Grass Valley Group ContentShare Platform

Overview (Developer's Addendum)

Dave Slack, Program Manager

ContentShare is a registered trademark of Grass Valley Group. All other trade names referenced are the service marks, trademarks or registered trademarks of their respective companies

This document is intended to give software developers and integrators a more applied discussion of the ContentShare™ platform and the features it brings to the industry. For a more general overview of what the platform does and how it operates, refer to the *Grass Valley Group ContentShare Platform Overview*.

Introduction

Software developers are facing challenges as customers are expecting workflow efficiencies to increase and operating costs to decrease. This is being fueled in many cases by the vision that new hardware devices such as videodisk recorders and advanced software will make changes in operations possible.

In many cases, the efficiencies are not being gained by replacing one product with another, but by replacing standalone products with products that allow integration of the process.

A common example is the videodisk recorder. When looked upon as a standalone device on a channel-by-channel basis a Video Disk Recorder (VDR) has little more to offer than a videotape recorder. In some respects, even less. When you combine a VDR with features like multiple channels, storage options like expanded disk and data tape, and networking like Fibre Channel that the VDR exceeds the VTR in its usefulness. When combined with these modern features it still requires a high level of software to complete the tasks that gain operating efficiencies.

The broadcast software industry is similar in many respects. Any number of existing processes within a television facility can be modernized with software but do nothing to increase the efficiency of the facility. Customers are beginning to realize that for any particular device or software application to add real value to their facility it must be able to fit into, and work within, a much larger system.

Few manufacturers believe they can provide the complete solution for every portion of the facility, from the newsroom to production to on-air. Most have concentrated on a single portion of the facility and have developed products primarily for that area. However, most customers have more than just a single area and increasingly demand products that function together.

For software developers this poses a challenge. Few believe they can provide a solution for every area. They are realizing that to be successful their products must work with other products. This realization is not come to lightly. Working with other products usually means a custom interface to third party products. Several factors must be considered when undertaking such a development effort. The overall opportunity must be properly sized so that the product combination will actually sell in sufficient volume to realize a profit. Developers may not be able to develop multiple interfaces in a given amount of time so, the decision of *which* third party can become critical. It is not inconceivable that products that may have been looked upon as a perfect match by one customer may not be suitable to another. Further, it is expected that both companies' products will continue to evolve which often requires both interfaces to evolve as well, even if only for the purposes of compatibility. While two companies can usually provide more of the answer to the customer's problem; often a complete solution requires more. The complexity of this solution increases exponentially with the number of companies involved.

ContentShare Platform Solutions

The ContentShare platform can be looked upon as a framework to solve many integration difficulties. The benefits that the ContentShare platform brings to the developer are much the same as those detailed in the *Grass Valley Group ContentShare Platform Overview*.

A general understanding of the concepts discussed below can be gained from the *Grass Valley Group ContentShare Platform Overview.* More in-depth technical information is available in the *ContentShare Platform Technical Overview*.

Open Communication for Asset Sharing

There are a number of features important to software development that are a direct result of opening communication.

Single Platform Interface

The broker interfaces that developers create to represent their software to the rest of a ContentShare-based system abstracts them from one another. Any other software tool on a system that is seeking out information goes about this through a User Agent. This agent communicates with the brokers that represent the application. Any application that has a broker representing it can offer information and assets to a User Agent. So, for a developer to make their information and assets available to the rest of the system, they only need create a broker interface. They do not need to create a direct interface to every other software application or tool in the system.

Conversely, for a software tool to get information *from* other applications it only needs a single User Agent. This agent is capable of communicating with any broker on the system. Again, only a single interface is necessary rather than a custom interface for every application.

The ability to integrate with multiple applications via a single interface means that the return on development is far greater than if multiple interfaces were employed. Also, there is less software to maintain with a single interface.

Software Changes

Because the agent and broker interface shields applications from one another, changes made by any application do not require change by any other application. For example, an application could change its underlying database from one type to another and by changing the broker to match, the rest of the system would be unaffected. This would require no change to any User Agent in the system. The change is isolated to the developer that initiated it.

Further, because the brokers are self-announcing a new application can join a system without requiring all the other software on that system to make changes to take advantage of that application.

