Telestream
Whitepaper

Infrastructure Design for High-Volume Video Workflows

15 Key Considerations for Large-scale File-based Processing

“high-volume environments, where effective media management is absolutely critical to success, depend not only on intelligent workflow design, but also on an infrastructure that is optimized to execute the workflow efficiently and reliably on a very large scale.”

Contents

Introduction

Less than two decades ago, digital video was in its infancy, with then-new technologies such as MPEG video, ATSC television, and QuickTime – not to mention the World Wide Web – all pointing toward a bright future for digital media. Today, while further exciting applications for digital video are no doubt still to come, the promise of those early days is largely realized. The acquisition, processing, and distribution of video content have transitioned to mainly file-based processes, resulting in unprecedented flexibility and reach for content owners, creators, and distributors. As new digitally-enabled capabilities have proliferated, however, so too have the challenges of making them practical and efficient for real-world use.

telestream

Nowhere are these practical challenges more acute than at major media companies, which now generate, process, store, and distribute vast numbers of digital video files. Such high-volume environments, where effective media management is absolutely critical to success, depend not only on intelligent workflow design, but also on an infrastructure that is optimized to execute the workflow efficiently and reliably on a very large scale. This paper outlines the essential elements required of such an infrastructure.

While the specifics of a high-volume workflow may vary in each situation, the key infrastructure considerations discussed below apply to any organization requiring high throughput in video-related operations such as:

- **Content repurposing** for multiple distribution channels that each have different format requirements (e.g. multiscreen to web, mobile, broadcast, cable operators and content delivery networks).
- **File transfers** from any given device to one or more different devices (e.g. transfer of news or sports footage from the acquisition medium to post-production storage).
- **Production** of commercials and syndicated programming (e.g. content preparation for broadcast and cable distribution.)
- **File conversion** for multiple departments that each focus on a different segment of the media market (e.g. motion picture, television, and gaming operations within a single enterprise).

In general, the turnaround time allowed for these types of video processing operations is decreasing, while the number of distinct deliverables is increasing as playback platforms proliferate. To see the pressure this puts on video processing infrastructures we need only consider the example of a media company that covers major sporting events. Footage captured by multiple cameras at each game must be accessible to multiple editors who each create different versions of the coverage. Multiple games may occur simultaneously, each requiring production for both live viewing and immediate post-game streaming. Each instance of the game coverage will be watched at many different times in many different ways – broadcast/cable television, Web-connected personal computers, mobile devices – each requiring optimization for different codecs, resolution, aspect ratios, and bandwidths.

The bottom line is that every increase in the flexibility and choice offered to content consumers involves a corresponding increase in the volume and complexity of content production and distribution.

When tens of thousands of files per week are involved, there's no time for bottlenecks and inefficiencies. The infrastructure supporting the process must be designed from the ground up to handle this enormous task.

The top design priorities for a high-volume file-based processing infrastructure are process automation, resource optimization, and system resilience. Implementing these characteristics requires the right mix of both software and hardware ingredients:

- **Software** – As volume increases, the key to handling the workload is to manage workflow and resources intelligently. Labor-intensive approaches that may be appropriate to an organization processing a dozen media files each week will be quickly overwhelmed when demand increases to hundreds or thousands of jobs each week. Throughput and consistency can both be enhanced by replacing "hands-on" intervention with intelligent software-controlled processes.
- **Hardware** – While software infrastructure facilitates the efficient execution of steps in a workflow, throughput depends as well on the individual performance of storage, networking, and processing components, as well as the overall design of the hardware system in which those parts work together as a whole. For high-volume operations, this hardware infrastructure must scale while delivering accurate and predictable results.

In the following sections we'll look at the key factors to consider in designing an efficient, highly productive video-file processing and management infrastructure for high-volume environments.

Efficiency through automation

People are good at thinking through what needs to be done and how best to do it. But the actual execution of most tasks is not only faster but more consistent when handled by machines under the direction of process automation software. The higher the volume of work, the more valuable such software can be in speeding throughput, assuming that it is designed with an intimate knowledge of the processes that are to be automated. In some cases a given task might be entirely automated, while in others automation might be used to create the conditions under which skilled operators can function most effectively.

