The Evolution of the Interactive Broadband Server

Part 2—The Cool Hairy Onion

Joan Van Tassel, Ph.D. and Steve W. Rose

Dr. Joan Van Tassel is the author of *Advanced Television Systems: Brave New TV*, published by Focal Press (1996). She writes for *The Hollywood Reporter*, *WiReD*, and other publications on the emerging communications infrastructure. Prior to entering graduate school, she was a documentary filmmaker, known for her work on social issues. Dr. Van Tassel has written for a variety of nonprofit public information campaigns, including the "Friends Don't Let Friends Drive Drunk" campaign. She is the recipient of the prestigious Kenny Rogers Media Award, an Emmy nomination, and many public service advertising awards. Dr. Van Tassel received her Ph.D. and M.A. from the Annenberg School of Communications at the University of Southern California.

Mr. Steve W. Rose is an interactive broadband server consultant with Pangrac & Associates. He started his career in the 1970s in video, became the chief engineer of a small Maui cable company, and ended the 1970s designing cable television automation equipment for CRC Electronics. CRC provided the tape automation equipment for the first Warner Qube system. CRC was later purchased by Texscan, so during the 1980s, Steve was a computer guy, working with multiuser and networked computers. In 1991, ATC (later Time Warner Cable) asked him to figure out how to provide video on demand for a system with a million subscribers, which launched him on a new career. In 1994, he worked with CableLabs on their Media Server RFI, visiting various server manufacturers and giving a tutorial at the resulting CableLabs meeting.

This article is the second of two parts. Our purpose is to clarify the assumptions made in designing, building, and implementing interactive broadband servers. It is our hope that these articles will advance the discussion and accelerate the development of these critical devices. For those readers who may not have seen the first article or who saw it and want to refresh their memories of the ideas we presented, we begin by summarizing our initial paper.

We considered the design of the conventional interactive broadband server (IBS) and specified the set of tasks it must carry out, regardless of its design:

- *Importation. Find the correct data and import it from storage into the stream generator.*
- *Generation. Reassemble content into a stream.*
- *Encryption. In many cases, encrypt the stream.*
- *Sort, switch and multiplex. Route stream to the correct subscriber.*
- *Encode stream for transport. Monitor quality of service, forward error correction, and modulation.*
- *System management. Operations (OSS) and business support (BSS) systems.*

Metropolitan-sized interactive broadband servers that can accomplish these tasks are deployed in many of the interactive television test sites around the United States. The architecture of most of these servers has evolved in an essentially linear manner, and each function has been addressed by adding a new layer of hardware and software.

The conventional design begins logically enough with storage and stream generation. Then, the switch is added for sorting, routing, and multiplexing. Ap-

pended downstream modulators encode the output for transport. The need for end-to-end management calls for the overlay of an operational support system. Finally, business requirements demand a third system to monitor usage, store the data, and charge customers for service. This entire assemblage of equipment is purchased from an army of vendors and glued together by elaborate hardware and software interfaces.

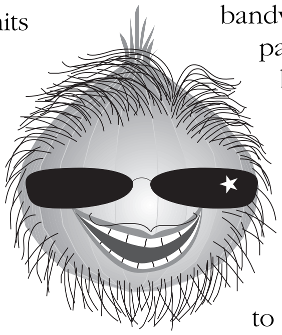
The first article ended with an identification of the problems of conventional IBS design:

- *Complexity.*
- *Vulnerability to single point failure.*
- *Lack of scalability.*
- *ATM switch limitations.*
- *Demands on headend facilities.*
- *Excessive power consumption and associated costs.*

In Part 2, we turn to an alternative, integrated design for the IBS that we believe will obviate these problems and will result in improved performance and reliability. The rest of this article will cover the rationale for the features and resulting benefits of an integrated approach to IBS design. Although this integrated server described does not yet exist, many of the pieces exist in isolation.

Why a Cool, Hairy Onion?

In the 1970s, discussions about the most efficient configuration for the structure of future computer networks centered around a metaphor close to our title: the hairy smoking golf ball. The image derived from the idea that, in order to minimize delay, all processors must be as close as possible to each other—hence the image of a sphere. The limits imposed by the speed of light meant the ball had to be small, like a golf ball. As the size was reduced, it was recognized that the power density would increase within this confined space, heating up enough to smoke. Finally, as all the communications lines were squeezed together by the shrinking surface, the ball would begin to appear "hairy."