Extensible Platform

Vendors can not anticipate all the kinds of assets or metadata that may be necessary in the future. Because of the extensibility of the underlying XML-based messaging framework, new asset types can be added by applications without requiring the framework to change.

Locating Assets

Typically, assets and material are transferred from one vendor's application to another throughout the production process. While pushing media from one application to another may be suitable in small systems, it is inevitable that one application will need to search into another's assets or information.

The ContentShare platform allows applications to use a single search tool to search *any* repository of information that is part of the ContentShare system. In other words, an application's search tool can search into multiple sources of information on a ContentShare system using a single User Agent interface. In fact, the search tool does not need to know what the source of information is.

Conversely, any application that wants its information or assets to be available to anyone else makes that known once through a single interface. It is then available to all others on the system. Again, with the ContentShare architecture, an application can collaborate with many other applications without having to write custom interfaces. This greatly speeds development time, and decreases development and support costs.

Linking Assets

Multiple software applications in a facility often means multiple types of assets. For instance, it is common to find high-resolution video, news scripts, archived material, wire feeds, etc. all within a single facility. Even a single asset type, such as high-resolution video, may be managed by multiple applications within a facility.

Often, many of these asset types are related. The edit system may be using video that was recorded by the automation system to be used within a news package for which there is a news script. Traditionally, links between these different kinds of assets have been possible only within a single system. The ContentShare platform provides a universal method for linking all of these assets together.

This makes the job of overall facility media management much simpler. The automation system can find out if any other applications have copied or based material on an asset from its system. It can then decide whether to delete, archive, etc. based upon that knowledge.

For example, an editor who wants to find high-resolution material can use the text information found in a news script as a link to video. They can see if a low-resolution browse copy is available and preview it prior to requesting the material be transferred from archive data tape. The link between the news script and video may be created and maintained solely by the newsroom system. The newsroom system decides whether to expose those links through the ContentShare platform.

The ContentShare platform provides a revolutionary method to connect applications together that not only benefits customers but also developers. It can increase the return on development investment and product opportunity for developers of even the smallest application.

Benefits for Specific Applications

Developers can benefit from the different types of applications they can develop using the ContentShare Platform. In this section, we discuss some of the benefits and application types.

Editing Application Developer

- You can quickly incorporate global asset management features into your application. Such as:
  - Global Search
  - Asset Preview (low-resolution & catalog)
  - Asset Transfer
- You can expose information and material created by your edit system to systems integrators and other developers. This increases the market for your product.
- You can develop applications providing proxy editing of low-resolution browse material.
- The ContentShare platform enables your product's material to be easily managed by a system administrator at a central location.

Control System Developer, such as Automation and Newsroom Systems

- Your system can now integrate with other control systems, editing applications, storage devices and other data sources that are ContentShare-based. You can write to a single, simple API and gain access to all assets made available by ContentShare developers. ContentShare's self-describing data representation enables your software to understand metadata from third party systems without requiring inherent knowledge of the data.
- You can expose information managed by your system and at the same time protect access to that information via ContentShare's security framework.
- You can connect your own systems across distributed locations for the purposes of global asset search and transfer.
- Your product can easily incorporate existing, legacy data sources. You can focus on building next-generation systems and spend less time trying to interface to custom applications.
- You can develop a Transfer Broker that exposes your system's ability to transfer non-data-network material using ContentShare-compliant applications.
- If your existing system does not currently have sophisticated search capabilities, you can add them using commercially available high-class search engines.
- Device vendors' ContentShare-compliant Resource Brokers can easily add new services without affecting existing systems based on previous Resource Broker versions.
- System configuration for ContentShare-compliant devices is much simpler due to ContentShare's agent-based dynamic discovery mechanism.
- Media management applications have secure global search capabilities, asset preview, and asset transfer.

Media and Asset Management System Application Developer

- Your system can now integrate with other control systems, editing applications, storage devices and other data sources that are ContentShare-based. You can write to a single, simple API and gain access to all assets made available by ContentShare developers. ContentShare's self-describing data representation enables your software to understand metadata from third party systems without requiring inherent knowledge of the data.
- Your product can be more extensible, scalable and fault tolerant. It can support distributed asset/media/metadata information stores.
- You can connect your systems for the purposes of global asset search and transfer.
- Your product can easily incorporate existing, legacy data sources. You can focus on building next-generation systems and spend less time trying to interface to custom applications.
- Media management applications have secure global search capabilities, asset preview, and asset transfer.