"Processing speed and transfer rates are obviously important, but overall system performance depends as well on whether a system is configured and managed to avoid bottlenecks and utilize each resource most effectively."

Automating process steps – Many of the steps in a typical video workflow can be handled entirely via software-driven automation, among them:

- automated file discovery;
- asset encoding and/or transcoding;
- metadata transformation;
- analysis;
- decision-making;
- delivery.

Tying these and similar steps together into a process automation model can drastically reduce the number of points of human intervention required in the overall workflow, which in turn can speed throughput while removing the element of human error.

Streamlining non-automated steps – While many video workflow steps are ideal candidates for full automation, other steps, such as editing, require the "hands-on" expertise and creativity of human operators. Even these steps, however, can be made more efficient and productive by using software solutions to eliminate unnecessary intervention points, to handle background tasks that set the stage for the remaining interventions, and to trap operator errors. Effective asset-search tools, for example, can drastically shorten the time operators spend evaluating and choosing the files needed for a given project. Ideally, workflow automation software will also enable an administrator to customize operator facing interfaces so that they present operators with only the information and controls necessary for the task at hand. Interface design options should also be flexible enough to allow the process to evolve and improve over time.

Optimizing performance

Processing speed and transfer rates are obviously important, but overall system performance depends as well on whether a system is configured and managed to avoid bottlenecks and utilize each resource most effectively.

Balancing CPU utilization – With multi-core machines now the norm, and multiple machines often harnessed into distributed processing systems, the optimization of resource utilization depends on load balancing within and between CPUs. Traditional queue-based load balancing, however, often wastes over 60 percent of CPU capacity by disregarding the relative "cost" of the jobs in the queue. A system with two servers set to process four jobs each, for example, could easily result in one processor handling four low-cost 3GP jobs that use only 20 percent of CPU and the other processor handling four high-cost MPEG-2 HD jobs that demand 300 percent of CPU. More advanced load balancing algorithms poll instantaneous CPU levels to determine task allocation, but they can be easily fooled because CPU utilization varies drastically from second to second, and because any scheme that is not cost-aware may inadvertently under- or over-utilize the server.

For large-scale systems, a far more effective approach to spreading the load is to consider both the processing cost of individual tasks and the capacity of the available systems. This cost-based load balancing will ideally be implemented to allow administrators to fine tune settings to squeeze as much performance as possible out of the overall system. For mixed content encoding, throughput gains of 30 to 50 percent over "Round Robin" scheduling may be achieved in normal operations, with a two to five times improvement possible when large quantities of low-cost or high-cost content arrives for processing at the same time.

Calculating network throughput – As servers become faster and faster, data transfer either keeps up or becomes a bottleneck. That makes it vital to correctly understand the total required data bandwidth both to and from storage. When transcoding at four times real-time, for example, a single transcode of a 145 Mbps DNxHD file requires bandwidth of 600Mbps just to read the data. For a reliable high-volume workflow, the hardware infrastructure must be able to handle the combined bandwidth requirements of all files that might be utilized at times of peak demand.

Using heterogeneous storage – When the seemingly insatiable demand for storage capacity runs up against budget realities, a mix of storage device types may be the key to maintaining performance without busting the bank. Shared storage, such as a high-speed fiber-attached SAN, makes all files involved in a given job available over the network and allows each task to be independently load balanced.

Distributing storage across multiple devices, on the other hand, can help save on costs and reliability, though it may also introduce unnecessary transfers between storage systems or across networks. It may make sense to deploy varied storage devices with different capabilities, such as fast local storage for high-bandwidth transcoding work and slower but network-accessible storage for low-bandwidth proxies.

Integrating specialized hardware – Certain tasks can benefit greatly from the application of specialized hardware to maximize performance. Machines may be equipped with GPU- or DSP-based accelerator cards for video transcoding, for example, or with a fiber connection to a storage device that provides the system's only access to that storage. It's essential to take these specialized hardware components into consideration from the outset when designing the overall system, so that the architecture supports efficient routing to and from these devices.