We are at a similar point with respect to interactive broadband platforms as these early designers were when they considered future computers. However, the complexity of the technology has increased so that the simple golf ball processor of yesterday's computers has become today's onion, with richly connected layers discriminated by function and responsibility.

This paper covers the design of the media server because it is the brain of viable, efficient, and cost-effective, large-scale, two-way networks. As Al Kovalick, the media server architect at Hewlett-Packard, noted, "People ask 'why work on video servers?' The answer is that if I solve that problem, I've solved everything."

"Everything" includes large-scale, two-way broadband networks of all kinds, whether they are constructed and operated by a cable, telephone, or computer company. The content provided may be interactive television programming, video on demand, interactive shopping, or—now a subject of significant current discussion—video-enabled Internet and World Wide Web traffic. Indeed, the quantity of information that must be stored, reconstituted, sorted, and delivered overwhelms any other current application, by about three orders of magnitude.

The relatively small populations served by Internet Service Providers and the geographical dispersion of system hardware has obscured the fact that the design of broadband networks is application independent. The rapid delivery of large amounts of data, especially time-sensitive material, to an enormous number of participants demands the same equipment, including the IBS, regardless of the type or purpose of the communicated content.

As more people begin to use the Internet, the same issues that arose in discussions of interactive television systems will come to the fore. Such aspects include the design of storage, servers, switching, security and encryption, error detection and correction, bandwidth management, and business support. In particular, the issue of bandwidth distribution becomes critical.

Understanding the Problem

We have a remarkably short attention span. In 1993 and 1994, the rallying cry was "Interactive Video on Demand." In 1995, VOD became passé, and it became popular to dismiss the whole concept. This new attitude is coincident with the failure of many telco and cable tests to deliver promised services, with the notable exception of Time Warner Cable's Full Service Network in Orlando.

The new bandwagon has become “video over the Internet.” What people have not seemed to recognize is that interactive television and video over the Net are the same thing. For example, take quality of service. There is no difference in the compression schemes used to deliver digital video via the Internet, ADSL, or cable systems. A given level of quality requires the same data rate regardless of the means of delivery. It is almost silly to state this, since the means of Internet delivery is generally via telephone or cable. People will tolerate a low data rate, poor quality presentation via their computer for as long as it remains a novelty or is provided free.

Planning for the level of demand also remains the same. For interactive digital video, cable providers have assumed that peak demand would fall between 7% and 40% of subscribers, with 20% commonly used as the design point. If this is a reasonable estimate for demand for these services, then it makes little difference how the services are delivered: Total peak bandwidth demand equals the number of subscibers times the bandwidth per subscriber. As we pointed out in Part 1, in a typical cable headend area serving 50,000 subscribers, at 20% peak demand allowing four Mb/s (megabits per second) per stream, the total headend output bandwidth is 40 Gb/s.

MCI announced (with pride) in March 1996 that it is upgrading its Internet backbone from 45 Mb/s (DS-3 rate) to 155 Mb/s (OC-3 rate) to keep up with demand for bandwidth. This major step forward at relieving Internet congestion is their backbone bandwidth for the entire Internet. It does not compare well with the requirement for 40 Gb/s for a community of 50,000 subscribers!

Note that the incremental cost of bandwidth from a headend or central office to its area of service is basically zero, whereas backbone bandwidth is scarce and expensive. When the nature of the video traffic to subscribers is generally the same content repeated at slightly different points in time, it is inefficient and costly to use backbone bandwidth for this purpose.

What is required is a local server, which receives one copy of the content by the least expensive means available, stores it, and reconstitutes it on demand for any subscriber. It doesn’t matter if you are talking about a “regional server” concept for cable, or “video enabled Internet services”—*material must be cached as close to the consumer as possible to make the system practical economically and technically.* The obvious exception for which backbone bandwidth must be consumed is real-time data of any sort. However, real-time data requires only a single stream per event. If the event is to also be made available on a delayed basis, then it too needs to be cached locally. We will return to this issue of the location of stored material and the server that retrieves it at the end of this article.

An Overview of the Integrated IBS—The Cool Hairy Onion

Any large-scale media server is inherently complex because it must address the full range of tasks listed above (e.g., importing, generating, switching, etc.). However, note that each of these functions involves relatively simple information processing, in effect, highly-effective bit shuffling. The integrated design differs from the linear design by placing almost all of the various processing in a structurally central location. While both designs incorporate layers of processing, in the integrated approach, these layers are functional and conceptual rather than physical. In this section, we briefly describe each functional layer. In later sections, we will cover them in greater detail.

