EBU SMPTE VSF VIDEO SERVICES FORUM

# Media Architectures fit for the IT data center

*Contributed by Dr. Richard Cartwright, Principal Software Architect, Quantel Ltd.,*

*Published: 17 July, 2015*

*Note: This article is being made available as part of the activities of the Joint Task Force on Networked Media. The author is solely responsible for its content. While the information contained herein may be of interest to those following the work of the JT-NM, this document is published by its author, and is not being published by the JT-NM itself.*

One of the challenges facing the professional media industry is that we increasingly find that the technologies in our plants are IT-centric, and that the designs and concepts we use to build facilities are coming from the IT world. It is incumbent on us as an industry to be able to operate comfortably both in the traditional media technology space and in the IT technology space. To that end, the JT-NM system view provides a bridge between these worlds in the important areas of technologies, virtualization and service replication and modeling. Each of these will be discussed (and build on each other), in turn.

## Bridging Broadcast-based Technologies to IT-based Technologies

Broadcast-based technologies and IT-based technologies have been together for nearly 25 years, moving from being neighbors in the company to neighbors in the data center, to neighbors in the facility to rooming in the same house, to living together. This progression is not only continuing, it is accelerating. However, as important and as fast as this is progressing, it is worth spending the time to put the two technology bases into context.

### Data center technology

What happens beyond your device when you open a web page from an Internet browser, connecting to a remote server in a data center somewhere in the world? Why is it that, given a good connection path from your browser to a service provider, an individual response can be returned quickly, even when thousands or millions of other people are using the same service at the same time? How can a service provider scale up and scale out their technology to meet demand on a global scale? The answers to these questions are the results of 40+ years of research and billions of dollars of investment in commodity data center technologies by the IT industry. When building new infrastructure and platforms, IT-industry innovation is available for exploitation by the media industry. This exploitation can only take place by aligning with IT architectures.

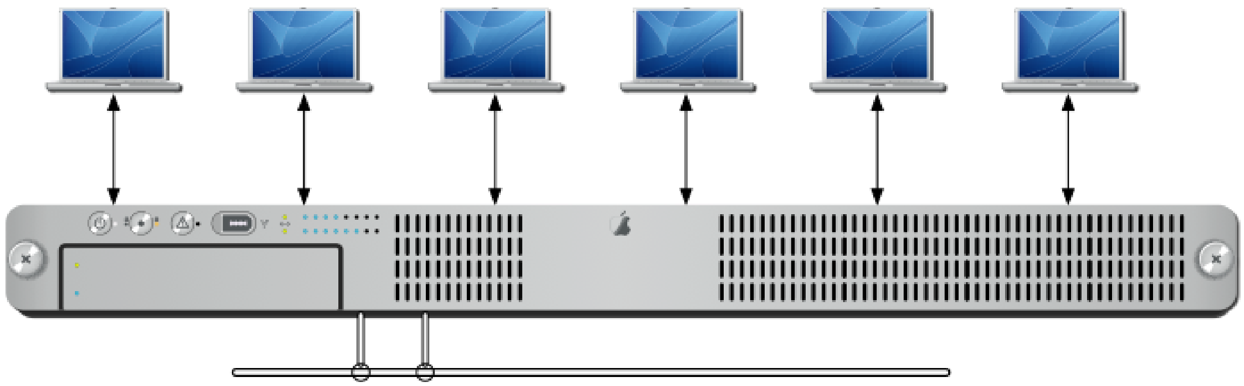

*Figure 1: Multiple Clients Accessing a Single Server.*

In the simplest case, shown in Figure 1 above, many consumers of the same service connect to a single server located in a data center somewhere in the world. The Internet's global Domain Name Service (DNS) is used to translate the name for the service into an address, and the Internet's routing systems route packets between the server and client.

The server contains one or more CPUs and/or GPUs, each of which may have one or more processing cores and some local hard disc. The processors and their processing cores share random access memory (RAM) and one or more network connections, with system components scaled and/or duplicated for resilience and performance. In turn, many identical servers are installed in a rack or blade enclosure, with many racks interconnected by top-of-rack non-blocking network switches.

So called *single-threaded* applications can only run on one of the CPU cores and so cannot exploit all the processing resources of the device. If an application is written as *single-threaded*, this would mean that the server has to finish serving *one user* before serving the next, limiting scale even if multiple instances of the same software were run. To scale to several synchronous users, applications are written to be *multi-threaded* and exploit facilities provided by the operating system to time-slice the application's execution for all users across all processors. This means that the server slows downs equally in its response for each user until it reaches capacity and runs out of resource.  While it does slow down for the individual user, the total amount of work completed for all users is increased so the overall *throughput* is higher.

