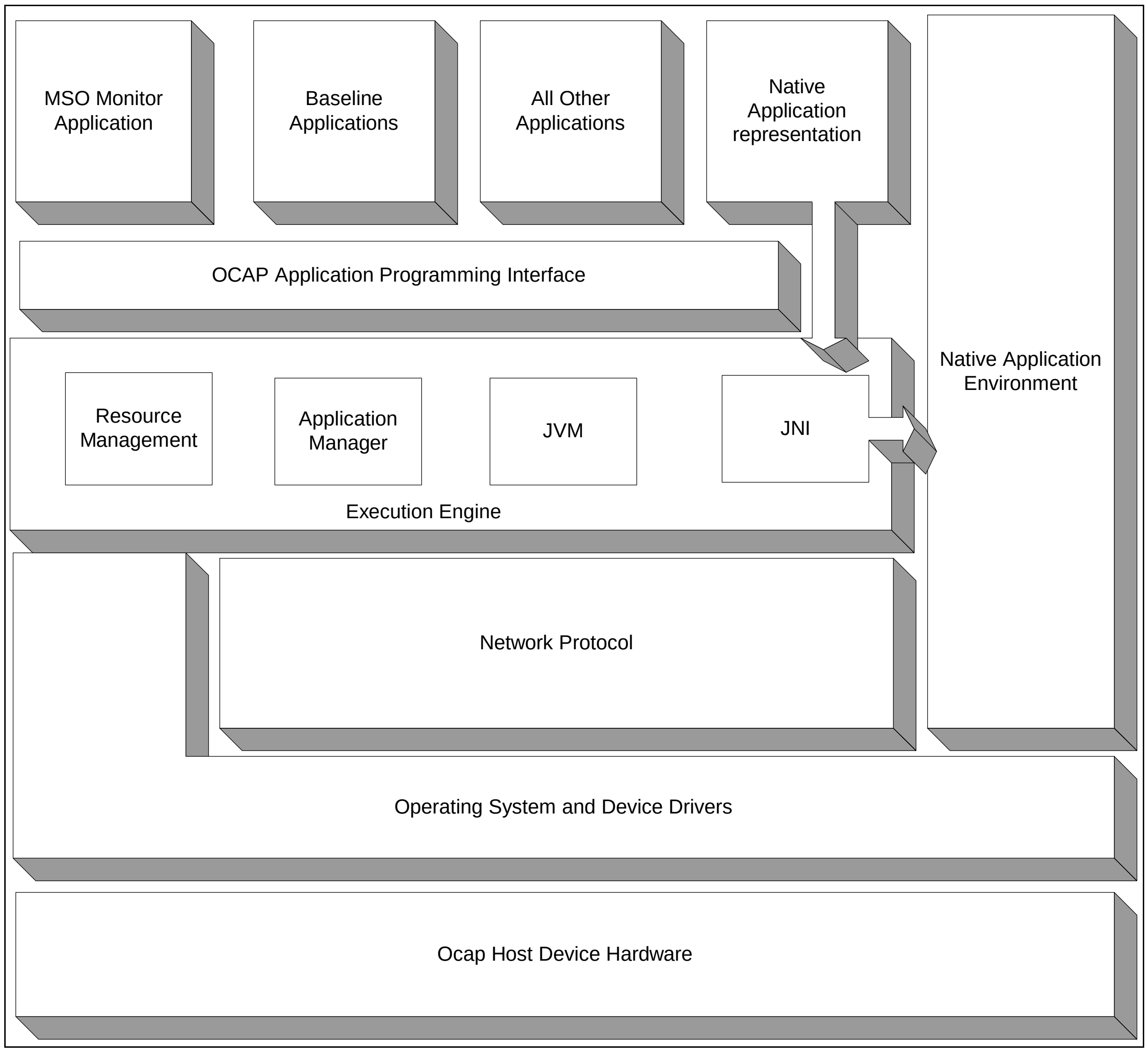
MSO Monitor Application
Baseline Applications
All Other Applications
Native Application representation
OCAP Application Programming Interface
Native Application Environment
Resource Management
Application Manager
JVM
JNI
Execution Engine
Network Protocol
Operating System and Device Drivers
Ocap Host Device Hardware

**Figure 1 - OCAP 1.0 Software Architecture**

For example, as part of the EE that is accessible via the bridge, JavaTV offers a common point of control and management of various system resources, includeing tuning. Thus, a PE application will access device resources through the bridge. This ensures that device resource contention is managed through a common control point for a PE and EE application that may be vying for the same resources.

Execution Engine

The EE provides a general application programming evironment for networking, file I/O, graphics, etc. The OCAP EE provides a full TV application environment. OCAP specific extensions have been added to MHP to cover elements such as a full time return channel, application management, resource contention mangement, and service information.

Major elements of the EE include control of application management through the pJava APIs, service information and selection through the JavaTV API's, media control through the JMF, and broadcast data through the MHP DSMCC APIs. Native applications are supported by creating an OCAP application that calls into native code via the Java Native Interface (JNI). In addition, the EE provides network management and IP data access and extensions from HAVI, DAVIC and DASE.

A fundamental feature of the EE utilizing Java is that security is built into the architecture from the ground up.

The components of the EE, some of which are shown in Figure 1, are described in detail in the OCAP 1.0 specification. In order to understand one of the key features of OCAP 1.0, which distinguishes itself from MHP 1.0.2, a discussion of the OCAP Application Model is warranted.

