# Open Middleware and the OpenCable POD Module: Versatile Solutions for Portable, Secure Digital TV

Anthony J. Wasilewski
Scientific-Atlanta, Inc

## *Abstract*

*Two aspects of a secure, portable environment for interactive DTV services are discussed. The OpenCable POD and an open middleware approach can be useful tools on "the road to retail".*

The OpenCable Point of Deployment (POD) module is an important component of the open standards specifications for digital TV. The POD supports the total separation of the conditional access system from the host terminal while still supporting a wide array of features and applications and providing high performance video/audio/data services. By allowing host terminals to become more generic, the POD may play a critical role in any transition to retail availability of set-tops.

This paper will cover the following aspects of the POD:

- Brief History/Origins
- Regulatory Issues and Timelines
- Relevant Standards & Documents
- Architecture and Features
- Interface Descriptions
- Copy Protection

To complete the support of an open platform for hosting of a rich set of portable services, the application software environment of the host also needs to be standardized. The goal is to provide an environment in which a large measure of freedom to craft applications with varied feature sets and powerful graphics exists, while also fostering a high degree of portability of those applications to different hardware platforms.

To this end, this paper describes an open set of middleware that includes:

- HTML
- Javascript (ECMAScript)
- MIME
- Personal Java
- XML
- HTTP
- SSL
- DOM
- XHTML
- ATVEF

These middleware components can be and/or are deployed on existing digital set-tops in currently launched systems.

## <u>POD History/Origins</u>

The concept of using a removable device to encapsulate all security and conditional access processing has origins in both the DVB (Digital Video Broadcasting) Common Interface process in Europe and the NRSS (National Renewable Security Standard) effort carried out as a joint engineering committee of CEA and the NCTA under EIA sanction in North America. Both groups eventually adopted a PC Card (PCMCIA) form factor although the NRSS specification also includes an extended ISO 7816 smart card format as an additional choice.

The POD extends both the DVB and NRSS standards by adding:

- explicit handling of out-of band (OOB) data channels
- a copy protection mechanism
- an application interface
- extensions for cable-ready applications

## Regulatory Issues and Timelines

From a regulatory viewpoint, the major influences on the existence of the POD have been:

•The Telecommunications act of 1996 requires that cable subscribers be given the option of owning the equipment required to receive cable services

•The FCC's Report and Order (63 Fed. Reg. 38095) requires that cable operators make separable security modules available by July 1, 2000 in order to facilitate commercial sale of navigational devices

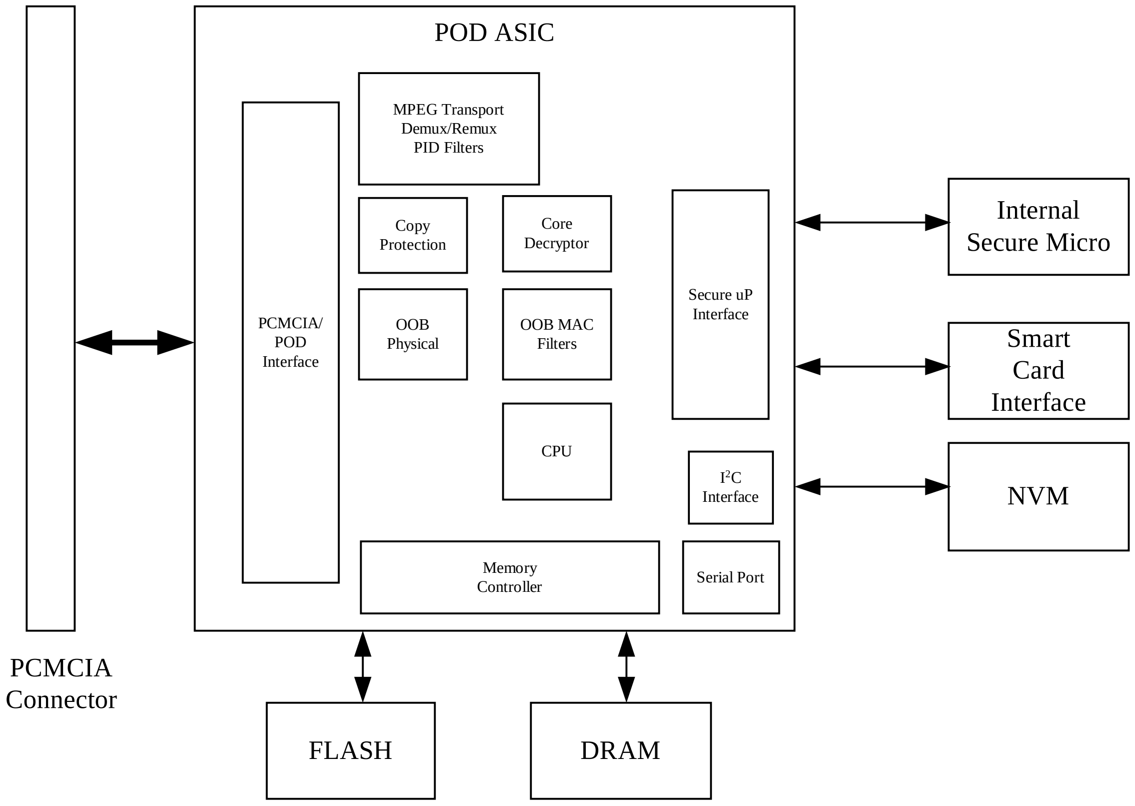
The industry responded by including in the CableLabs® OpenCable® process, a working group to define the functionality for a Point-of-Deployment or POD module and its interfaces and also to specify a suitable copy protection method that would be acceptable to content owners. This work has progressed well and manufacturers are responding to the challenge in a manner that should result in suitable product being available in the FCC-mandated time frame.