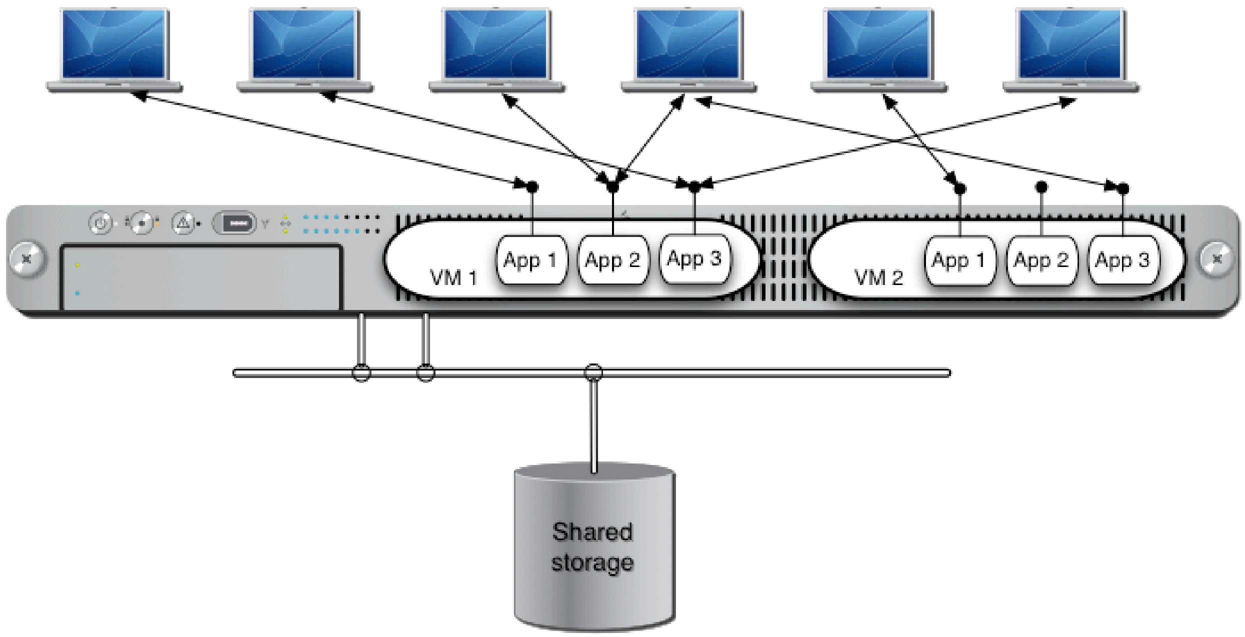


*Figure 2: Multiple Clients Connecting to Applications in Virtual Machines*

The overall resource of a multi-processor, multi-core server is substantial and it is often desirable to control the partitioning of that resource across more than one application. A common way to do this is to deploy one or more *virtual machines* or *containers* onto the same server, with a virtual machine or container used for one or more related applications. This is illustrated in Figure 2.

Resources can be allocated per virtual machine or container and shared out between them. A key benefit of this approach is that the whole virtual machine or container can be managed as a unit, with consistent versions of software and operating system managed together. Rather than installing software, virtual machine images or containers are deployed, updated, scaled and taken down as a unit. (Virtualization and Service Replication are discussed in the next section.)

Each user gets a consistent user experience because they provide credentials with a sequence of requests. The current state of their interaction with the server is kept in shared storage. With careful management to avoid conflicts between two paths, two subsequent interactions between the client and the server may use different instances of the same service that are running in different virtual machine instances. This is referred to as a *stateless application*, where no long-running interaction state is stored directly within the memory of the application itself.