Allocating resources

From the workflow perspective, a system is a set of resources that are available to handle a set of tasks. Because different jobs require those resources in greater or lesser degree, an efficient infrastructure allows resources to be dynamically allocated as needed based on the mix of tasks being performed at a given moment.

Optimizing the load – An intelligent process management system knows the capacity of each machine, how much of that capacity is or will be tied up on currently executing jobs, and what the capacity requirements are of the upcoming jobs in the queue. Based on that information, it should load balance between servers and dynamically allocate resources on the fly, passing work to under-utilized machines and avoiding machines whose capacity isn't sufficient to handle a given job. Machines with lesser CPU power, for example, may be assigned tasks such as simple file transfers, while more powerful machines may be reserved for high-performance tasks like transcoding.

Enabling job prioritization – All jobs are important, but some are more urgent than others. System control software must be flexible enough to fast-track specified jobs through a workflow – even pausing low-priority jobs if necessary to free up resources – without throwing off execution of standard-priority work. Consider, for example, a system used to handle multiple processes, one involving thousands of jobs that are needed in a week's time and another involving a number of jobs that must be ready in five minutes.

It's clearly impractical and inefficient to manually stop thousands of lower-priority jobs, only to restart them again after handling the urgent items. Instead, a strong software solution will allow preemptive processing of files so that both needs can be addressed.

System hardware may also be specifically configured to facilitate simultaneous processing at multiple priority levels. It may be desirable, for example, to dedicate some machines for highpriority workflows with fast turnaround, keeping them more or less independent of a more general pool of transcoders.

Supporting plug-and-play expansion – When a system is running 24/7 to meet production deadlines, it's counterproductive to have to stop all processes in order to add capacity. The hardware infrastructure must be designed to allow the addition of processing or storage hardware without interrupting work-in-progress.

Interface and Reporting

The power of a processing and management system derives in large measure from the extent to which operators can get what they need from the system with minimal investment of time and effort. That requires a user interface that is easily understood and fast to set up for the task at hand. The interface must also collect and present the ongoing process information needed to either assure operators that the desired results are being achieved or alert them to issues that should be addressed.

Integrating process/workflow analytics – Identifying points of inefficiency in a process is clearly a key step in making that process more efficient. The payoff can be dramatic: shaving 15 minutes off of each iteration of a process that is applied to 10,000 files adds up to over a hundred days of computing power saved. But without effective tools, process analysis can itself be a time-consuming task. Software should minimize the burden of this task on operators and administrators by tracking media workflows, analyzing task execution times, and using visual feedback to highlight the location of bottlenecks.

In one real-world example of how bottleneck analysis can help, a broadcaster was processing HD files by first performing automated analysis to locate a good (i.e. not black) thumbnail, then generating the thumbnail, and then transcoding to generate a file proxy. Process analysis revealed that this approach involved accessing each file three separate times, once for each of these tasks. By changing the workflow to create the proxy first and then use that proxy for analysis and thumbnail generation, the broadcaster reduced overall processing time by more than 30 percent.

Tracking and reporting on jobs – The progress of each job through the system tells a story about the performance of the system itself. By tracking jobs and the resources utilized to complete them, reports generated by well-designed control software can provide valuable information about system strengths and weaknesses, which in turn can shape system expansion or re-configuration to better keep up with demand. Knowing how long different types of jobs take also allows you to better calculate the true cost of moving jobs through the system and to extrapolate how many machines you will need to process a given number of files in a given time frame. The information should also be exportable, allowing analysis in alternative environments.

Supporting dashboarding – A list of a dozen jobs can be informative, but a list of a thousand jobs is simply overwhelming. In large-scale processing situations, seeing the status of an individual job may be less important than quickly grasping the overall performance of the entire process. To fully support large-scale media processing, software solutions should enable high-level "dashboard" views with job-status reporting that is aggregated into an overall metric, either for the entire system or for a user-specified batch of files.