## POD Architecture and Features

The POD handles both in-band MPEG Transport and Out-of-band (OOB) control channels on behalf of the host device. One of its primary responsibilities for MPEG processing is the conditional access-level decryption of subscriber-selected content for which the subscriber is authorized. The POD accomplishes this in conjunction with the CATV headend through the conditional access system that is implanted within it. For the OOB channel, it interprets the format of the control bit stream that has been sent to it from OOB RF receivers in the host.

The POD must support five different interfaces: PCMCIA, MPEG Transport, an out-of-band channel (either DAVIC or Motorola (formerly GI)), the POD data channel and the POD extended channel. It must support traditional applications such as digital broadcast and IPPV but must also be capable of other OpenCable services such as VOD. It must also be able to support the man-machine interface (MMI) of host applications utilizing a graphical interface based on HTML 3.2.

A basic block diagram for a POD module is shown below:



To fit the amount of functionality the POD specifications require into a small form factor such as Type II PCMCIA, considerable integration in silicon is called for. Thus, much of the major components in the POD block diagram above are found on an ASIC. This chip, of course, has interfaces to memory and other chips such as secure microprocessors to complete its mission.

## Interface Descriptions

The following are the classes of interfaces supported by an OpenCable POD module:

- PHY
- Extended Channel
- Link Interface
- Application Interface

The *PHY or physical interface* is compliant to the 68-pin PCMCIA interface. This supports parallel, full-duplex transmission of MPEG-2 Transport streams at the bit rates typically deployed in North American CATV systems. There is also support for signaling and CPU-to-CPU communication. Upon power-up, the POD performs the standard 16-bit PC Card Memory Only initialization, after which the POD and Host activate the "POD Module Custom Interface" which has a registered interface ID of hexadecimal 341. The POD also follows PC Card power management standards.

The *Extended Channel* provides a data path between the POD and Host for information flows outside the MPEG-2 Transport that are not terminated by the POD. Thus, for example, it supports the flows of IP packets or MPEG sections that have arrived for the Host via out-of-band (OOB) pathways from the headend. Some data, such as Entitlement Management Messages (EMMs) from the CA system, arrive over the OOB pathways and are not forwarded by the POD to the Host.

The POD *Link Interface* is compliant with the Command Interface of the NRSS Part B specification. This implements a set of protocols that establish communication about and to *resources* relating to conditional access, host control, the man-machine interface (MMI), copy protection, generic IPPV support and the Extended Channel. The link interface takes care of identification of flows and Protocol Data Unit (PDU) fragmentation.

The *Application Interface* is used to support "cable-ready" applications that use the data channel that are not defined or adequately covered by NRSS Part B. Such functions include Host Generic Feature Control, POD Emergency Recovery and specific application support. This interface defines the POD/Host MMI, additions to the low-speed communication interface, additions to the host control resource, additions to extended channel support, modifications to the generic IPPV resource and specific application support for hosts that have software download capability.

## Copy Protection

Because the POD processes (decrypts) and passes digital content streams to the host, suitable copy protection is required. If these content streams were sent "in the clear" over the PCMCIA interface to the host, it would be relatively straightforward to make exact digital copies of this content. Thus, to protect the bit streams as they flow from POD to host, the POD must apply additional encryption. While this is fairly simple to do, the POD is also required to authenticate the host to verify that it has not been previously identified as an illegal device. This is accomplished through the use of digital certificates and signatures. In this manner, a highly secure form of key exchange may also be practiced.

Once the POD and Host have agreed on keys to be used, the POD encrypts the content of the MPEG-2 Transport packets that require copy protection as signaled in CCI (copy control information) bits which are sent in authenticated form via the conditional access system.

## Open Software Environment

The POD module only provides part of the solution for a portable application and content environment. There must also be a method that provides content and application interoperability.

