Interoperability Requirements For Interactive Cable

John Richardson
Dave Williams
Robert Adams
Intel Corporation

Abstract

The term 'interoperability' denotes a concept quite familiar to the computer and communications industries. Interoperability is the concept of systems working together for the purpose of transparent exchange of information. It is a very difficult thing to accomplish and is full of both technical problems and political pitfalls. The computer and communications industries have learned, with great pain, that interoperability must be addressed at several different levels to be effective. The Cable industry is about to come face to face with these issues as it tries to implement a broadband network to the home.

This paper will explore the essential elements of interoperability and outline some of the essential requirements to stimulate debate in the cable industry. It presents some of Intel's findings in developing systems for cable that are open to interconnect with other systems and will cover those areas considered most crucial to the achievement of interoperability in interactive cable.

1. WHAT IS "INTEROPERABILITY"?

Interoperability is one of those terms where "everybody understands what it means but nobody can define it." It is often defined by example. A remote is said to be interoperable with a TV if it can control the TV. This is a rather narrow interpretation. If the remote and the TV were designed together, then interoperability is not too interesting. However, if the same remote can control many TVs, then interoperability becomes a market factor and interoperability becomes interesting. This broader interpretation of interoperability has two requirements:

- (At least) Two devices must communicate, and
- Those devices must have been independently designed.

Perhaps this terminology from the computer world below will help with the distinction:

- *Portable* — typically applied to software that can be moved to another environment and it will work. Binary portability means that the software can be moved to any system and it will work without modification. Source portability means that the original program needs to be recompiled in the new environment for the software to work, but that no source changes are required.
- *Connectivity* — this term is applied to multiple systems that are designed to work together (for example, a TV and remote from the same manufacturer)
- *Coexistence* — groups of systems that provide connectivity, but do not interoperate (for example, 2 TVs and 2 remotes where each remote only works with one TV)
- *Interchangeability* — systems that have identical functionality and interfaces
- *Interoperability* — multiple systems that were independently designed to work together

The cable industry is striving for *interoperable* systems — not necessarily limited to *interchangeable* systems. For example, a set-top converter might offer an electronic program guide. A PC connected to the same cable system that was able to also read the electronic program guide would be interoperable with the set-top, though not interchangeable.

Interoperability is required when there could be incompatible solutions for the same problem. Interoperability is achieved through standard interfaces.

Many of today's standard interfaces are actually a compatibility layer that translates between one interface and another. A *compatibility layer* is software that translates between one interface and another. In the "real world", a language translator serves this function. Using a translator, neither the speaker or the listener need to change. This is important in the computer world, because it supports diversity, and diversity allows for innovation. Interoperability should not require that two solutions be identical, just that a developer can easily take advantage of either solution.

Complete interoperability could lead to one or more of the following:

- The ability to move cable box from one house to another
- The ability to have settops and computers share the same network
- Consumer choice about what to devices plug into the cable network — but any device will work

2. DISCUSSION FRAMEWORK

To provide a framework for the discussion below, we must agree on some basic concepts. These concepts have been the source of much debate in the cable and computer industries. I do not claim to have *the* answer. Instead, I will simply provide working definitions relevant to the discussion of interoperability.

It's not just video any more — the cable system is evolving from the distribution of entertainment video into a full service network. This network will include today's video, along with digital video, computer data, telephony and digital multimedia services.

Full service is not just a spiffy video selection system — data in the cable plant will be used for far more than video on demand and electronic program guides. Data will be the basis for a whole host of rich interactive services. Often cited examples include shopping, tele-commuting, personalized news, personal messaging, and interactive games.

Tomorrow's applications won't look like today's computer interfaces — This interactive, full service network will be competing for the consumer's leisure time. Its face will be competitive with today's TV -- fast moving, engaging, with sound, action, and rich colors.

It's the Services, stupid — Data connectivity is not just pushing bytes. Consumers are not interested in just connectivity, they are interested in services. They want to be able to do new and interesting things. Consumers will need to be isolated from the details of network implementation and management. They want to believe that they are interacting personally with the service of their choice.

Services will come from far and wide — while many services will be provided at the head-end, many more will come from regional or national distribution centers. The full service network will need to connect to these remote service centers.

