Systems

S.M. Weiss
The Merrill Weiss Group

Future television production systems are going to be far more complex than anything we've dealt with in the past. Thus, in order to pull together all the various dimensions of what the EBU/SMPTE Task Force had set out to do, it created the Systems subgroup specifically to concentrate on matters relating to system integration.

This article describes a system model for television production operations, based on the transfer of bitstreams within a distributed studio object environment. It puts forward a migration strategy and also describes a different economic model for equipping television production facilities in the future.

1. Introduction

Traditionally, broadcasting operations have been based on a single distribution channel that produces an integrated continuous programme as its output. More recently, we've seen a proliferation of distribution mechanisms to the consumers / viewers, and there is today much discussion about the need to "re-purpose" programme content for use across these newer distribution channels.

Arising out of this diversity, future television production systems will be quite different in the way they are structured when compared with today's systems. Traditionally, systems have generally been built around tape recorders and linear equipment which run in real time. Systems of the future, on the other hand, will use servers and will be able to move programme items around much faster than real time, or much slower than real time, to take advantage of bandwidth-versus-cost trade-offs that are becoming possible in the new digital world. New workflows will result, and we will have the capability, for instance, of having a series of different processing capabilities located at different facilities, and having those facilities interconnected on an ad-hoc basis under the control of a single operator. This will be achieved by using *object-modelling* techniques. By using the interconnectivity that will become available, we will be able to take a system that is distributed and have it appear on a single monitor screen, thus giving an operator the chance to concentrate on the functionality of the content

Original language: English
Manuscript received: 27/10/98.

and not have to worry about the technical complexity that really exists underneath the whole system.

In the past, when we designed a system, we were concerned about what would be connected to what – the hard wiring of the system more or less determined the functionality that the system could provide. Thus, it was important to keep accurate documentation on which cable went from where to where, in order to find a connection quickly for maintenance or fault-correction purposes. In the future, we will be more concerned about where we need to locate the servers and the various kinds of processors that make up the system. We will also be just as concerned about selecting the right drivers and the right software to load onto those new systems. We will also need accurate documentation if we are to control and maintain the software, its various revisions and its various configuration files – all of which may be spread across different parts of the system.

2. Object models

In order to plan the control and representation of a system, we decided fairly early on in the systems review process to work with *object models*. For those who are not familiar with object models, there is a good description of how they work in Annex B to the Final Report of the Task Force. Basically, we have selected object models both for the control of systems and also for the representation of content.

One thing that an object model does is to integrate the treatment of the processes and the information. Thus we can have a single representation or a single structure that can deal with, for instance, the functionality of a server and the content that is on that server. By using object models we are able to divide the management of those things into blocks which can be further sub-divided. Then, for instance, if we move some programme material from one place to another, we can have the associated metadata transferred with it, if that is appropriate.

Object models also allow us to have updates to the software, and to control the functionality in confined areas of the system without causing us then to have to make changes in other areas at the same time. They also permit us to have extensions to functionality through the use of registries. Ultimately we may get to a point where the appropriate drivers come with the purchased equipment and can be loaded onto any other device that needs to control that equipment – without needing to have a new interface written by every manufacturer. For example, whenever a new tape recorder comes on the market, the manufacturer will supply a driver that can be installed on all the existing editing systems. This flexibility arises from a

Abbreviations

API	Application programming interface	**MPEG**	(ISO/IEC) Moving Picture Experts Group
FTP	File transfer protocol	**OSI**	Open systems interconnection
GPS	Global positioning system	**SDI**	Serial digital interface
IEC	International Electrotechnical Commission	**SDTI**	Serial data transport interface
ISO	International Organization for Standardization	**SMPTE**	(US) Society of Motion Picture and Television Engineers

recognition by many control system manufacturers that there really is no added value for them in writing drivers – it is simply a cost and a drain on resources. If we can set things up in such a way that it only has to be done once instead of being done by everybody, then there are significant benefits both for the manufacturers and the users.

2.1. Distributed objects

Fig 1 illustrates a studio which consists of distributed objects. Inside the small circles are *transaction-based models* – which refer to the kind of control that we have had in the past (where we probably had RS-422 networks running at 38.4 kbit/s, over which could be sent a command that said "play" and another command that said "stop," and so on).