An open software environment relies on published standards that enable portability between platforms. If we begin with that assumption that the TCP/IP protocol suite will be the foundation for messaging, client-server

TCP/IP protocol suite will be the foundation for messaging, client-server applications and data access and that MPEG video and transport and Dolby AC-3 audio are the foundations for transmission of entertainment content, then portability concerns need to be focused on other aspects of the system. There are at least three other major areas of support needed to enable the desired portability:

a) Content rendering
b) Application code execution
c) Network protocols

These areas can be covered by an architecture that includes standardized elements for the following functions:

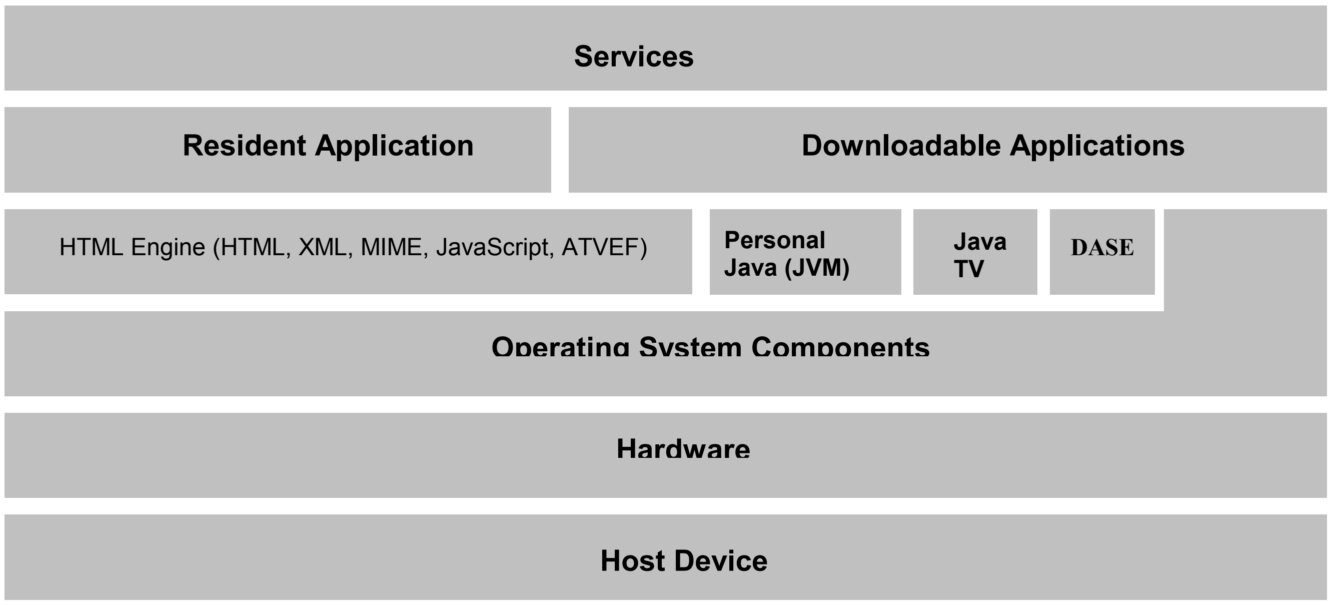
- Presentation Engine
- Application Engine

In addition, the platform would require elements that provide the following functionality on behalf of applications:

- Network services
- Platform services

Ideally, these network and platform services would be supplied via an *operating system* that, in conjunction with the middleware, completely abstracts the details of the underlying hardware platform and network and permits applications to migrate between platforms with little or no modification. The operating system may or may not be a standardized element, however, it is likely that, within any one MSO system, only one operating system would be deployed.

A layered model of an interoperable software environment for a Host device is shown in the figure below. In the figure, the *middleware* components are located between the applications and the operating system (i.e., in the "middle", hence the term "middleware"). The foundations of the middleware, are the *presentation engine* components: HTML, MIME, ECMAScript, ATVEF, DOM) and the *application engine:* PersonalJava. Examples of *extensions* to the middleware are the Java.TV classes and the ATSC Digital TV Application Software Environment (DASE) which has been defined by the S17 Group of ATSC.


Services
Resident Application
Downloadable Applications
HTML Engine (HTML, XML, MIME, JavaScript, ATVEF)
Personal Java (JVM)
Java TV
DASE
Operating System Components
Hardware
Host Device

The *multi-purpose Internet mail extensions (MIME)* provide useful standards for content rendering such as JPEG and *Portable Network Graphics (PNG)* and audio standards, such as WAV and AIFF.

The *Document Object Model or DOM* is a platform and language-neutral model that allows programs and scripts to dynamically access and update content. The DOM provides a map of a document's structure and style and supports generic access to its parts. Combining DOM0 with ECMAScript is equivalent to Javascript 1.1.