Asset Storage Device or Server Vendor

- You can quickly build a Resource Broker for your product, allowing it to be used in any ContentShare system. With the broker
    - Your assets and their metadata can be located globally, using the search services of whatever ContentShare-compliant search engine you or your customers choose.
    - ContentShare-compliant applications can transfer material to and from your storage device.
    - You can easily add security to your product's Resource Broker.
- You can easily add new services to your Resource Broker without affecting existing systems based on previous versions of your Resource Broker.
- System configuration for your device is simpler due to the platform's agent-based dynamic discovery mechanism.
- The ContentShare platform enables your product's material to be easily managed by a system administrator at a central location.

Cataloging and Indexing Application Vendor

- Your product can be sold and integrated into systems you are not required to understand.
- The market for your product is increased by system-level support. Your product can compete with other catalogers based on its true value-add.
- The ContentShare platform enables your product's material to be easily managed by a system administrator at a central location.

Platform Integrator and software developers integrating complete systems

- When your customers choose new components and applications, you can quickly integrate them.
- The ContentShare platform's inherent scalability does not place limits on future growth. It actually enables system growth.
- You can integrate complex systems with little or no programming required.
- ContentShare-based systems are easier to configure and upgrade due to the platform's dynamic discovery and agent-oriented communication mechanisms. ContentShare-based systems are secure due to the platform's flexible security framework.
- ContentShare's tracing and error logging capabilities enable rapid system-level debugging.

ContentShare-based Software Development Tools

Any discussion of software development tools needs to look at several key issues. For example, what types of APIs and languages are you using and what type of software are you developing. Associated issues include

- what are the operating system/hardware platforms you are working with,
- which of these tools work with your existing software base (assuming the need to integrate ContentShare into an existing product), and
- which tools you are trained to use.

APIs – Java or COM

The ContentShare platform provides four object-oriented APIs for use by ContentShare developers. A key platform goal is to support a wide range of developers by providing User Agent APIs that are accessible as both Java and COM objects. Therefore, you can use development environments that support Java or COM to develop most of the different types of ContentShare-based software.

Both Java and COM runtimes can be freely distributed with your product.

Another goal in the ContentShare API design is to not require application developers have an intimate knowledge of the underlying agent communication infrastructure (i.e., the KQML language and the ContentShare System Agents). Of course, a general understanding is recommended and can be gained from the *ContentShare Platform Technical Overview*.

Data Exchange Language - XML

The ContentShare platform uses XML as the primary data representation language. Developers building ContentShare Applications and Resource Brokers must understand this popular, ASCII-based Internet standard language.

The ContentShare Platform uses and ships with a parser that you may freely distribute.

