Playout Automation in the Virtual Environment

How technology and business developments are changing the landscape of playout automation

By Ian Crockett

As consumer demand for video continues to increase, the range of delivery models expands and provides us with more ways to consume it. The challenge facing today's broadcasters and service providers is to keep pace with this explosion in demand, and to anticipate future delivery models and behavioral trends.

Whilst computer and storage clouds promise to supplement and ultimately replace today's media factories, for these implementations to be truly beneficial, the applications and solutions they deploy must be designed and built for the virtual environment.In this article we explore how these technologies apply to 'the guardian of the cash register' within the broadcast facility: the playout automation system. Playout automation touches every area of a broadcast television or video service distribution facility, and a successful solution will need to keep pace with evolutions across the board, for example in working practices, file formats and delivery platforms.

Introduction

Advances in image capture and creation, digital compression, IP bandwidth and security, accessibility and the emergence of internet as a ubiquitous social, entertainment and information utility have all driven change in the broadcast arena and will continue to do so. As a result, previously distinct functional areas have been replaced with more connected, complex and varied versions, with greater task overlap (see figure1).

But can this new model really be more flexible, scalable and adaptable, and what role, if any, can virtualization play? More importantly, how can a facility or manufacturer keep pace with this accelerated and uncertain environment?

Sponsored by Pebble Beach Systems

Figure 1. Task Overlap

Defining Virtualization

With its roots in the 1960s mainframe environment, virtualization has developed into a commercially viable, widely adopted IT toolset and as such has become pervasively deployed over the past few years. But what does the term really mean?

Hardware virtualization typically incorporates an abstraction layer between the operating system and the hardware, and aims to deliver optimized resource usage, resilience, and management in addition to reducing capital investment. Virtual machines allow a user to run multiple operating systems on a single hardware platform, and virtual LANs allow a single hardware switch to present multiple isolated networks to connected systems.

Software virtualization can be further subdivided into application server virtualization and workspace virtualization, which has more commonly been referred to as client/server architecture and is exemplified by web browsing.

Virtualizing the server application can be achieved by distributing it between multiple instances. Used together, these two techniques can bring huge benefits in resiliency, by distributing the load over multiple operational instances; in accessibility – by freeing users from geographical restrictions; in serviceability – as parts of the system can be taken down without affecting the entire system; and in scalability – since virtual server applications can be added as required.

While all of these technologies can be implemented discretely, greater benefits can be achieved by combining them together in an integrated, systemic approach: service virtualization.

In this scenario a playout and publishing application may appear to a user, or group of users, as a single cohesive operational instance. In reality, the entire system may be geographically dispersed, using applications load-distributed across a number of servers over various networks, making web service and API requests to integrate with archive subsystems, business management processes and/or real time audio and video devices.

The Caveats

However, one size does not fit all. Whilst there are advantages to be had from the systemic use of virtualization technologies, different architectures need to be applied for different reasons, and not all elements will benefit directly. For example, if the automation server needs to communicate via RS422 with legacy devices, hardware virtualization will be of limited benefit since the service is inherently bound to the hardware it is hosted on.

It is also vital to ensure that the architecture that is implemented should make operation and management of the system easier, not harder, so establishing the suitability of a software platform for a virtualized deployment is an important step.

Virtualization for Playout

So what would it take to build a virtualized implementation of a playout automation system? When considering hosting the hardware in a virtual machine, it quickly becomes apparent that any I/O which relies on point-to-point signals restricts the implementation to a specific instance of a hardware platform, creating a barrier to a VM hosting. The core elements which have been typically virtualized most effectively are storage, IP networks, CPU and memory.

Broadcast vendors have made significant progress in this direction in recent years with the increased use of network APIs and use of deferred commands to counteract the inherent lack of determinism when trying to control something frame-accurately over an IP network. There have also been considerable advances in network based time distribution which is poised to supersede LTC.

As far as software virtualization is concerned, the first and most obvious prerequisite is a client/server architecture. This should ideally be able to distribute work over multiple servers and be scalable within either a local or wide-area network, providing an abstraction of the implementation of the system from the user, which in turn provides increased scalability, serviceability and adaptability.

As systems grow it is of course vital to manage security, user tracking, and system monitoring for exceptions and real-time performance metrics. These factors also need to be abstracted and distributed appropriately.

Architecting a Virtual Playout System

So how might one architect a system with these factors in mind while leveraging both workspace and server application virtualization? When used in conjunction with hardware virtualization these become a very powerful architectural proposition.

It is useful to start with the building blocks for all the tasks which the automation system may need to fulfill. In addition to control tasks such as device drivers, there will be media management tasks, interfaces with business processes like traffic and MAM systems, and of course the core automation tasks including playlists, ingest and logging. These can be logically grouped into a container which we will refer to as a node: an executable package created by configuration (see figure 2).