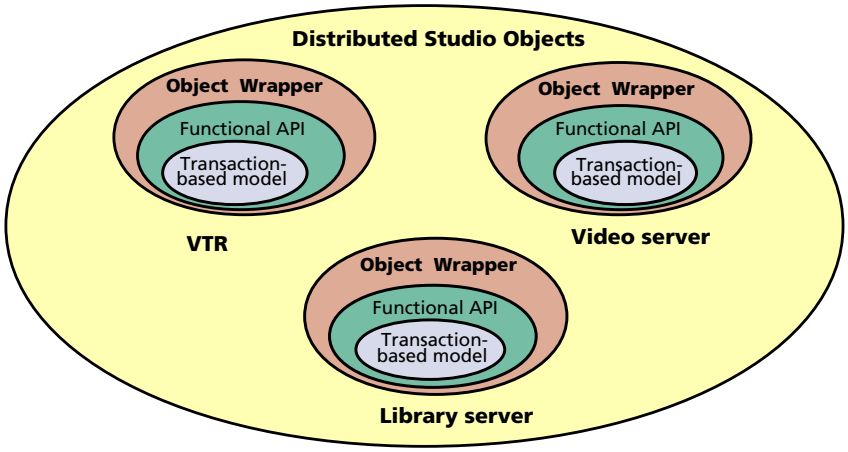

Figure 1
A studio based on distributed objects.

Wrapped around these models are a more-recent phenomenon which has come out of the IT world – *functional APIs* that allow us to make various kinds of calls to devices and to control them on a much more software-oriented basis. Surrounding the functional APIs are *object wrappers* that allow us to have a single approach to a given device type. Thus, in the case of a tape recorder (VTR), if we approach the object wrapper and say "play," it will understand our command and translate it to any chosen machine within the wrapper. By having wrappers appear at different places throughout the system it becomes distributed, and that is why we call these *distributed studio objects*.

2.2. Object registry

Ultimately, to make all of this work, we will have to have a *network object registry*. A sample of the hierarchy of such a registry is shown in *Fig. 2*. *Object categories* in the diagram are shown in green, while *specific objects* are shown in brown. At the top of the hierarchy, we have *studio objects* and below that we have *devices* and *services*. Under devices we have such things as *file servers*, *transmitters*, *monitors*, *network routers* and *tape decks*. In the case of

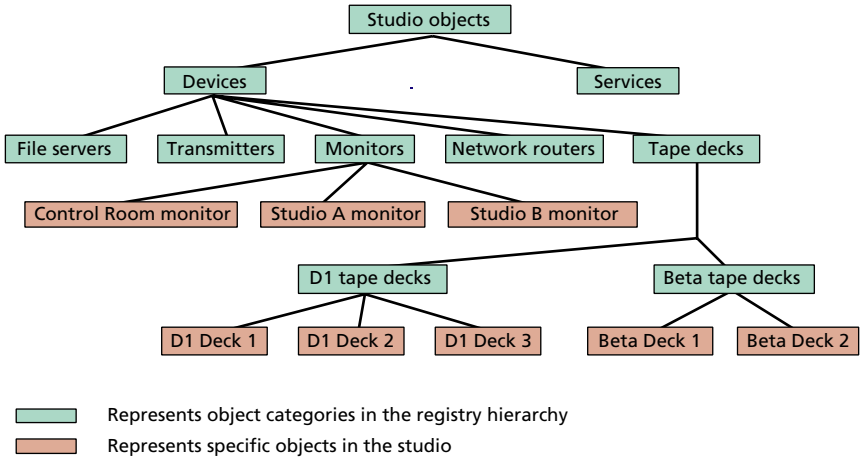

Figure 2
Network object registry.

monitors, just as an example, we have a *Control Room monitor*, a *Studio A monitor* and a *Studio B monitor*. In the case of tape decks, you can see that there are three D1-type tape decks and two Beta-type tape decks in this particular system.

All this equipment would appear in a central registry which would allow us to control them from anywhere on the network, to learn about their capabilities and to make use of them. Another way to look at this arrangement is in terms of a *networked studio* where we have a studio network that views, equally, those objects that represent physical devices and those objects that represent software entities (*Fig. 3*). They are all part of the network and are perceived equally, and that is a major reason why object-modelling techniques work very well for studio applications.

