The ISP Column
A monthly column on all things Internet

Blurring the Lines

November 2003

Geoff Huston

Maybe its a sign of ageing, but when I look at Internet architectures these days I don't see a cleanly defined layered architecture any more, with sharp lines dividing the so-called layers of functionality. Instead when I try to map the current set of technologies into an architectural layered model I'm seeing the layers start to blur into each other.

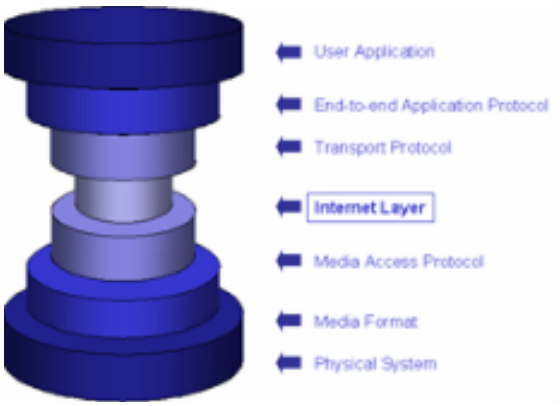
Of course the related question is whether the Internet really has an architecture at all, let alone whether it can be broken down into an ordered hierarchy of functionality. When you look at the engineering details of various ISP networks, and also look at the interactions in various Internet protocols then you observe a rich collection of interdependencies. Its tempting to conclude its an architecture-free zone, with or without layers!

Certainly much of the Internet we see today is the outcome of subsequent incremental feature creep in a relatively simple core model, rather than the outcome of a continual and deliberate process of design and implementation. There is no strict imposition of architectural standards on deployed networks, and each network designer manages to glue a selection of these components together in a huge variety of ways.

Perhaps its useful to go back to basics and look at the architecture of the Internet and then look at the various strains and stresses its been subjected to.

At the heart of the Internet architecture is the IP protocol itself. IP was originally designed as a universal adaptation layer, sitting above a variety of packet carrying media. The current suite of supported media includes all flavours of Ethernet, various types of serial circuits, and rings, and intermittent and constant connection circuits, as well as unicast, multi-drop and broadcast media. IP is also supported on packet switched services, such as Frame Relay and whatever X.25 is still alive in today's networks, as well as cell switching using ATM. It used to be a fond joke within the IETF that a method for carrying IP over carrier pigeon was defined in the IETF's RFC document series, until some Norwegians actually implemented it! The intent of IP was to able to mask the particular characteristics of any lower level data transmission service and provide a consistent end-to-end packet delivery interface to the so-called upper level service interface. The two defined next level up services are of course UDP and TCP.

This view is that of the 'hourglass' model of IP, where IP forms the thin waist of the hourglass. Below the waist are a variety of network media, while above the waist is a uniform model of the transport protocols TCP and UDP. Above these protocols sit a suite of end-to-end application protocols, such as the http protocol used for web page retrieval, or SMTP for e-mail delivery



This architecture, using IP as an adaptation layer, has been effective because of its simplicity. IP virtualizes the network layer. Applications need not be written for a particular form of transmission network, nor do applications need to worry about state changes in the underlying network media, or even the details of the networks over which the applications operate. The same application protocol will operate in an identical manner whether the packets are passed across a single local network, or whether the communication spans the globe, traversing a sequence of quite different networks on the way.

Some laudable restraint was shown in defining only a single adaptation layer in IP, rather than succumbing to the temptation of defining many such protocols. A single adaptation layer maximized the interoperability of the IP protocol, while minimizing the complexity of the service interface. And the protocol is "narrow" in the sense that it offers quite a sparse set of services. By offering an unreliable asynchronous packet delivery service, or datagram service, IP assumed a lowest common denominator of network functionality, and maximized the number of different types of networks that IP could utilize. The observation that IP not only operates on the very highest speed and lowest speed networks available today, but interoperates across these two extremes is a considerable feat. The fact that it also operates across networks no larger than a circuit board or even a chip, with nanosecond delays and at the same operates at a scale of seconds of delay, allowing packets to loop around to the moon and back is even more impressive.

Being a simple adaptation function, the task of placing IP on a new network media is not challenging. The tasks centre around specifying an IP address to media address mapping or resolution protocol, and defining a mapping from IP packets to network media frames.

This adaptation model of IP also allows composite networks to be constructed, where a packet may traverse a very diverse sequence of media. If you are reading this article from home on a DSL network, the packets containing display the text on your screen, as delivered from the server have already traversed Ethernet, ATM, serial circuits, SDH rings, TDM circuits and IP over optical circuits, and possibly a wireless system as well. The composite nature of IP implies that beyond coordination of addressing, routing and nameing no single coordinated effort is requirement, and there are inimical interdependencies between component networks, and the network-to-network interface is incredibly simple. In short, the Internet is simple to scale.