Dashboarding is also important for system hardware views. When your system involves more than a few servers your control software should be able to analyze them for you automatically and highlight health metrics such as failure rates per machine or per system. It should also be able to isolate root cause and failure in large systems with multiple elements. This allows you to be proactive about identifying potential system failures and addressing issues such as a choking server or a bad connection to the network.

Supporting hardware profiling – The ability to track the capacity of individual pieces of hardware within the system is important because it allows you to identify infrastructure bottlenecks. For example, the fact that a given storage device is the slow point in a workflow may not be obvious until observing the behavior of the underlying hardware and the workflow performance at each step.

Resilience and reliability

Even the simplest video workflows typically rely on a system that comprises multiple components, including CPUs, networks, storage, and servers. Because no system is immune from failure, the best-performing systems over the long haul are those that are designed to anticipate failure rather than to ignore it.

Built-in self-recovery protocols can help avert catastrophic failure while maintaining confidence in the output of file-based processes, ensuring that results meet or exceed the same quality standards as signal-based workflows.

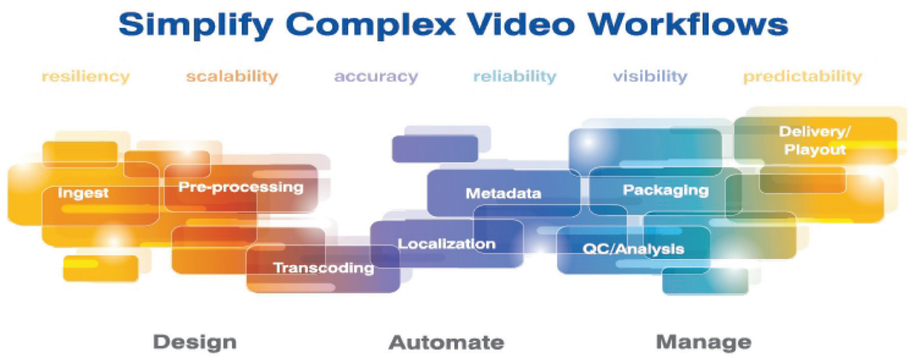
Building in resilience – In all but the smallest of operations it's simply not a viable option to manually retry every task when an overall process fails to complete successfully. Instead, large-scale systems should be designed with extensive process resilience, which means offering two forms of protection against failure:

- **Failover** offers resilience in case of a software component failure, automatically switching to a redundant component upon failure of the primary component handling a given task.
- **Automated retries** offer resilience in case of an external system failure. Automated retries should allow the administrator to control recovery steps with retry schedules that are customizable based on the type of external failure and the type of task that was disrupted. If the task is a file-based transcode, for example, immediate restart of the transcoding might be appropriate when the cause is a momentary local storage hiccup, whereas waiting a few minutes to restart would be preferred when the cause is related to the transfer of a file over the Internet.

Providing process predictability – Software that supports the various capabilities described in the sections above ultimately makes a system more reliable by ensuring that system capacity is appropriate to demand, that failures won't bring production to a standstill, that system resources are optimally load-balanced, and that high-priority work will be allocated the resources needed for timely completion. With that reliability comes predictability, which is the reasonable certainty that each and every job will result in the intended outcome.

Conclusion

In today's video-intensive media environment, effective workflows for file-based video processing and distribution are an essential part of any viable large-scale media enterprise. The hardware/software infrastructure that supports such workflows must allocate resources effectively, accommodate multiple levels of job priorities, provide transparency for tracking and troubleshooting, and maintain scalability as demand grows. It must also provide the exceptional reliability and ease-of-use required to perform effectively in real-world, high-pressure environments. Careful consideration of the factors outlined above can help ensure that both the hardware on which the system runs and the software used to manage it are appropriate to the scale of the task.



Copyright © 2014. Telestream, CaptionMaker, Episode, Flip4Mac, FlipFactory, Flip Player, Lightspeed, ScreenFlow, Vantage, Wirecast, GraphicsFactory, MetaFlip, MotionResolve, and Split-and-Stitch are registered trademarks and Pipeline, MacCaption, e-Captioning, and Switch are trademarks of Telestream, Inc. All other trademarks are the property of their respective owners. May 2014