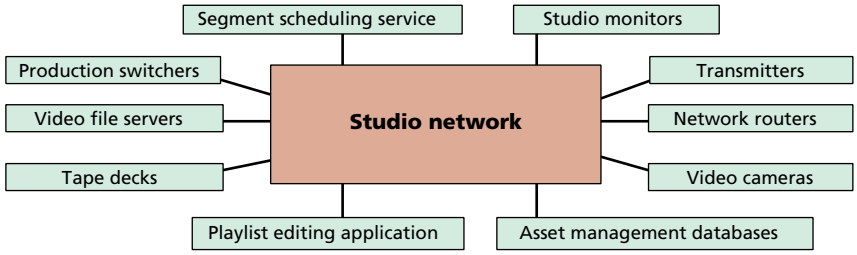

Figure 3
Concept of a network studio.

3. System model

Over the long term, if there is one diagram that will come to represent the work of the Task Force, it is probably the system model (*Fig. 4*). Some of the other Task Force subgroups may well have their own favourites, but certainly within the systems area, this is probably the one diagram that tells the biggest part of the story.

Fundamentally, in television production, we have a number of *activities* which are represented on the model by the blocks distributed across the first vertical plane (*Video Essence*). Notice that there are equivalent activity blocks on each of the other vertical planes behind the first one, i.e. the *Audio Essence*, *Data Essence* and *Metadata* planes. Cutting through all the activities attached to the vertical planes are four horizontal *Communication Layers*, namely *Applications*, *Network*, *Data Link* and *Physical*. And underlying these two sets of orthogonal planes and tied to all of them is a *Control and Monitoring* plane.

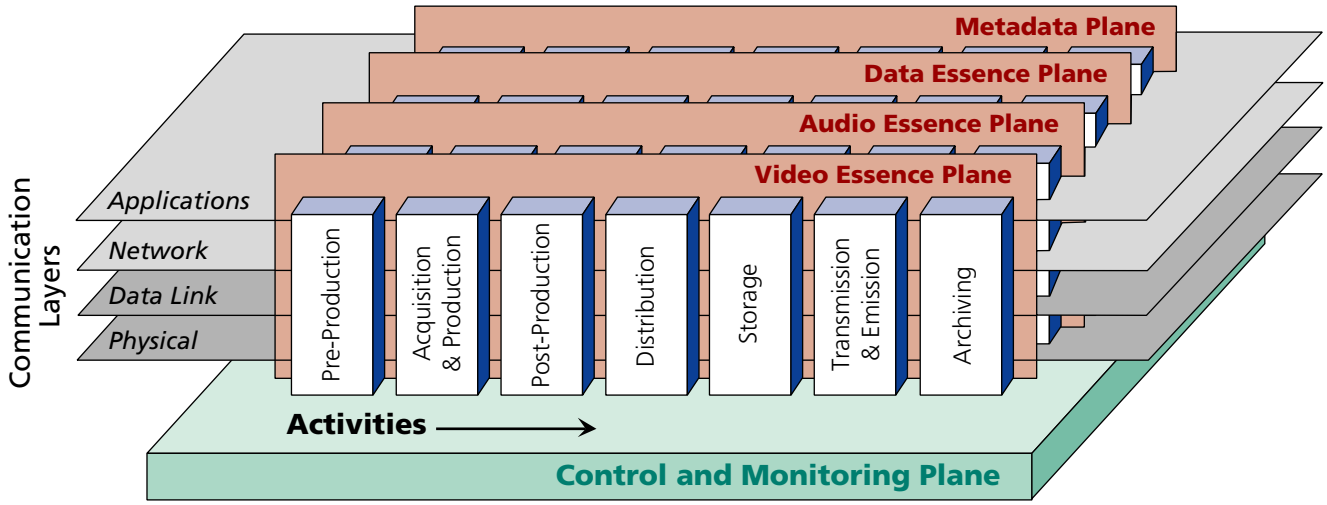

Figure 4
The system model comprising activities, planes and layers.