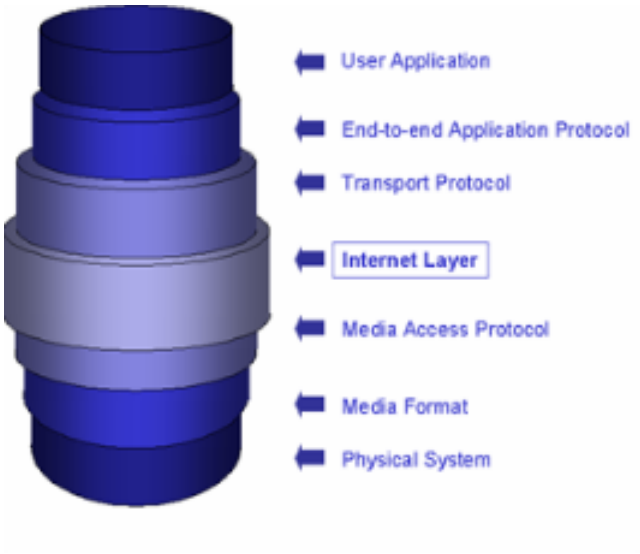
This hourglass model also makes it simple to create Internet applications. In terms of the model applications do not need to understand, or even adapt to varying transport characteristics within network components.

All of this leverage has come from using a very narrow waist in the hourglass model, that has managed to do enough, but no more than enough.

The blurring of the layers refers to the process of contortions that we are managing to apply to this hourglass model of the Internet architecture. The waist of the hourglass is being wasted. We seem to be entering the Internet's middle age, and putting on additional weight, as well as loosing some of the acute clarity of focus of its youth.

These days IP is being augmented with various forms of additional functionality. Each additional module may have some legitimacy of use, but its one more load to strap onto the waist of the architectural model, and the trim narrow waist is looking more and more like the most bloated component of the overall architecture. These additional modules

include the addition of QoS parameters to IP, where some datagrams are intended to be handled different from others. There's policy input to identify those 'special' packets and further policy input to define what the different treatment should be. Of course this, in turn, may cause IP to make further demands of the underlying mechanisms in order to implement these considerations. For example, IP never took into account the underlying considerations of link level bit error rates - a corrupted packet was discarded in the same fashion as a packet would be dropped in the face of congestion. These days with Explicit Congestion Notification, IP is making a careful distinction between these two cases. There multicast, scoped addresses, dynamic address assignment and zero context address generation and distribution being added into IP. Also we see the inclusion of additional hooks to support reliable multicast, policy- based routing. Also there's a concept that appears to borrow something from almost every layer, the approach of embedding handling directives into the packet header in a form of expressing a handling algorithm, usually termed 'active networking'.

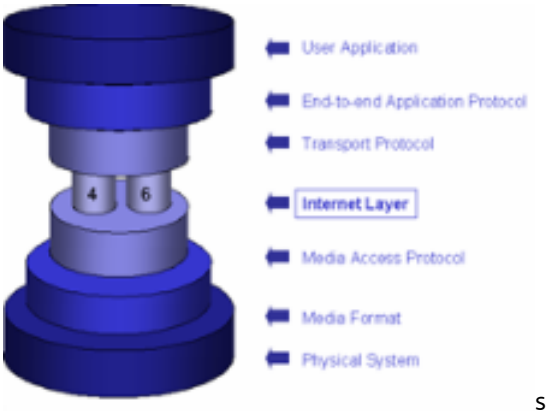


As useful and as valid as all these IP additions may be, and each appears to have a very useful domain of application, the cumulative effect is putting weight into IP. It seems that its the thin middle of the architectural model that has been the most susceptible to these fattening temptations, and the narrow waist has now headed well into the realm of obesity!

The bloat isn't only in the IP layer. It may be fine that we can operate IP over avian carriers, but what forms of 'assists' will we need to make it support a gigabit low latency connection? Maybe that's going from the sublime to the ridiculous, but we've seen various performance enhancing proxies being inserted into TCP to allow it to adapt to the particular circumstances. We've seen TCP modifications for satellite- based circuits, TCP modifications for various forms of wireless circuits, TCP modifications for very high speed circuits, and so on. here are now other forms of 'helpers' and 'agents' being added into the environment that appear in the protocol architecture as some form of end-to-end disconnection at the TCP level. It may be that the agent modifies the TCP window size advertisements, so that the window- based flow control that both ends think they are controlling is being rewritten by an agent in the middle. We've even managed to some up with a new transport protocol along the way, so that the transport layer is now populated by SCTP as well as UDP and TCP.