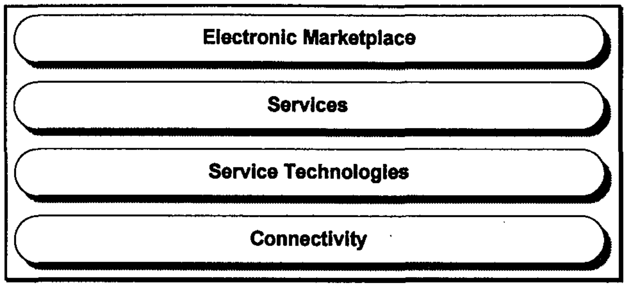
New billing models will be possible — by providing computer-controlled connectivity, control and usage information will be available with more detail and accuracy than every before. This will enable billing systems based any combination of subscription, time, per-use, resource usage, application activity (e.g., when you use a certain part of the system), electronic transactions (e.g., buying the right to view a photo), and, perhaps, supplemented by advertising.

3. THE LAYERS OF INTEROPERABILITY

These new applications will need to interoperate at all levels. If one layer is missing, the whole capability falls apart. For example, you can pick up the phone and call anywhere in the world. However, that doesn't mean that you'll be able to carry on a conversation with whoever picks up the phone.

In this section we will build interoperability from the bottom up. We will start with the most fundamental layer (electrical) and work all the way to the most complete (interoperable applications). In a later section we will provide an overview of the relevant standards for each layer.

Connecting services together is a complex task. At Intel, we use the model below to help organize this connectivity so that we can focus on key problems independently. This model is based on standards and experience from both the cable and computer industries. It is not meant to be a rigorous treatment of networking, but instead a framework for discussion. Those readers familiar with computer networking will recognize this as a collapsed version of the ISO stack. Those familiar with cable TV standards will see that the model is largely built on top of existing industry standards and practices.



3.1. Connectivity

3.1.1. Electrical

Today's cable TV provides a basic framework for looking at electrical connectivity — electrical levels and timings are specified and frequency spectrum allocation is determined. For example, in the US, downstream bandwidth is allocated in 6 MHz units. Depending on the individual cable plant, downstream bandwidth may be available anywhere in the 50 MHz to 750 MHz range. In most cable plants that are 2-way capable, the upstream bandwidth is limited to the 5 - 35 MHz range.

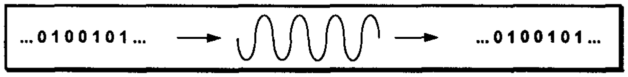
It should be noted that individual cable operators are the ones who actually assign the specific channels to various services. Standards may specify the available options, but cable operators will determine the service mix based on their particular market and installed technology.

3.1.2. Modulation

The next step is deciding how to represent binary data in these channels. This is the modulation scheme.

Modulation provides a method for sending binary data over an analog medium. For example, a very simple modulation scheme might specify that an 1800 Hz signal for 10 mS represents a zero and a 2400 Hz signal for 10 mS represents a one. Using this scheme, one could send 100 bits in a second.

Today's modulation schemes are far more sophisticated, utilizing amplitude, frequency, and even phase to communicate data. Modulation scheme selection is based on carrying capacity, immunity to noise, and cost to implement. Current modulation schemes include QAM (quadrature amplitude modulation), VSB (vestigial side band), QPSK (quadrature phase shift keying), and FSK (frequency shift keying). Each has its strengths and weaknesses. Currently VSB and QAM are used primarily for downstream transmission, while QPSK is used for upstream.



Once we have a method to send streams of bits down the wire, the next step is selecting a coding scheme that allows the receiving unit to tell data from idle noise. A coding scheme typically involves specifying a unique pattern that begins a bunch of data and a similar pattern that signifies the end of the data. For example, one system specifies that each data burst begins with 24 bits of alternating zeros and ones.

Many of the systems in use today also select a coding scheme that includes some error correcting capabilities. For example, an error correcting scheme may use 6 bits to code a 4 bit value. If any one of the 6 bits has an error, the receiver can detect that an error has occurred and, if the error is only one bit gone wrong, can decide what the actual data *should* have been. Error correction schemes that allow the receiver to determine the correct data, even in the presence of small errors, are called *forward error correction (FEC)*. FEC is particularly important for time-critical information, since the receiver doesn't have time to say, "I'm sorry, Mr. Sender, I didn't get that last bit of data, could you please send it again?" The science of error correcting codes comes largely from the telecommunications industries and is the subject of a vast amount of research.