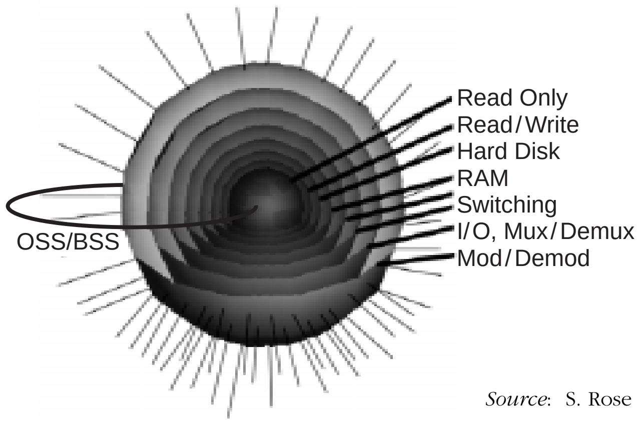
At the innermost core of the onion is the system’s content, stored on several different types of media. There are three levels of storage:

- Optical disc-based permanent, read-only.
- Optical or magnetic media, recordable and rewritable.
- RAM-based recordable and rewritable.

At least two robotic mechanisms will have access to the full library, so that no one mechanism can malfunction to produce a single point failure.

The library can store any type of material, including text, data, graphics, compressed audio, animation, and video. It will include pre-produced, read-only material, such as movies. Each title is stored individually to enable random access (access to a second title on the same medium would be blocked during a read of the first title). It will also incorporate read/write media which might be on tape or recordable optical disc. Incoming data streams intended for later regeneration, such as live events, Internet downloads, news groups, or other content not available on high-capacity Digital Versatile Disc (DVD), are routed and written directly to this part of the library. These tapes or discs can be served by the same robotic storage system that accesses the read-only media.

Figure 1
The Integrated Approach to IBS Design



Source: S. Rose

The Core—Storage

The hard disk storage array acts as a cache for the primary storage of the library. In a medium-to-large server, this will involve hundreds to thousands of disk drives, arranged in redundancy protected subarrays (with RAID 5 and a five-drive array, a 25% redundancy factor ensures that no single drive failure will affect data reproduced from that array). With an appropriate architecture, the data stream coming from primary storage for a given title is striped immediately across all drive arrays, allowing worst case access time to the material within seconds.

The RAM cache is not a separate storage area. Rather, it is implemented in each of the many stream generating processors—about 32 megabytes of RAM per processor. One second of compressed video data can be stored in about half a megabyte, so each processor has about a 60-second RAM cache associated with it.

From the imported content, the server generates a coherent stream. In the integrated server, this is done with an array of multiple interconnected processors. They have sufficient processing power to allow them to reassemble the individual chunks of content into a contiguous isochronous data stream, addressing and routing them to the correct output for the subscribers requesting them. Each stream can be reassembled at speeds at least as fast as standard rates for video data.

Layer 1—The Switch

The switching layer comes next. However, there is no switch; the switching function is carried out by the same processors that generate the streams, addressing and routing them to appropriate destinations. For example, a subscriber's telephone call might be routed bidirectionally via SONET interconnect to a local exchange or interexchange carrier.

Layer 2—Encryption

The processors in the layer where coherent streams are reassembled are lightly loaded in managing the switching matrix. Thus, these same processors can again be used to encrypt each stream, which must be done in real-time on a stream-by-stream basis.

Layer 3—Multiplexing

The interconnected processors have sufficient capacity to perform additional tasks, including multiplexing and demultiplexing. The multiplex layer combines the individual data streams destined for one output. The streams are combined in a manner which maintains their isochronicity and labels them in a way that allows them to be identified individually by equipment at the subscriber's premises. Given a data rate of four Mb/s, an OC-3 rate port (155 Mb/s) can carry about 32 simultaneous compressed video data streams plus overhead. This layer also maintains information about individual subscriber requests, and provides additional multiplexed stream services such as forward error correction (FEC). It also demultiplexes incoming subscriber data.

Layer 4—Transducers

The final layer consists of transducers which translate the internal server communication to the optical, RF, and electrical standards of the outside world. For the first time since retrieving content from storage, the data has left the server and additional hardware is needed:

- QAM modulators.
- Demodulators for incoming multiplexed data streams.
- Electrical-to-optical transducers for outgoing and optical-to-electrical transducers for incoming fiber optic connections.
- Appropriate electrical transducers for any external equipment of varying standards.