A system may consist of multiple nodes, with individual components grouped within nodes based on the system architecture. These may be primary and backup nodes, nodes which host all the storage management and content management components, or nodes which group all logging and monitoring components together, depending on the goals and constraints of the system in question.

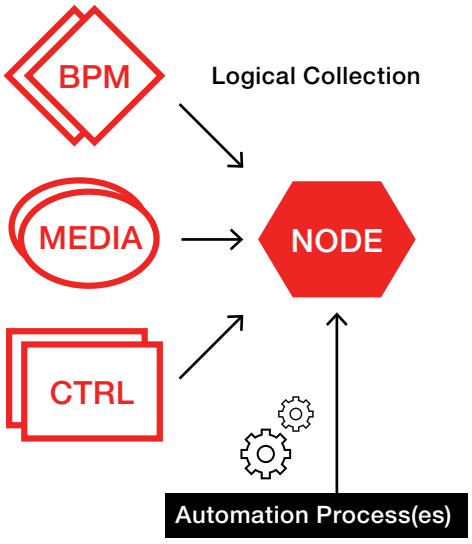


Figure 2. Virtual Software Components

This collection of nodes ultimately needs to execute on a hardware platform and, as all of the interfaces are to the network and storage, these hardware platforms can be completely virtualized.

Having created a framework of modular software components, nodes to group them in and hardware on which to execute them, let's examine what happens if those nodes are connected to a network. Firstly, the clients, or users, will see a representation of the services running on those nodes, but if the system has been designed with software virtualization in mind, it should also be possible to abstract a user from which node a particular task is being executed on. This in turn opens the door for scalability, as services can be added without any interruption to running services. Furthermore, with the appropriate resilience mechanisms in place and distributed across nodes, it is also possible to take parts of the system down and back up without interruption to running services.

The scenario described so far represents a full service virtualization of automation components on a local network, which we can term a domain.

For the sake of continued modularity, why not take it one step further and see what's possible when we connect multiple domains together? Consider a scenario where a news studio requires certain services to operate in a domain. Right next door, in the same building, is another news studio with similar requirements. But the savvy system designer made a point of removing any dependency the studios have on each other so they cannot interfere with each other, and can be upgraded independently of one another. This would give us 2 domains. But what if the system needs engineering positions to see into all the studios, and a mechanism whereby media management decisions can be made for all the studios in the facility? With the right implementation of software this virtualization can be achieved by connecting domains together.

With the appropriate resilience mechanisms in place and distributed across nodes, it is also possible to take parts of the system down and back up without interruption to running services.

While they remain self-sufficient, select data flows between them can allow a central/administrative domain to monitor status and even control playlists across all domains. Media management decisions can also be made and prioritized since the engineering domain can assess the requirements in all the studios to ensure a logical allocation of resources.

Disaster Recovery

This notion of connecting domains together is a powerful one, and has a number of use cases, for example Disaster Recovery. If the distribution mechanisms employed by the automation platform have been designed to operate in a WAN environment, why not put domains at different locations, different countries even? Under normal operations any site can see the playlists of all the other sites, allowing full optimization of human resources across the enterprise. In the case of WAN or site failure, sites would simply decouple and run independently.