ABCDEFGHIJKL → ABC DEF GHI JKL

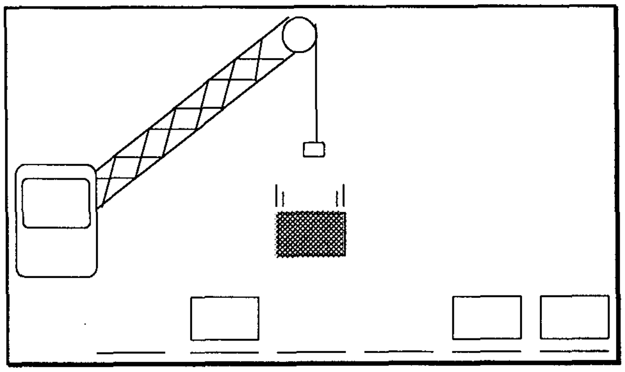
Rules are also set up to define how many characters can come "in a row". These are called packets and the rules specify the largest number of bits that will come across in a group. Special codes are allocated to mark the beginning and end of a packet.

At this point we have defined enough to send packets of data.

3.1.3. Media Access Control

The next step is to figure out how to share the data channel. Because spectrum is a precious resource, we must be efficient in its utilization. This is even more important in sending upstream data, since all consumers must share the scarce 30 MHz of bandwidth. This means all consumers using all services, including pay-per-view television, telephony, wireless communication, and computer data.

There are a number of standards that define media access control. For example, Ethernet sets rules that say anybody who wants to send something should listen first (to make sure nobody else is in the middle of sending) and, when the line is clear, go ahead and send a packet. Of course, there are complex rules about what happens if two devices start to send at the same time.



In the world of cable, access control is a much larger issue for the upstream connection than it is for downstream. For downstream there is only one unit "talking" -- the device at the head-end. For upstream, however, every home could potentially have data to send.

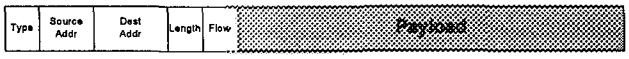
The upstream problem is aggravated by the fact that individual homes in the cable plant can't "hear" all the other homes. This is because of the architecture of the coax plant and amplifiers. Suffice it to say that this makes using the scheme that ethernet uses more difficult (though not impossible).

Sharing upstream cable connections is an area where no standards exist. There are many solutions that have been built (or at least tried in the lab), but none is acknowledged as clearly the best. For the industry to achieve interoperability, upstream media access control will be a major obstacle.

To continue building our interoperability layers, we now have the ability to put a packet onto the network.

3.1.4. Packet Layout

The next step is to decide what the data in the packet means. Packets are defined to have some *header* information that is used to deliver the packet to the right place at the right time, and the rest of the packet is known as the *payload.* Part of the header is reserved for addressing information. This provides a way for an individual network component to determine whether the packet is of interest or not. In data networking, packets include information about who sent the data and who is supposed to receive it. These are referred to as source and destination addresses.



Certain types of data (broadcast video, for example) don't need both a source and destination address. The destination is "anybody who want's to see the video [and is authorized to do so]". The receiver selects the desired stream based on the source address only (e.g., "this is HBO").

This layer has many incompatible standards. There are a number of packet layouts defined by the cable industry for digital video transmission (e.g., MPEG). The computer industry has standards such as ethernet, token ring, ATM, IP, IPX, SNA.

3.1.5. Layering — Division of Labor

Layering is used as a way to isolate the details of connectivity from the software that takes advantage of it. In the computer world, a PC might see a file server as just another disk drive. Any program that wants to get information just reads and writes files as if they were local. However, the next layer down worries about *redirecting* those file access requests to a file server running on another machine on the network. The redirector is only responsible for intercepting the file requests and saying to the next layer down, "please send this request to the file server". The next layer down is charged with making sure the request gets there, retransmitting the request if the file server takes too long and didn't appear to hear, and making sure that the response from the file server has the valid data.