Management—OSS and BSS

Connecting all layers are the operations support system (OSS) and business support system (BSS) communications. The OSS, which enables system management, is frequently a separate, relatively slow-speed network connecting all of the server elements. It reports on status conditions and allows changing the operating modes of individual components. System management is responsible for reporting and handling system element failures and preventing bottlenecks during periods of peak demand. The BSS tasks are typically undertaken by yet another separate slow-speed data network. It tracks system utilization on a subscriber-by-subscriber basis and enables billing by almost any parameter:

- By amount per period (from a mere second to an entire month).
- By system utilization (amount of data transported or processed).
- By percentage of dollar transactions (credit card or shopping center models).
- By title.
- By type of service.
- By almost any criterion formulated by the marketing department.

In the integrated model, these functions are also carried out by the array of interconnected processors that generated the stream and switched and multiplexed them. Since *all* these information processing functions have occurred within the confines of the integrated server and complete knowledge of that processing already resides there, it makes sense to keep network management, the OSS, and business support systems in the server as well, with data spooled out to the appropriate record-keeping MIS network.

The Brains Behind the Integrated IBS

As we have seen, nearly all of the functions carried out by an IBS are entirely controlled and ultimately accomplished by what we have identified as a massively interconnected processor array—the MIPA. (The only exceptions are the read-only, read/write, and hard disk storage at the center of the onion, and the modulation and transduction at the periphery.) The MIPA consists of many identical processors, each with its own RAM memory, embedded in a high-speed communication matrix or mesh. The MIPA approach results in a truely scalable IBS, with an attendant reduction in size and power consumption.

We believe the case for the MIPA is compelling. Consider the fact that a server with multiple processors, whether ganged processors connected by a bus or mounted on a matrix, is, essentially, a micro network. For any network, given *ceteris paribis* conditions, the longer the distance, the slower the link. Thus, proximity buys carrying capacity—bandwidth. When nodes are very close, high speeds can be achieved at modest cost. For these reasons, and the others listed below, we think some form of MIPA architecture is the only way to produce the thousands of simultaneous streams required by even a medium-sized network when digital services are offered to more than a small number of test customers:

(1) It provides the interface bandwidth between the storage elements and the server.
(2) It is inherently segmented to provide the composite throughput unavailable in any other architecture.
(3) It is effective as a switch, to sort the streams to their destinations. It can assemble the streams into any desired format, including ATM. This capability offers enormous savings over using an external switch.
(4) It can provide the necessary output multiplexing.
(5) Since only a portion of the processing horsepower is consumed in generating and sorting streams, there is power left over to provide FEC, encryption, and quality of service monitoring. Thus, the modulators can be reduced in complexity to simple line cards and installed in the same box, rather than requiring space-grabbing, power-guzzling intelligent modulators.
(6) Combined with carefully thought out storage strategies, a MIPA will also handle the requirements for operations system support.
(7) With appropriate firewalls and security to impede hacking, a MIPA can manage the business system support as well. Since all aspects of the session are accomplished within the MIPA, it already has the needed information to track the services that are requested and provided, so BSS becomes a simple function. Most architectures call for a super reliable system to be used for BSS—but a MIPA, with its inherent redundancy, is able to be built to be the most reliable possible computer!

However, MIPA architecture is not the whole answer to building IBSs. As we shall see, the design of the layers of storage is crucial.

Storage Design: The Heart of the IBS

Primary Library Storage

Library storage is physically separate from the server. It holds the assets of the on-demand system in readiness for use. Preproduced material is likely to be distributed on read-only media, such as DVD or write-protected individual magnetic tapes. DVD involves the lowest cost of reproduction (as a stamped, rather than recorded medium) and the highest distribution bandwidth (air freight). The discs and tapes must be loaded manually into a physically secure robotic storage system. (A substantial level of protection is required by the studios which own the content to protect their assets against piracy.)

Read/write media (now tape or hard disk and most likely recordable optical disc in the future) are also part of the library for storage of content for later download or playback. Library storage will accommodate hundreds to tens of thousands of titles available on demand to a subscriber. Our design defines library storage as primary storage, with all other stages considered as caches. We place the primary storage library at each server site, and suggest a transfer rate from primary storage which is fast enough to accommodate fast forward functionality. This means that the number of titles which may be offered in real time to subscribers is determined by the capacity of the archival library, rather than the hard disk storage capacity of the server, as it is in most conventional architectures.