3.1. Activities

Activities are basically the various stages of "production" that we are familiar with; for instance, we start with pre-production and move on to acquisition and production, then on to post-production, distribution, storage, transmission and archiving. We can take those seven activities and use them to model virtually anything that we wish, in terms of television production. Of course, these definitions will be specifically different if, for example. we are referring to news rather than a situation comedy, but in general they will be applicable to all instances of production.

3.2. Essence and metadata

The next thing we want to look at on the model are the vertical planes – there are four of them. The first is *Video Essence*. If you have read the first Task Force report (April 1997), you will remember that *Essence* and *Metadata* together are equal to *Content*. In the terminology of the Task Force, when we talk about video essence, we are referring to streams of video only. Similarly, *Audio Essence* refers to audio streams only. Video and audio files, on the other hand, are treated as *Data Essence*.

Data essence is information that inherently has stand-alone value; it is not video essence or audio essence. Thus, for example, teletext and closed captioning (a real-time subtitling system used mainly with analogue TV systems in 525-line countries) are considered to be data essence as they both have meaning on their own.

Finally we have metadata, which has no stand-alone value. It is tied to the video, audio and data essence in some manner. A good example of metadata is timecode: when taken by itself, it has no meaning but, in association with a frame of video essence, it allows that frame to be identified.

3.3. Communication layers

Looking again at our system model in *Fig. 4*, we have various communication layers that are represented by the horizontal planes which transect the model. These communication layers are in many ways similar to the familiar ISO/OSI seven-layer stack but, in our case, we have decided that there should be just four layers to represent a television facility, or a television system. We start with the *Physical* layer on the bottom which carries the electrical and mechanical characteristics of the system, or the interconnection. We then have a *Data Link* layer that involves the protocols that interconnect the *directly-connected* devices. So if a controller is connected to a VTR, the data link layer controls the direct communication between those two items. The *Network* layer above that controls the protocols between *indirectly-connected* devices. Finally, on the top, we have the *Application* layer that deals with both specific applications and what the ISO/OSI model would call the presentation layer.

3.4. Control and Monitoring plane

Underlying the aforementioned elements of the model, we have the *Control and Monitoring* plane. In most respects, it touches all the other elements of the model, basically providing co-

ordination between everything else that sits above it. In the case of transfers, storage, manipulation, monitoring, diagnostics and so on, the control and monitoring layer also provides overall management of content across the various activities, planes and layers. Thus the control and monitoring plane not only performs a control function, it also provides a management function. For example, if we need to allocate bandwidth between devices, this matter will be handled by the control and monitoring plane.

Ultimately, this plane provides the interface to the human operators of the system.

4. Television operations

The next thing to consider is what we can do with the system model, in terms of television operations.

There are various aspects of operations that are of concern to us:

- control;
- monitoring, diagnostics and fault tolerance (which, when taken together, are related to control but which otherwise are separate from it);
- data essence and metadata management;
- content multiplexing;
- multiplexing essence into containers;
- timing, synchronization and spatial alignment.

4.1. Control

Control can be split into a number of types; for example, there is *strategic control* which is at the fundamental planning level; *tactical control* which determines how we are going to make things happen, and *peer-to-peer control* where one device sends information to a second device in order to make control possible and to achieve the tasks that the second device has been designed to do.

Managing resources is another type of control. For example, the control of bandwidth or data space in a server, or what you are going to do if you have only a certain time availability on equipment, are all resource management issues.

Yet another type of control is the physical control of networks. If equipment or subsystems are to be connected together, we have to make sure that the control system understands what is linked to what, what can connect to what and, if something cannot be directly connected to something else, how we are going to send our content there. For example, if our content was recorded on a DV tape, but we are using an MPEG-based editing system, the control system has to know that when the DV content is sent to the editing system, it must be passed through some sort of a translating device on the way – so that when it arrives at the editing system, it is in a compatible format.

There are various forms of control implementation, as already discussed above in terms of object models. The Task Force will be implementing control through the use of various logical control layers, and that ties in with some of the layering already mentioned above.

4.2. Monitoring, diagnostics and fault tolerance

When we think about monitoring and diagnostics, we are really referring to a feedback process that allows us to get information about what has happened and to feed this back to a controller. We must be able to predict failures by monitoring our systems and knowing how they are performing. We must also be able to carry out diagnostics on-line, while the system is in operation, and off-line by taking part of the system out of service.