So, above we have created three layers:

- application (thinking it's reading and writing local files),
- redirector (pretending it's a local disk and sending requests to the file server), and
- network (making sure information is reliably sent between the client PC and the file server).

Notice the division of labor — the application doesn't care how its requests get carried out, it just wants the data, while the network layer doesn't care what information it needs to transfer, it just cares that the information gets to the right place and that it's accurate. This is a powerful concept and very important to networking. Each layer has a single job to do. It relies on the support of the layers below to perform their job, but the basic job of a layer is to provide services to the layers above.

Let us return to our packet example from the last section. As information is passed *down* the networking stack, each layer treats the request from the layer above as data (payload), adds its own control information around the data, and passes the request to the layer below. At the bottom of the stack, the physical layer gets a bunch of bits that it needs to send out on the wire. On the receiving end, the reverse happens. Each layer of the stack looks at its control information, makes sure that it does its job, and passes the payload to the layer above.

So, for example, TCP is responsible for the reliable delivery of information to the other end of the connection. IP has the responsibility of taking a single packet and asking that it be sent over the network. IP is refereed to as a datagram service — it will put the packet on the wire but it won't guarantee that it gets there. So, TCP must pass a packet to IP, and mark it so that the TCP layer of the stack on the other end of the connection gets that packet. The receiving TCP layer gets the packet and sends an acknowledgment to the sender saying that the packet was successfully received.

If there is a problem and the packet doesn't make it, the sender's TCP will recognize that "it's been too long — maybe the receiver didn't get my data, so I'll send it again." This is how TCP makes sure that it can reliably deliver data — it has the receiver send an acknowledgment back for each packet. Note that the IP layer doesn't care what TCP does. It gets a packet and does its best to send it but doesn't really worry about it if the packet doesn't make it. TCP, on the other hand, doesn't worry about how IP delivers the packet. It just passes the packet to the IP layer and counts on IP doing the rest. If IP doesn't get the packet through in a "reasonable" amount of time, then TCP will try again.

It should also be noted that IP doesn't really care what medium the packet is sent over. If the layer below IP uses ethernet, token ring, or ATM, the packet will still get to the receiving end. In fact, if the packet is sent part way over ethernet and then the rest of the way over ATM, IP still won't care. The actual method for packet delivery is the responsibility of the layers below IP.

This is a good example of how the computer industry has used layers to hide differences. The upper layer formulates a packet and asks the next layer down to deliver it. It is the responsibility of the next layer to worry about actually how to deliver the packet. The upper layer only cares that its counterpart on the receiving system gets it.

To resume our narrative, we now have achieved the ability to send a packet to anybody on the network. This provides basic connectivity. We can now start to "do something"!

3.1.6. Protocols

Protocols are conventions for interaction. We have so far achieved the ability to send a packet to somebody on the network. What will they do with it? How will they know what it means? How can we be sure they got it? That's where protocols some in.

Let us invent a protocol that will allow you to ask your bank for a list of the last 5 checks you wrote. In our protocol, the bank's computer hangs out waiting for a request from you (or anybody). When you formulate your request, you need to tell the bank computer what you are asking for (last 5 checks, please), and what your account number is. (We'll ignore the security implications for now.) You write this information into a packet, address is to the bank's computer, and ask the next layer down to deliver it.

Source = You	**Dest = Bank**	**Request Last 5 Checks**	**Account number**

The bank's computer gets your request, looks up the information on its database, and creates a response. The response is formatted so that there are 5 entries, each one has a 5 digit check number and a 10 digit amount. The bank's computer gets your address from your request and sends the response.

Source = Bank	**Dest = You**	**Check 1**	**Check 2**	**Check 3**	**Check 4**	**Check 5**

One problem we might have is that the connection is noisy. Suppose your response got slightly garbled and it says you wrote a check for $1,000,000. To deal with that problem, we will modify our protocol. We'll have the bank computer put the 5 checks in, as before, but it will also add a 6th entry that is the total of the 5 checks. That way when you receive the packet, you can add up the 5 check amounts, compare their sum to the bank's total, and see if they match. If they don't, you know you got a bad response and you just ask the bank to try again.