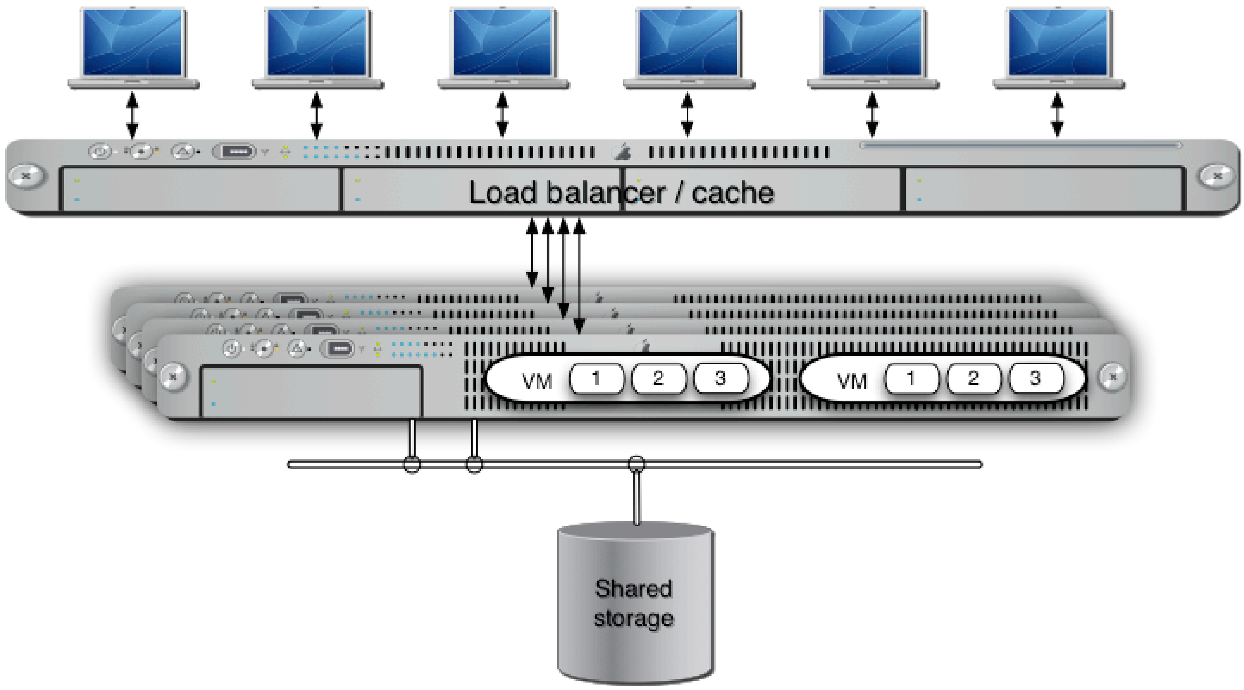


*Figure 3: Using Load Balancers & Caching Devices to Transparently Scale Applications*

When an application scales beyond the resource of a single server, as shown in Figure 3, the same virtual machine image or container can be deployed to either a different server with greater resources or to multiple servers. When it is moved to multiple servers, the service is then advertised through DNS as the address of a load balancer. The load balancer distributes requests to the service amongst a number of virtual machines running on a number of servers.

The load balancer can detect the failure of any single virtual machine or server, fail fast to the client with a suitable error and make sure no further requests are routed there. Client applications are written to be fault tolerant, automatically re-requesting to the server in the event of a failure. With the addition of commodity caching, resilient configuration of the shared storage and appropriate software architectures, single data centers can scale flexibly and remain highly resilient.