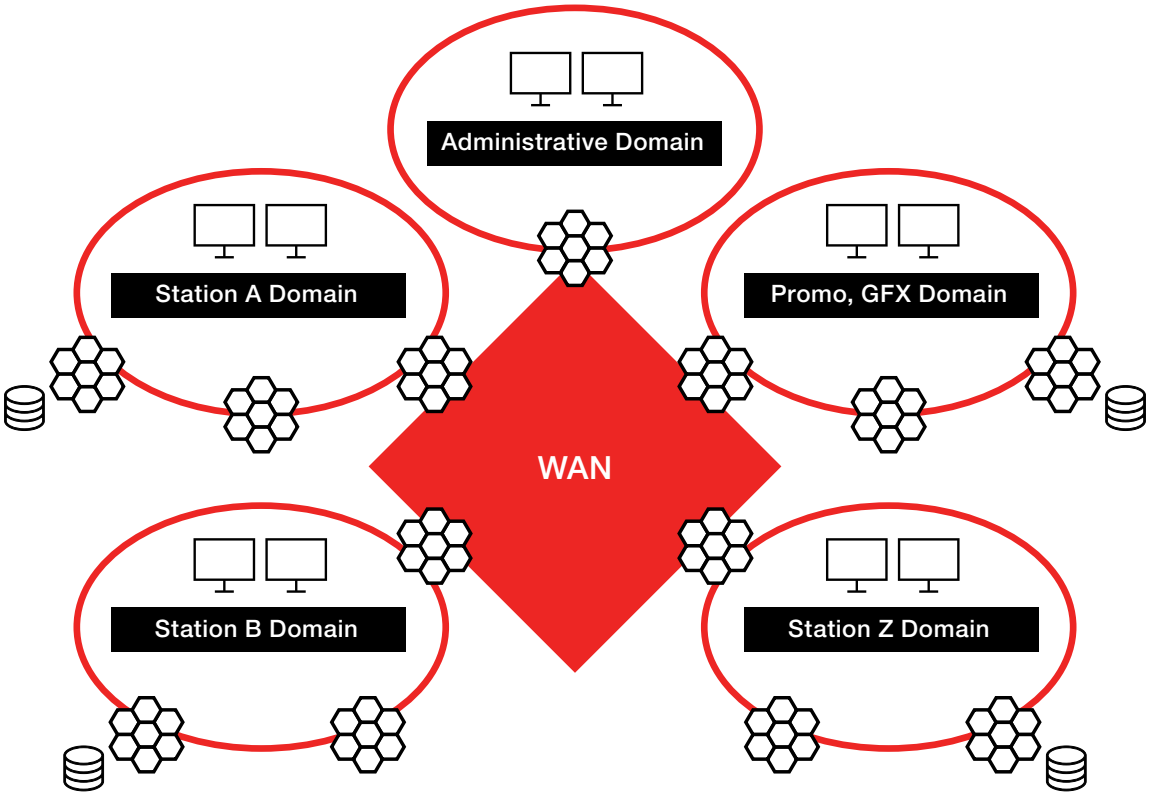


Figure 3. Station Centralization

Centralization

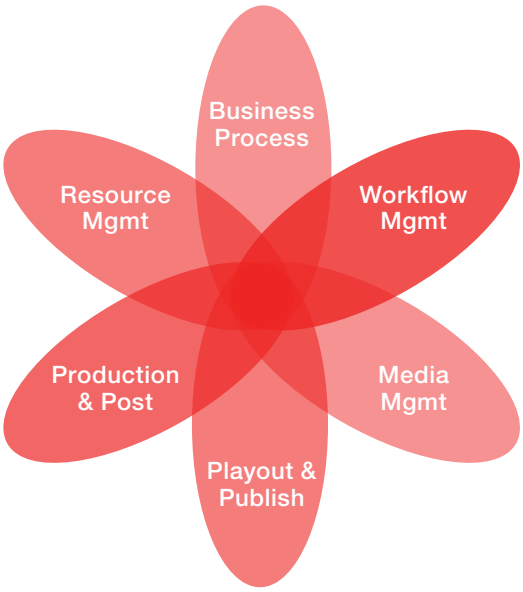
Station centralization is another common use case, and a topic which currently enjoys a high profile. The same architectural principles can be applied to abstract the implementation of channel playout from the user geographically (see figure 3). With a software virtualization implementation in place, a user should be able to see and control channels implemented in various locations side by side in the same user interface in real time. And as more and more control, monitoring and business logic is moved to a central location, skeleton-staffed or even unmanned remote playout becomes a very real proposition.

The Cloud

It is of course possible to virtualize hardware in the cloud, and the same software virtualization techniques described above apply just as well in or out of cloud environments. The real questions with regards to playout from the cloud relate more to infrastructure performance, QoS, rights management, distribution networks and so on. Many of these questions apply equally to any media service which could be hosted in the cloud, but playout automation amplifies these risks and concerns because it is so close to air and service failures are often instantly felt.

It is important to start building systems that can be virtualized (in the cloud or otherwise) and which offer the appropriate levels of abstractions to allow operations to scale and adapt to the ever-changing business challenges.

As more and more control, monitoring and business logic is moved to a central location, skeleton-staffed or even unmanned remote playout becomes a very real proposition.



The deployment of a virtualized solution can empower one operator to see across playout, workflow and production on multiple sites.

Figure 4. The Virtualized Solution

Summary

The broadcast business is without a doubt becoming increasingly complex. Technology shifts and operational realities mean that departments cross over more and more, with each needing to know what is going on across departmental lines. Overcoming these complications can only be resolved with a systemic approach. The deployment of a virtualized solution can empower one operator to see across playout, workflow and production on multiple sites (see figure 4).

While the applications of virtualization in the playout space are still evolving, this incredibly powerful architectural tool can add huge value in scalability, resilience and adaptability. It will undoubtedly reduce interdepartmental barriers and put focus on best practices and away from forced structures and fixed architectures.

Ian Cockett is Technical Director at Pebble Beach Systems

Sponsored by Pebble Beach Systems

pebble beach systems