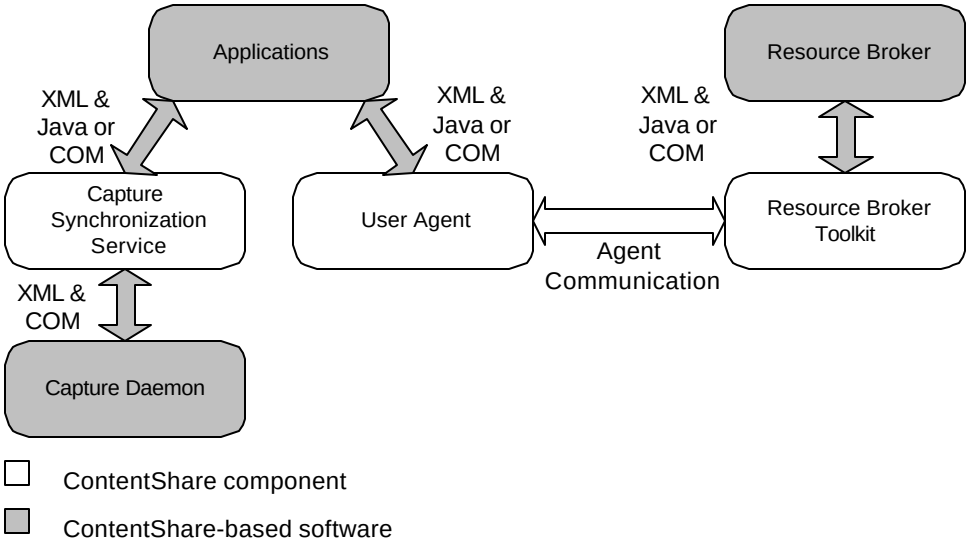


Figure 1 ContentShare-based software & APIs

ContentShare-based software types

As indicated in Figure 1, there are several different types of software that can be developed within the ContentShare platform. Most developers will only create one type. The following breakdown indicates each general type of ContentShare-based software and discusses the preferred tool, if any.

Building ContentShare-based Applications

ContentShare-based applications (applications that are based upon the User Agent API and/or the Capture Synchronization Service) may be developed in Java or COM-enabled environments. The three currently recommended environments are

- any Java environment (including web servers),
- Microsoft Visual Basic, and
- Microsoft Visual C++.

Java's "write-once, run anywhere" philosophy leads to it not being preferred for Windows User Interface development. The side effects of this 100% Pure Java philosophy are performance penalties. There is an inability to take advantage of the native operating system that gives Windows its look and feel, and inter-application communication.

To developers for whom Windows performance, look-and-feel, interoperability with Windows applications is key, and for whom cross-platform support is not required, we currently recommend Microsoft Visual C++, Microsoft Visual Basic, or Microsoft Visual J++. Visual C++ and Visual Basic both include excellent support for COM objects, and the majority of today's Windows applications are built with these tools. Microsoft's Visual J++ (especially with the introduction of the Windows Foundation Classes [WFC]) is quickly becoming viable for developing applications in Java. It also supports COM objects. Note however applications built with Visual J++'s WFC only run on Windows-based platforms and are not 100% Pure Java.

To developers for whom portability of the application is key, we currently recommend two options:

1. 100% Pure Java – these applications run anywhere there is a compliant Java Virtual Machine. Note that Sun's Swing User Interface package makes 100% Pure Java more viable for GUI development.
2. Web Applications – these HTML & Java applet-based applications run anywhere there is a Web Browser. A Web Server is required. Performance penalties, user interface restrictions, and browser incompatibilities are the negatives. However, potential worldwide access to the application is the major benefit.

Building ContentShare Resource Brokers

You may develop ContentShare Resource Brokers with any tool that can build Java applications. Java is widely accepted as a good vehicle for developing middleware. This is due to its support for threading, robustness, and garbage-collection. At the same time, Java also provides methods for accessing native APIs, allowing the best of both worlds. The use of native APIs (and thus the lack of 100% Pure Java) is not generally an issue for Resource Broker developers. This is because the underlying operating system/hardware platform is already well known and porting the device isn't generally a requirement.

We recommend the Visual J++ environment for developing Resource Brokers for devices built on the Windows platform. This is because of the simplicity of the J/Direct mechanism it provides for accessing native APIs (DLLs).

Building ContentShare Capture Daemons

ContentShare Capture Daemons are not agent-based and must provide a COM interface (no Java interface is used). The performance requirements of Capture Daemons is such that a low-overhead communication mechanism with less flexibility (COM objects with custom interfaces) is preferred over the agent-based approach used elsewhere in ContentShare.

The recommended development tool for implementing a ContentShare Capture Daemon is currently Micosoft's Visual C++. This is because of its ability to develop high-performance COM objects with custom interfaces. However, you can use any development tool that can develop COM objects with custom interfaces (including Microsoft's Visual Basic, Microsoft's Visual J++ as well as similar third party tools).

More Information

Developers interested in more information can visit the public ContentShare platform web site at http://www.contentshare.com.

The next step is to become a licensed ContentShare platform developer. This allows the SDK to be used commercially for the development of interfaces, gives access to the complete ContentShare platform and tools, and offers better development training and support opportunities. There is no charge to become a licensed developer. More information about becoming a licensed developer is available at http://www.contentshare.com.