Source = Bank	**Dest = You**	**Check 1**	**Check 2**	**Check 3**	**Check 4**	**Check 5**	**Total**

Why is this important to interoperability? Well, interoperability requires that both the sender and receiver be following the same protocol. The sender needs to know how to make requests of the receiver and the receiver needs to know to listen for those requests. They each need to agree on the meaning of the data that is sent. It would be pretty chaotic if the bank thought the first character of the packet was the type of request while the sender thought that was where to put the account number.

3.1.7. Gateways

Often systems are designed under different circumstances and build on top of different standards. To allow these different systems to interoperate, *gateways* are often used. A gateway is basically a translator. To the sender, a gateway appears to be a receiver of the matching standard. However, the gateway takes the original sender's request and reformulates it to another standard and forwards it on to a recipient. The recipient thinks that a system matching *its* standard has made the request and responds appropriately. The gateway translates the response into the form appropriate for the original sender and sends it back.

Gateways are interesting in that they require no changes to either the client or the server — they both think that they are interacting with a "like" counterpart. It is not always possible to build a gateway to translate between two standards, since the basic operations of different standards may not always have matching counterparts. However, since similar standards often perform similar functions, it is usually possible to map the majority of the capability through a gateway.

Gateway systems are often used to connect dissimilar database management systems. In computer networking, this functionality is often performed by a bridge or a router — both devices that take input in one format and output that same information in a different format. The distinction between bridges, routers, and gateways is beyond the scope of this paper.

The interested reader should consult a text on computer networking for more details.

3.2. Service Technologies

3.2.1. Making this a business

Cable operators are in the *business* of delivering services. This means that they need to deal with all aspects of running that business. All of the problems below must be addressed — whether with custom solutions or common technologies. These technologies include:

- *Access Control* — customer management, enabling and disabling services for customers

- *Security* — dealing with the issues of privacy, theft of service, theft of content, perhaps through technologies such as encryption

- *Metering & Billing* — metering is the measurement of usage, billing is using that information to bill the customer, along with all the other functions that come along with billing (accounts receivable, ...).

- *Management* — making sure the network stays up, dealing with capacity issues, solving problems, planning for expansion, etc.

All of these are "behind the scenes" activities. However, they still have serious implications for interoperability. Access Control, for example, will play a key role in interoperability. If the consumer's system doesn't have the appropriate capability to participate in the access control protocol, then it will not be interoperable.

3.3. Services

From the consumer's point of view, they are connected to services. The fact that they are using a network-connected device — whether it's a set-top or a PC, is less important. Each service does something for the user and many of the services are interconnected. The consumer doesn't necessarily distinguish the lines between services — the user wants something done and asks "the system" to do it. How it happens is only relevant to the builders of the system, not the consumer.

3.3.1. Directory

The directory provides the most basic service to the user. It provides information about "what's out there". This is where the user learns about all the other services that are available.

From a software point of view, we should be careful to distinguish between the application that presents information to the user, and the underlying service that provides the information. The user could be presented with the perspective that they are running "their portal to the wide world of cable services". In fact, they are running a local application that queries the directory service and presents what it finds to the user.

The directory will contain not only service information for presentation to the consumer, it will also contain enough information to enable the network access device to find the server for that service, and how to talk to it (the protocol).

3.3.2. Individual Applications

Each service the user can activate will have its own way of interacting with the user. However, there are a number of capabilities that will be similar for many applications. These capabilities will be supported by common technologies. The basic idea is that the problem is solved once and the different application developers can take advantage of it. This is not a requirement — each developer will have the option to build their own solutions.

From an interoperability perspective common technologies will mean service will behave in a similar fashion. For example, if each service that charged the user money asked for confirmation in a different way, the user could quickly become confused. A common (interoperable) technology to manage the user's account would solve that problem.

3.4. Building an electronic marketplace

As new and richer services become available, the cable industry will move toward creation of an electronic marketplace. Home shopping will be the likely fore-runner, but it will by no means be the only participant. There are many potential players in this arena, from the cable operators themselves to traditional retailers, electronic retailers, and even today's financial industry.