Caching and the Least-Recently-Used (LRU) Strategy

To make an integrated IBS with on-line reliability, the hard disk and RAM memory must be implemented in a very specific manner. Both function as cache memory that is managed automatically using a "Least-Recently-Used" (LRU) algorithm.

Let us first consider the advantages of caching and LRU. As we observed in Part 1, the fact that many customers want the same material poses great problems for conventional server design. For example, as many as 40% of the audience may want to see the same movie; similarly, a majority of Internauts may want to access the Netscape website at some time during their on-line session. Unavoidable redundant storage occurs because popular material must be stored on each group of connected single processors because they don't access a shared library. In addition, a single title cannot entirely dominate the limited number of streams output by each processor group, or customers won't be able to receive anything but the most popular content.

These difficulties mean that both storage and transport bandwidth must be closely managed relative to anticipated demand. Some harried executive whose job depends on the quality of his or her decisions must estimate how popular each type of material will be (which may vary by area of service, time of day, and day of the week), and make an educated guess as to how much demand there will be for the content. In a conventional system that doesn't offer real-time replication, these guesses must be made significantly in advance. The selection is irrevocable, as both storage and throughput bandwidth will be allocated, based on these estimates (guesses). The more popular the executive believes some specific material will be, the fewer choices will be available to customers, since each redundant copy occupies the storage required for additional material. Incorrect decisions result in content not being available to all viewers, which can lead to dissatisfaction with the system and a return to rental store habits.

By contrast, LRU (least-recently-used) is an invisible hand that will manage the system perfectly, even though it has no awareness of content. The actual popularity of requested content will determine where it is stored, ensuring that the most frequently asked-for material will be immediately ready for customers, while the least popular material will be available after a short latency.

The means of implementing LRU is caching. The hard disk arrays are an immediate cache for the primary library storage; the RAM is the immediate cache for the hard disks. Without a cache, a program must be reread from storage every time it is requested because a characteristic of primary storage devices such as tape machines and optical discs is that they transfer only one stream at a time. Thus, if this stream is to feed the server directly, then only one customer can be served: One storage area, one stream, one customer.

With caching, if the requested portion has been so recently read that it is still in RAM, it is served from RAM with no need to access the hard disk. When the RAM cache area is full, and a new portion of the program must be read, the oldest segment of data in RAM is erased by the new content that reuses that area of memory. The same process holds for the hard disks. The most recently requested material is still on the disks; as newer material takes up storage space, less recently requested material is erased and the new content is stored in that area of the disk.

However, suppose there is both a hard disk cache and a RAM cache. The first customer is served by a stream that goes from the primary storage to the hard disk, to RAM, and out to the consumer. The second, third, and fourth customers (up to the maximum number of customers the system will support) will be served from RAM, which allows multiple access within at least a fraction of a second, or from hard disk, which allows multiple access within at least the entire period a movie is in demand: One storage area, two caches, many streams, many customers.

Note that at any moment, the 60 seconds of material cached in one processor's memory is the most recently requested individual seconds from all of the material stored on the array. They are *not* 60 contiguous seconds from a single title or file. This seems strange at first; however, contiguous seconds from the same material are stored on *successive* arrays. Thus, each second cached in a single array controller is from a different title or from widely separated seconds from the same material. As the relative popularity of particular content shifts moment by moment, the holdings of the cache will change automatically. The contents of the cache are always determined by the instantaneous demand for the seconds stored on the array, and never has to be "managed" externally.

This multi-access capability of RAM might lead a designer to consider making all memory RAM. However, this strategy has been tried by at least one vendor, and its cost is too great, approximately 100 to 400 times as much as hard disk storage. By utilizing primary storage and a double-cache hard disk/RAM system, the operator can maximize the cost/speed ratio for the optimum level.

Visualizing the Operation of LRU Caching

In a hypothetical system, the server has enough RAM to hold about 300 minutes of content. Suppose a customer requests a hit movie ranking #1 in popularity or access to the Netscape site. Since both these choices are very popular, they are already stored in RAM and can be sent to the subscriber immediately. The storage of the material in RAM is determined automatically by the popularity of the title and the specific system. Now suppose another customer requests the movie ranking #23 or the Elfnet homepage. These are viewed often enough that they are already cached in hard disk storage, so they are sent directly from there, without generating a library request. As they are streamed out, they are moved second by second from disk to RAM. But since demand for them is relatively low, the RAM in which they are stored is likely to be reused for other material before another request for this content is received.