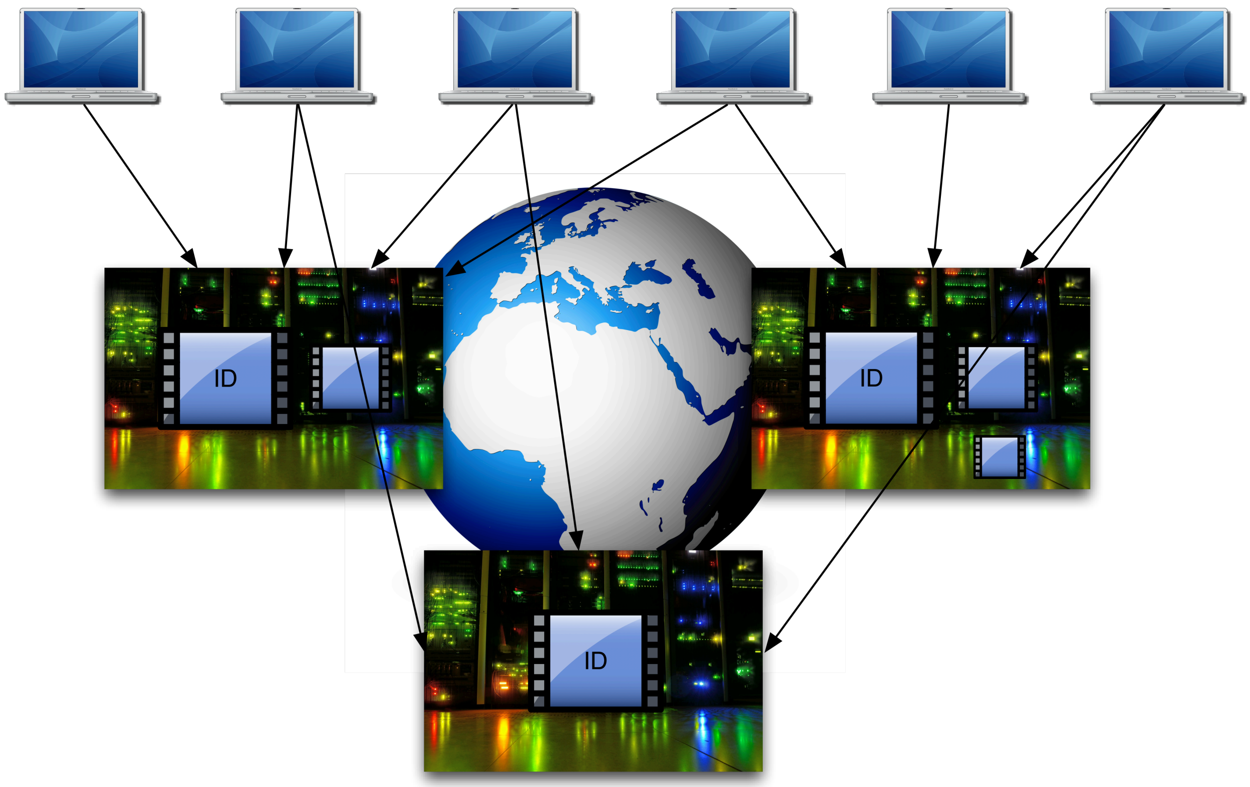


*Figure 4: Scaling Beyond a Single Data Center with Content Distribution Networks (CDNs)*

Scaling a single data center does not solve the problem that the speed of light is too slow! To provide scalability and retain globally acceptable response time for a service, it is necessary to run that service geographically close enough to the user. It may also be necessary to protect against the temporary failure or catastrophic loss of any one data center. To do that, as shown in Figure 4, identical services can be deployed to more than one data center and the shared data they used copied between the centers using *content distribution networks* (CDNs). DNS resolves identifiers for services to an optimized server based on location, availability, cost, and other metrics.

The physical architecture of commodity, network-connected data center technology is defined outside the scope of the media industry. The differentiating artifacts that are deployed to this generic technology to make it industry-specific are software applications. Although it is expected that legacy media infrastructure will need to coexist with IT infrastructures for some time to come, a Reference Architecture for Media needs to include a model that embraces current concepts and is a good fit for the data center of the future.

### Modern software development practice

Imagine being able to upgrade servers in a media facility at the push of a button? How about scaling infrastructure up for an event or during an event and releasing the resource afterwards? Hand-in-hand with the evolution of data center architectures, flexible, software-based means of managing network, infrastructure and platforms in non-industry specific ways have also been developed.

The *continuous delivery* approach to deploying software builds pipelines that update Internet services with new features up to several times a day, using automated processes that take a few minutes to run. The key to these techniques is:

- Version control of everything, where all aspects of software, operating system and hardware configuration are incrementally tracked as they changed, allowing them to be backed out if necessary;
- Automated building and testing of complete environments (application, OS, shared storage) where as many aspects of an item of software's function and performance are tested as possible;
- Automated deployment of software – often as complete virtual machine packages or containers – that can be quickly backed out to previous versions.

Lights-out management of servers in data centers is commonplace, whereby the physical control and monitoring of hardware is done remotely. Once a server is installed and commissioned in a rack, it may never be physically touched again until it is decommissioned and removed. With the continuous delivery approach to software development, where any single change a developer makes may make it into a release build, a similar lights-out management style can be used for software too. As such, the skillset required to operate a data center – a role that is known as *dev ops* (i.e., “development operations”) due to the need to develop and manage automated pipelines from an operational point of view – is different from the hands-on skill required to set up and manage a typical media facility.

## Starting Point for Discovering and Provisioning Capabilities

In such dynamically provisioned and potentially remote environments, or a hybrid environment consisting of current media facilities and IT infrastructure, it is necessary to be able to discover and provision capabilities. The starting point for this is a model for what that capability is and how to use it. To build such a model requires a means to describe it. In a separate paper[1], a tutorial on one methodology to build such a model using UML – the Unified Modeling Language is discussed and applied to the professional networked media environment and the JT-NM requirements and user stories.

[1] “Modeling with the Unified Markup Language (UML) – A Tutorial,” Contributed by Dr. Richard Cartwright, [place URL here], March xx, 2015.