Support for the electronic marketplace will require the technologies of an electronic "trading floor". These could include:

- *Electronic transactions* — the ability for a service to ask to have a customer pay an amount. This must include audit trails, customer verification of the amount, and so on.
- *Authentication* — ways for the customer to "prove" they are who they say they are (something similar to a PIN), for the service provider to "prove" they are legitimate (and not some hacker trying to steal money), and so on. This activity has many parallels in today's credit card world.
- *Account Management* — the ability for the consumer to control their expenditures. This includes setting limits, managing sub-accounts (the kids), verifying purchases, and funds transfer.
- *Marketplace* — some way for consumers to find vendors. This could be operated by the cable operator or, perhaps, by outside companies. This could be far more than "yellow pages". These gathering places could include interactive advertising, support test drives, and provide vendor "presence" with customized identities. Different marketplaces could set up reputations for having different types of vendors, and so on. The technology to provide this "world" will instantiate the electronic marketplace.

Interoperability is absolutely critical for this environment. Today's credit card companies will likely play a role here, but existing standards are not sufficient to deal with all of the problems and opportunities of electronic transactions.

4. SHARING INFORMATION

The top level of interoperability is sharing application information. Consider the ability for the shopping application to interact with the personal finance application, automatically verifying funds from the appropriate account and allocating the purchase to the correct budget category. Or, perhaps, providing your personal likeness to the shopping application so you could see what that sweater would look like on YOU.

The problems posed by this level of information sharing are large. They will require evolution of the applications and services before interoperability can be achieved. The evolution of interoperability might start with established services that have natural synergy working together to provide direct connectivity. Once experience is gained with specialized connectivity, generalized interoperability can follow.

5. STANDARDS

This section provides a short list of some of the key standards mentioned above. This list is not all inclusive; please forgive me if I miss an important protocol or standard. I include this list mainly to highlight the fact that interoperability and standards require cooperation at many levels.

As described above, connectivity is a many-layered problem. Standards are applicable at each layer. In fact, if the layering model is done correctly, the standards at one layer will have no impact on the layers above or below. In the computer world, this is illustrated by the fact that the TCP/IP standard does not change whether it uses ethernet, token ring, or ATM to provide the packet transport.

Interoperability is achieved through standard interfaces. Standardization can be as simple as two people agreeing to do things the same way. It can also be an arduous process involving entire industries. It's an evolutionary process.

The concept of a full service network is one of the offspring of convergence. It is still in its infancy. Many companies are working hard to provide solutions that work. They will worry later about interoperability.

There are many standards being brought in from other industries. Where possible, those will be used to save time and money. However, many of the key problems are not yet solved.

The important thing to remember is that standardization *will* happen in this area. If it doesn't, the converging industries are doomed to the same result as the tower of Babel.

Key Standards for Each Layer

Modulation	QAM, VSB, QPSK, FSK, ...
Media Access	Ethernet, X.25, ATM, MPEG, proprietary, ...
Reliable Connection	TCP, SPX, ...
Computer Addressing	IP, IPX, ATM, ...
User Addressing	DNS, X.400, phone numbers, postal service,
File redirection	Windows for Workgroups, Netware, Network File Service (NFS), Vines, ...
Access control	Cable equipment vendor proprietary, network OS vendor proprietary, ...
Directory	Proprietary, X.500, ...
Database	ODBC, DBMS vendor proprietary, object database interfaces, ...
Electronic Mail	X.400, SMTP, MAPI, VIM, CMC
Transaction Processing	DCE, Tuxedo, Encina, ...
Billing and Electronic Transactions	Proprietary

6. HOW DO WE ACHIEVE INTEROPERABILITY?

- Use existing standards wherever possible (don't reinvent the wheel)
- Provide gateway capabilities where standards differ
- Provide layered software that isolates the impact of differences
- Work together in industry forums to drive toward common interfaces

7. CONCLUSION

Interoperability is not a simple matter of electrical connectivity. It must happen at all layers — from connectivity through services. Existing standards will provide a strong foundation, but much work remains to be done, especially at the services and service support layers.

Delivery of rich, interactive services is new for everyone. There is lots of experience in related areas from all of the contributing industries but nobody has experience with delivery of these new services on a large scale.

Convergence has spawned a new industry by combining the skills and products of the cable, computer, communication, and content industries. Interoperability will be required if this fledgling industry is to survive.