Redundancy has typically been used in broadcasting facilities to provide *fault tolerance* in systems. However, fault tolerance has also been achieved through the segmentation of systems so that if there is a failure in one segment of the system, it does not have to affect the entire system. Fault tolerance in systems has also been achieved through various monitoring and verification processes.

4.3. Data essence and metadata management

When we think about data essence and metadata management, again we are referring to the layered structure of our systems model. We have a number of ways of handling content transfers including:

- file transfer mechanisms;
- streaming.

Streaming is fundamentally an isochronous process which allows us to send content at a continuous rate from one place to another. File transfer, on the other hand, allows us to re-send the content if there has been a problem. They are very similar to the typical file transfer that occurs when downloading from an FTP server on a computer network.

It must be possible to link together data essence and metadata when necessary, but to keep the metadata in a separate place so that it can be treated within a searchable database. We also have to be concerned about whether the metadata that relates to particular essence is permanent or temporary. For example, when we send a control command as metadata along with essence to a particular device, that command only has a life from the time it is issued until the time it is executed. There are other metadata elements that may have much longer lives and again others that may need to be changed as they pass along the production process. We have to be able to recognize what the lifetime of a metadata element should be, and to properly maintain it until the end of its useful life.

4.4. Content multiplexing

Next we come to content multiplexing. Here we can have either a single-step multiplex or we can have cascaded stages. Several of the possibilities are as follows.

- We could have individual channels that have to be put together to form a multiple-channel component. An example would be multiple audio channels that multiplex together either to form a stereo pair or a 5.1 format surround sound audio signal.
- We might assemble an SDI stream or, similarly, an SDTI stream from multiple components.
- We might form a content package from multiple individual components.
- We might form a multi-programme package from several different existing single-content packages.
- We could take elements that are already multiplexed into various packages, strip them out of those packages and put them together in yet other packages – this is often called *grooming*.
- We might assemble multiplexes for faster-than-real-time transfer, or for slower-than-real-time transfer.

4.5. Multiplexing of essence into containers

We have to be able to transfer content between multiplexes. If some content comes in as part of a multiplex and needs to go out as part of a different one, we must be able to handle that – again, this is in the domain of "grooming."

If we have a certain amount of content that we are trying to convey across a channel, but if that content does not fully fill the channel, then we may want to send additional data along with the content in order to fill the channel – this is called *opportunistic data*. We have to be able to control this use of the channel, and there is already some standards work going on to support such applications.

We also need to be concerned about the multiplexing of signals originating in different systems and formats, e.g. DV compression and MPEG compression. So how do we put this type of multiplex together and convey it from one place to another? We would not wish to have separate infrastructures to handle different formats or systems. The Serial Data Transport Interface (SDTI) is one very important solution to this problem, the completion of which was due largely to the efforts of the Task Force.

Finally, in this section, we must consider statistical multiplexing. If we have a number of things we wish to convey through a channel and we want to be able to share that channel in the most efficient way, then we will want to allocate the bandwidth dynamically to each of the encoded signals – without in any way compromising any of the respective content elements.

4.6. Timing, synchronization and spatial alignment

When we consider putting all these elements together into a system, we have to start thinking about issues of timing and synchronization in rather different ways from how we thought about them in the analogue days. Of course, we will still need reference signals. Up to now, we've used the black-and-burst reference signal of analogue systems. The thinking until fairly recently was that the same black-and-burst would even do for the digital world, but that thinking is now changing somewhat. There will probably need to be an augmentation of the information that can be carried in a black-and-burst signal to enable us to achieve some additional timing and sychronization objectives.

We are also likely to require some form of absolute time reference – something that has not been needed in the past. This, at least in the thinking of the Task Force and in other interested bodies, is likely to be a GPS-based reference signal.

In analogue systems, we have been able to use frame synchronizers to achieve temporal alignment. When the clocks at the two ends of a circuit were running at slightly different frequencies, a buffer placed in the middle could overflow or underflow and, by simply repeating or skipping a field or a frame, the problem could be fixed, albeit with a little hiccup in the signal if somebody was watching very closely.