The Advanced Television Enhancement Forum or ATVEF is a cross-industry alliance of companies that have defined protocols for HTML that can be used to deliver enhanced programming over many transports to a range of intelligent receivers. These enhancements include announcement protocols, trigger handling for real-time events and a local identifier URL scheme.

XML or *extensible markup language* is a very promising addition to the markup language approach started in HTML. XML supports the separation of the definition of the data from the description of how it should be displayed. This promotes more dynamic content as well as providing strong portability since the rendering can be specified for TV-based graphics, print-oriented graphics, speech synthesis or even *Braille* without changing the content coding at all. It also supports domain-specific data definitions that can be used to formulate standardized formats for specific types of data, so that all applications can use and interchange the *same* data.

Further capitalizing on this separation of content definition and rendering is XHTML (*Extensible Hypertext Markup Language).* XHTML is a reformulation of HTML 4.0 as an application of XML 1.0. It has the advantage of being easily extensible, which allows applications to be updated with relatively little effort and it is designed to be highly portable, so that content can be transferred to many diverse platforms (such as PCs, cell phones, PDAs, TVs, etc.) and be acceptably displayed without modification.

The *Application Engine* is a complete execution environment within itself. One example is PersonalJava® or pJava, one of the Java® application environments developed by the Javasoft division of Sun Microsystems, Inc. Java is implemented on a Host device as a virtual machine. This is a software program that executes byte codes, which are standardized instructions for the machine. As long as a Java Virtual Machine (JVM) is available for a platform, applications written in the Java language can be readily ported to that platform. The JVM also provides a security framework to ensure that "renegade" applications do not wreak havoc on the host platform.

The *operating system components* supply the interface (and abstraction of) the *hardware components* of the Host device. Thus, applications and the middleware layer need not concern themselves with the details of different types of tuners, on-screen display graphics drivers, conditional access elements, and other vendor-specific components. Also, the operating system must provide a robust event model and the facilities to handle the event-driven aspects and requirements of applications. These include display focus, inter-process messaging, timers, semaphores, memory management and the like.

Finally, the operating system must also support important networking protocols such as TCP/IP, HTTP and SSL. Hypertext Transfer Protocol is one of the foundations of the World-wide Web and the Secure Sockets Layer has rapidly become the *de facto* Web protocol for securing communications between clients and servers.

## Conclusion

Critical parts of the support of the portability required to support retail availability of Host devices in CATV systems are supplied by the OpenCable POD module and by open approaches using standard middleware. Combined with a robust operating system that abstracts the details of the platform hardware and provides system services to the middleware and applications, the basic foundation of interoperability can be formed. The POD module supplies complete separation of conditional access functions from the Host device and copy protection functions that are acceptable to the content industry. An open middleware approach provides additional standardization and abstraction for content rendering and the application execution environment.

Using this foundation, application developers can produce support for new services, confident that their efforts will be applicable to a wide range of Host platforms.

## References and Standards

The following documents are some useful references that provide additional reading and information regarding the OpenCable POD and open middleware environments and standards:

| | |
|---|---|
| 1 | Document markup language HTML 4.0: http://www.w3.org/TR/REC-html40/ |
| 2 | Document scripting language ECMAScript: http://www.ecma.ch/stand/ecma-262.htm |
| 3 | Document Object Model DOM Level 0: http://www.w3.org/DOM |
| 4 | Hypertext Transfer Protocol (HTTP) 1.1 (RFC 2068): ftp://ftp.isi.edu/in-notes/rfc2068.txt |
| 5 | Aggregation & encoding of multiple resources into a single resource for delivery:<br><br>MIME multipart/related: http://info.internet.isi.edu/in-notes/rfc/files/rfc2387.txt<br><br>MIME HTML (rfc2110): ftp://ftp.isi.edu/in-notes/rfc2110.txt |
| 6 | Extensible Markup Language (XML) 1.0 Specification, http://www.w3.org/TR/REC-xml |
| 7 | OpenCable Host-POD Interface Specification, IS-POD-131-INT01-991027 |
| 8 | NRSS Part B Specification, EIA-679-A, Part B |
| 9 | OpenCable POD Copy Protection Specification: IS-POD-CP-INT01- 000107 |
| 10 | Java Specification: http://www.javasoft.com/aboutJava/communityprocess/maintenance/JLS/index.html |
| 11 | ATVEF Specification: http://www.atvef.com/library/spec1_1a.html |

*Author's Contact Info:*
Tony Wasilewski
5030 Sugarloaf Parkway
P.O. Box 465447
Lawrenceville, GA 30042
Phone: 770-236-5004
Fax: 770-236-3080
E-mail: tony.wasilewski@sciatl.com