A third customer requests an obscure Iranian film, movie #874, or the homepage of their old college friend in Poughkeepsie. No one has requested these materials for 10 months (or perhaps never before). The film is stored only in primary storage, and this personal homepage will have to be downloaded from the cousin's server in Poughkeepsie. If a connection cannot be made immediately while the user is on-line, the homepage will have to be stored on a read/write medium, awaiting downloading at the user's convenience. In the case of the movie, after a delay of a few seconds for robotic retrieval from permanent, primary library storage and delivery to the playback unit, the first segment of the actual film is transferred, and then the consumer starts receiving the requested material.

According to the LRU algorithm, all of hit movie #1 and the Netscape website will probably stay in RAM for some time—as long as they retain their popularity. However, the less popular materials, like the cousin's homepage and the Iranian film, will probably be replaced in RAM within minutes, or even seconds, by more popular content. This infrequently-requested content will also be replaced on the hard disk drives by more recently requested material, whenever it is not being used and the space is required. All this is accomplished without any person guessing at the likely demand for provided content. The system itself automatically retains the second-by-second demand for all material, allowing the marketing department to track requests for information, titles, and activities, by time and by consumer demographics.

In this manner, the most popular content (or the last 300 minutes or so of it) is automatically buffered in RAM or hard disk—without any active management. If there were a five-minute spurt of demand for a movie right after the news, only that five minutes would remain in the RAM cache, rather than the entire movie. This "pig in the python" continues to move as time passes—the period of the movie in the cache would continue to change in real time. Individual viewers might jump into or out of the cache as they used VCR-like controls. New popular periods of the movie might emerge with large viewer clusters. LRU ensures that it all happens automatically.

Hard Disk Storage

Dr. Foaud Tobagi of Stanford invented a technique he called "wide striping," which we refer to as vertical striping. It is one of two closely-related striping techniques used for storing material across multiple hard disk arrays. The second technique we call horizontal striping.

Horizontal striping refers to the technique of spreading a file across a small number of hard drives (typically five) for the purpose of increasing storage bandwidth to more closely match processor bandwidth and to increase reliability. It determines the number of disks to be placed in any given RAID array when a fixed amount of content is spread across a variable number of disks in that single array. For example, at four Mb/s, a one-second (:01) allocation would correspond to .5 megabytes of storage. By definition, as a fixed amount of storage is spread over more and more disks, there is a diminishing return for each additional disk, as the seek time from one allocation unit to the next becomes the dominant factor. Two to eight disks seem to be optimal for RAID techniques of increasing speed and using redundancy to protect against a single device failure.

Vertical striping exists for the purpose of load balancing. It allows demand to be spread evenly across all of the horizontally striped arrays in the system. Vertical striping is accomplished by distributing small chunks of every title over all the arrays in the system, rather than storing the title on a single array. This technique requires a system which can reassemble output from all of the arrays into isochronous streams, which can be accomplished only by a massively interconnected processor array architecture. Since a fixed number of simultaneous streams can be supported by each array, the number of arrays and therefore the number of drives in a system is directly proportional to the number of subscribers: The greater the number of subscribers, the more hard disk arrays will be required.

Distributing content over all of the drives allows the system to support any number of subscribers from a single copy of a program. As we have seen with conventional server architectures, storing titles on individual drives or arrays results in uneven utilization of system capacity, the need for outrageously redundant storage, and interactive intelligent management of storage and replication. Vertical striping, LRU caching, and MIPA architecture avoid these difficulties. However, it is important that vertical striping be approached correctly so that it doesn't introduce unacceptable latencies between the time a title is requested and the time delivery begins.