In the compressed digital domain, if we were to use a buffer to absorb timing differences – for instance those caused by Döppler shift in satellite links – we would have a much more serious problem if the buffer were to overflow or underflow. There is no simple way to jump forward or backward in small increments, and the irregularity of the data repetition exacerbates the matter. However, the application of an external frequency reference to all the systems in a network will keep them from drifting apart, eliminating the possibility of buffer overflow and underflow, so long as the buffers are large enough to absorb the variations that occur in the network delay.

Temporal alignment of compressed signals as they pass through concatenated compression processes is an important consideration if quality is to be maximized. This means ,basically, that we must code a given frame in the same way each time – so that an I-frame remains an I-frame, for instance. With the many compression schemes that are likely to be cascaded, it may be impossible to maintain temporal alignment rigidly, but it should be managed to the best extent that other considerations allow.

When compressed signals are to be processed in certain ways, it may be desirable to constrain the compression systems in ways that optimize the efficiency through a system, even though the compression process itself may operate at less than maximum efficiency. Thus it may be best to use constant values for the number of bytes and the number of frames in a GoP when the primary objective of a process is editing. This will reduce compression efficiency by wasting the bandwidth but may yield improved overall efficiency.

Latency is the delay that occurs between the input to a process and its output. Compression, by its nature, has associated latency. The greater the amount of compression, the higher the latency is likely to be. When playing back recorded content, latency can be compensated for by back-timing the start of the playback. When live programming is compressed and interactions are required between participants in different locations, latency can become a significant complicating factor. Special attention is required when using such techniques as audio clean feeds.

When compression systems are concatenated, spatial alignment of the image – so that block and macroblock boundaries always fall in the same place – is an important consideration in maximizing the image quality. This can be very difficult to achieve in an environment of mixed compression schemes. Within a single family, it should be achievable and is recommended.

During a transition period that is likely to be somewhat drawn out, both analogue and digital signals will be in use in hybrid facilities. In such operations, account must be taken of the time required to convert between signal forms; the video, audio and data must remain time-aligned despite passing through different processes. It will be necessary periodically to restore their time relationships in order to keep their divergence within acceptable limits.

5. Interconnection options

In a layered system, such as that depicted by the transecting layers of the system model in *Fig. 4*, it is possible to use any of several interconnection solutions at each layer of the stack. The total number of possible implementations of any portion of the system for which individual selections can be made then becomes the product of the number of choices at each layer. When there are more than a couple of choices for each layer, the number of potential system design combinations can quickly become staggering.

Understanding the confusion this could bring, the Task Force developed the concept of a matrix of preferred implementations. The matrix listed an array of applications for which implementations would be required and then indicated one or two combinations of choices at each layer that were to be preferred over the other possibilities. It was not possible to complete this work within the time-frame of the Task Force's efforts. Hence, only a general outline was provided, along with a number of definitions, in the expectation that the continuing work within the SMPTE would complete the task. The goal is to provide the industry with a series of templates that can be used for the different applications that will arise.

Some of the definitions of transfer types that came out of the work on implementations can help us to differentiate between the types of applications that may require different solutions. In particular, the concepts of "hard real-time," "soft real-time," and "non real-time" should be understood. Real-time here means that the transfer occurs in the actual time in which a process occurs, i.e. at 1 x play speed.

Hard real-time operations must happen *at* a certain time; there is no possibility to repeat and there may not be a chance to re-do (as, for instance, in the case of a live event). Hard real-time operations demand the highest priority.

Soft real-time operations must happen *by* a certain time, and there may be an opportunity to re-do. When the time allotted for soft real-time operations runs out, they become hard real-time.

Non real-time operations need not be completed within any time constraints. They are typically file transfers that can be either faster or slower than real-time.

6. Migration

The Systems subgroup recognized that making the proposed change in the approach to control mechanisms is revolutionary in concept. It requires the adoption and widespread use of standards in an area that previously has seen little use of the standards that existed, however painstakingly they were developed. Thus the subgroup conceived a migration strategy that would serve (i) to make the transition reasonable to accomplish and (ii) to build confidence among manufacturers – in particular that their investment in using standards for control would reap benefits in the long term. This was coupled with a recognition that the manufacturers of control systems are starting to realize that they derive no benefit from having to write drivers for each new device that comes on the market; their added value comes instead from the application and the user interface, which is where they should concentrate their efforts. The increasing complexity both of future systems and of future controlled devices will make a standardized set of protocols increasingly attractive for all.