Application Model

OCAP 1.0 relies very heavily on the Application Listing and Launching APIs defined by DVB-MHP. This set of APIs enables lifecycle management of those applications that are bound to a service or program. OCAP 1.0 extends this model to include management of unbound applications or those applications that are not bound to a service or program. Signalling of such applications is done by a mechanism similar to the AIT specified by MHP 1.0.2, called the XAIT that is delivered via the traditional OOB or the DOCSIS channel (when available).

A special unbound application is called the Monitor Application The monitor application has a number of specific capabilities that can over-ride baseline functionality of the the OCAP 1.0 implementation (see Figure 1).

This functionality includes:

- Registration of unbound applications with the applications database.
- Validation of the starting of all applications through the setting of application filters.
- Registration of system errors that are propagated from the OS middleware, and/or OCAP 1.0 implementation, including OCAP applications.
- Request system reboot, and regitration of system reboot event.
- Control of copy protection bits and output resolution using the org.ocap.application.CopyControl interface.
- Filtering of User Input events and change their value before sending them to their final destination.
- Management of storage of any MSO unbound application, including itself using the persistent storage API, as defined by MHP 1.0.2 .
- Application lifecycle management, including that of the Monitor Application
- Resource Contention management of resource contention deadlocks.

If the Monitor Application implements any or all of the above functionality, it over-rides the implementation on a per MSO basis. Otherwise, the OCAP 1.0 implementation performs these functions in a generic manner.

OCAP-MHP Comparisons

The table in the appendix was generated by enumerating all of the java methods specified by the OCAP 1.0 and MHP 1.0.2 specifications. OCAP 1.0 shares a number of java packages with MHP 1.0.2. These packages are listed in section 12 of the OCAP specification. The packages come from SUN java JDK 1.1.8, PJAE 1.2, JavaTV, JMF, DAVIC, HAVi, and DVB-MHP. It is a useful but daunting task to enumerate the methods and interfaces that both OCAP 1.0 and MHP 1.0.2 share in common. Utilizing the javadoc tool and the javadoc APIs provided by SUN with the

JDK, the task is made somewhat less difficult. Doclets were written utilizing the javadoc APIs to produce custom javadoc output that list and count the methods in each package. These results are summarized in the table attached hereto as an appendix.

It should be noted that the number of MHP-specific methods is less than 10% of the total number of methods specified by MHP 1.0.2. Thus, even if OCAP 1.0 did not utilize any of the MHP-specific methods, it would still be highly compliant with the MHP 1.0.2 specification, with an 89% correspondence. The SUN java, javaTV and JMF components represent 71% of the total number of methods. However, the methods specified by OCAP 1.0 and MHP 1.0.2 add the useful and necessary functionality needed by applications.

Both specifications share the following packages:

org.davic.media
org.davic.mpeg
org.davic.mpeg.sections
org.davic.net
org.davic.net.dvb
org.davic.resources
org.havi.ui
org.havi.ui.event

OCAP 1.0 does not currently utilize the following packages. Ongoing efforts to harmonize OCAP with MHP may effect this list:

org.dvb.net.ca
org.dvb.net.tuning
org.dvb.si
org.davic.net.ca
org.davic.net.tuning

The org.dvb.event package is not currently specified by OCAP 1.0 but is under consideration for addition to the specification via the ECR process.

## Conclusions

The OCAP family of specifications were developed in response to specific needs of the cable industry. In order to minimize time to market for new services, and to enable downward price pressure and facilitate innovation via a retail market for cable receivers, OCAP provides a full featured and robust software platform. It offers a very high degree of portability and uniformity for content display as well as offering a platform for the broadest possible range of application support. The OCAP architecture ensures security and robustness.

## References

Schmitt-Gordon, A., OpenCabel Application Platform Architecture. Proceedings NCTA, June 2001.

OpenCable Application Platform Specification, OCAP 1.0 Profile, OC-SP-OCAP1.0-I01-011221, found at http://www.opencable.com

Kar, M.L., Vang, S., and Brown, R., Architecture of Retail Set-Top Box Application Platform for Digital Cable Netork, Proc. ICCE, June, 2001.

Zundel, J-P, Emergence of Middleware in Home Telecommunication Equipment, IEEE Communications, June, 2001.

Draft Multimedia Home Platform Draft 1.0.2 for review, Document TM2208r7, TAM 232r29, can be found at http://www.dvb.org

Digital Video Broadcasting (DVB) Multimedia Home Platform (MHP) Specification 1.1, tm 2485, tam 668r12, can be found at http://www.dvb.org

ATSC DASE AEE, Doc. T3-530 09, Feb 2001, Rev 1.

Havi Level 2 User Interface, section 2.5.2, http://www.havi.org

Documents relating to the PE including DOM, CSS, HTML can be found at http://www.w3c.org

Documents relating to the Davic specification can be found at http://www.davic.org

## Appendix – Syntactical Comparison of OCAP 1.0 and DVB-MHP 1.0.2

| PACKAGES | total methods | # methods used by OCAP | # methods used by DVB-MHP | syntactical compliance with DVB-MHP % |
|---|---|---|---|---|
| org.dvb.* | 488 | 357 | 488 | 73 |
| org.ocap.* | 207 | | | |
| javax.* | 414 | 414 | 414 | 100 |
| java.* (see note 1) | 3440 | 3440 | 3440 | 100 |
| org.havi.* | * | 785 | 785 | 100 |
| org.davic.* | * | 139 | 268 | 52 |
| | | | | |
| TOTALS | | 5135 | 5395 | 95 |
| note 1 - number of java.* methods actually used by DVB-MHP is somewhat less than required by SUN for the implementation | | | | |