Picturing Vertical Striping

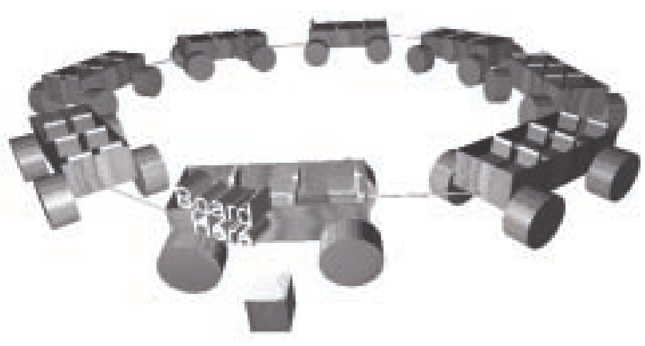
In order to make it easy to visualize the way vertical striping works, we will look at the way a group of individual drives would be organized. Assume that we have about 150 subscribers, and expect a peak demand of 30 streams from our digital system. An inexpensive drive can support about six 4 Mb/s streams simultaneously, so we will use six drives (we'll explain the sixth drive later). We will use a vertical striping chunk size of one second, so that for a given title, the first second will be on drive one, the second on drive two, the sixth on drive two, the seventh on drive one, the eighth on drive two, and so on.

Picture the drives as a circle of wagons, each with six seats, moving past the loading point at a rate of one wagon per second (see Figure 2). Each wagon represents not a drive, but its capacity to accommodate simultaneous viewers during one chunk period. When someone wants to board (start receiving a data stream), they have to wait for an empty seat to come around. If there were only five wagons, and 30 seats to accommodate an anticipated peak load of 30 riders, the last rider might have to wait for the entire circle to pass to claim the last seat. By putting on an extra wagon, and managing the boarding, we can guarantee a seat will always be available almost immediately.

This ability to accommodate customers becomes much more important as the number of wagons increases. For example, with 9,000 riders and 1,500 wagons, the last rider would have to wait up to 25 minutes to board. By adding 20% more wagons, we can guarantee an empty seat on each wagon, and almost no boarding latency.

So much for the wagon analogy. The important thing is this: In building a system, we can take a very good guess as to the anticipated total peak demand. Vertical striping allows us to build a system to meet this peak demand by distributing the demand evenly over all of our resources. Conventional designs incorporating localized storage of content force micro-management of the system title by title, service by service, and moment by moment as the relative demand for different titles shifts. They require wasteful replication of material across as many storage elements as are required to meet the anticipated demand.

Figure 2
Wagon Illustration



Source: S. Rose

A Final Note on the Location of the Server

Schemes for the placement of the IBS range from a single, enormous, centralized facility to miniservers sited in neighborhood nodes. An example of the centralized approach was AT&T's original design that called for locating the server in Manhattan to serve headends around the nation.

Earlier in this article, we called for placing the server "as close to the consumer as possible." However, we did not mean to advocate putting some kind of remote server in a box on the telephone pole outside each house. The optimal location for the server is at the point where no further cost reductions can be achieved by further distribution. This point is at the cable headend or telephone central office, beyond which there is no further expansion of bandwidth to the subscriber, and at which point the operator owns the entire bandwidth to the consumer. From the cable headend downstream to the household, the operator has paid for the entire bandwidth already, and loses income only when it is not used to provide services.

The economies of scale available with an appropriately constructed interactive broadband server imply that it may be smart to consolidate as many headends or COs as possible to be served by one IBS. This move toward geographical consolidation is well underway through a variety of interconnection schemes and other means. For example, in March 1996, Time Warner agreed to swap its systems in Hampton and Williamsburg, Virginia for Cox Cable's system in Myrtle Beach, South Carolina.

A Final Note from the Convergence Front

One interesting conclusion that can be reached from the foregoing discussion: The IBS is the switch. It is even possible to leave out the storage, and use the IBS as an exceptional broadband ATM switch. By the same token, ATM switch manufacturers are moving away from single processor switches at the high end, and toward matrix-based switches. With an appropriate design, it would also be possible to add storage to a matrix-based ATM switch, turning it into a MIPA-based Interactive Broadband Server!

In summary, there are several critical elements to the architecture of an appropriate, scalable, interactive broadband server, exemplified by the cool hairy onion model. First, the number of titles and the variety of material which may be offered is determined by the size of the archive, not the size of the server. Second, the storage is organized so that the server may be sized to meet aggregate anticipated peak demand for all services, rather than having to be micro-managed on a title-by-title or service-by-service basis. Third, a Massively Interconnected Processor Array architecture results in a server that offers genuine economies of scale, while at the same time offering the greatest possible reliability by taking advantage of inherent redundancy and avoiding single points of failure. ntq

Authors' Note—Questions or comments can be sent to the authors via e-mail to Joan Van Tassel at JoanVT@earthlink.net and Steve Rose at roses@maui.com.