The first step along the migration path is the placing of all existing control protocols in the public domain and making them accessible through a single point of contact. Using the Internet for accessibility will permit developers to obtain accurate information on protocols and will also provide a mechanism for disseminating information about protocols, even after their developers have discontinued supporting them. This step will build confidence in the concept of open control standards and will demonstrate an industry commitment to the process.

With existing protocols in the public domain, it will be possible to develop a set of *Essential Common Protocols* that build on the existing protocols, and an *Object Reference Model*. The Essential Common Protocols will, in turn, serve as the basis for a *Distributed Object Model* mechanism for control. Once these pieces are in place, it will be possible to interconnect both new equipment that makes use of the object-model approach directly, and older equipment that does not. This can be done through the use of control systems that support both the old and the new protocols, controlled devices that support both the old and the new protocols, or proxies that translate protocols.

7. Economic model

The history of the broadcasting industry has been one where the costs of both development and support of equipment are built into the initial sales price of the hardware, while the software – including maintenance and upgrades – is provided at low or no cost. When manufacturers wanted to update features, they sold completely new units, even when in reality only the software was being changed. With the value and cost of digital equipment now primarily lodged in the software rather than the hardware, other industries (such as the IT industry) have long since used a different model – in which the hardware and software are sold and supported separately.

The Task Force perceived that the current economic model in the broadcasting domain no longer serves the industry well. It causes hardware to be replaced when it is not necessary to do so. It fails to recognize that the value in products is now dominated by the software. It also fails to recognize the real costs of developing and supporting the software. All of this results in much more expensive equipment, which must be capitalized at the beginning of its life cycle when the cost of money generally is highest, thereby compounding the discrepancy.

The Task Force therefore proposes the adoption of the computer industry model in which hardware and software are unbundled. This is expected to encourage better software support from the manufacturers, by rewarding them for upgrades and maintenance. It should be attractive to users because it will lower the initial cost of the equipment and spread the total costs of ownership over the life of the equipment, thus reducing the overall financial impact.

8. Standards

As in the rest of the work of the Task Force, the ultimate focus of the Systems subgroup was on the eventual development of standards. It is expected that the standards development effort will largely be carried out by the SMPTE. To that end, a series of requirements for standards at the systems level is included in the Final Report. It is divided into two categories: those items that were sufficiently well understood that they could be turned over to the SMPTE at the earliest possible time for standardization work to commence, and those items that required more development within the Task Force before being turned over to the SMPTE. These latter items

***S. Merrill Weiss**, a graduate of the Wharton School of Finance and Commerce of the University of Pennsylvania, has spent more than 30 years in the broadcasting business, starting as a technician and working his way up through the ranks in maintenance, system design and management. Over that time, he was responsible for the design and construction of a number of major television station and network facilities. For the past eight years he has been a consultant to the industry in the areas of electronic media technology, technology management and general management.*

Mr Weiss has been involved in the development of standards for the television industry for over 21 years. He organized and produced the tests that led to CCIR Rec. 601, the foundation for all digital television standards since. He has contributed to many other standards in areas including digital control, the Serial Digital Interface (SDI), the Serial Data Transport Interface (SDTI) and Video Index, a precursor to the more generalized form of Metadata.

Currently, Merrill Weiss serves the SMPTE as Engineering Director for Television, in which role he is responsible for organizing all of the SMPTE's television standards development activities, and he chairs the SMPTE Television Steering Committee. He is a Fellow of the SMPTE and a recipient of the David Sarnoff Gold Medal Award.

Mr Weiss served as the SMPTE co-chairman of the EBU/SMPTE Joint Task Force and also chaired the Systems subgroup.

have now been given to the SMPTE with the release of the Final Report, and the SMPTE is in the process of taking them over as standardization projects.

To help maintain the close working relationship that has developed between the EBU and the SMPTE during the Task Force effort, the SMPTE has also brought several highly-regarded EBU participants into leadership roles within the SMPTE. This will allow the EBU to help in guiding the work going forward and will assure continuing close liaison between the two organizations.