Similarly, there is lower level 'help' of various forms. For example, ATM cell discard behaviour often includes Early Packet Discard, with the effect that discard of a single cell may trigger discard of other cells that are associated with the same IP packet. Wireless systems can enable their own resiliency system with automatic detection and retransmission of corrupted radio transmission units. There is also the concept of partial checksums, where only a certain part of the radio data cell is 'protected' by a checksum, leaving the other part of the data unprotected. This is augmented by various layer 2 tunnelling protocols, the so-called layer 2.5 protocols of MPLS and PPPoE, LAN Emulation and Virtual LANs. It not always obvious where the "value- add" is with some of these approaches, and its often a hard call as to whether it make's IP's functions easier or harder. Introducing QoS, access control, virtual edge-to-edge circuits, and opaque switching models into the lower layers is not always appreciated at the IP level, and the interactions between the functionality at each layer can quickly become quote chaotic.

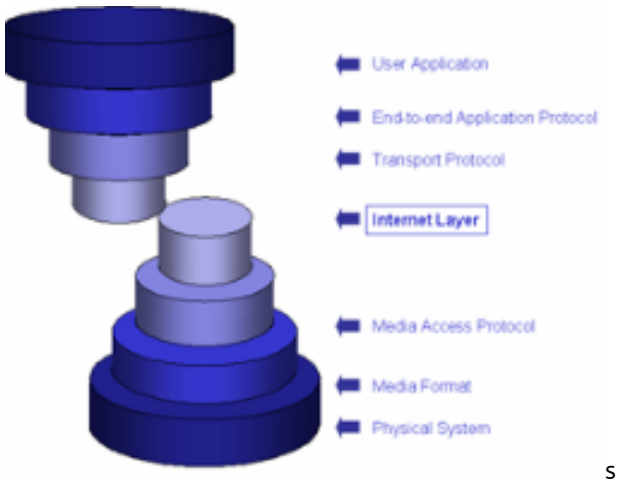
So IP is by no means a youthful character any more. Being a middle- aged entity it should really be no surprise that IP is having a mid- life identity crisis. Its now not just IP, but IPv4 and IPv6. Admittedly the two identities are not very different in terms of the functional model they require from the lower levels of the protocol stack, nor that different in terms of the functions they provide to the transport protocols, or even very different internally. Both are unreliable asynchronous packet delivery architectures using global addressing as the means of controlling stateless packet forwarding. But 'not very different' is not the same as 'identical', and right the way up into the application and beyond to the end user IPv4 and IPv6 are just different enough to be an issue for applications. Equally, when looking down, IPv4 and IPv6 make similar, but subtly different, demands of the underlying network media.



s

Its not just that this split identity doubles the number of service interfaces that are presented to the transport level, nor that it requires changes to other elements of the protocol suite above and below the IP layer. Its that is creates subtle (and some not-so- subtle) interoperability problems. One perspective here is that IPv6 is similar enough to IPv4 to be considered as equivalent in a functional sense, yet different enough for this to not be the case when you get into the details of the functional interfaces. Switching between persona in this multiple-identity IP environment often gives strange context switches. The topology of the two networks differ, so what may be blinding fast in one IP realm becomes a leisurely pace in the other due to a tourist trip twice around the globe. The web pages that are delivered in response to a URL may be different, or your mail may head in a different direction to a different destination for no apparent reason.

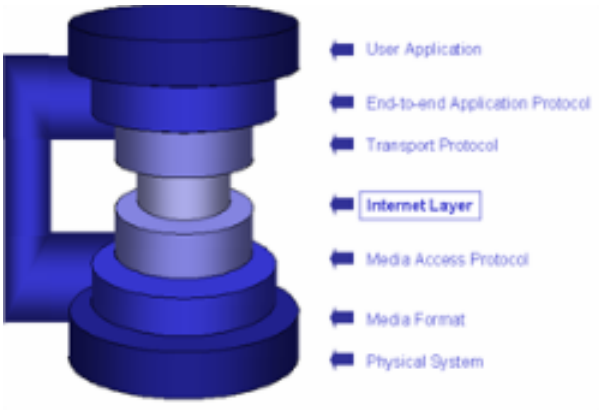
And being, notionally, the narrow part of the hourglass, we've also managed to snap it in two. Network Address Translators sit within the middle of the IP module and functions by systematically transforming IP addresses in the packet header as they pass through the unit. The address at one of a connection is not the same as the other, as applications how place these addresses into the application level data stream soon discover. Its confusing enough when we discover that the 'you' that I think I'm talking to is not the same as the 'you' that you see when you talk to yourself, but its even more confusing when this happens only when you and I are at certain locations in the network. And, by necessity a NAT is like a diode, in that connections can be initiated across a NAT from only one side of the NAT. The NAT intrinsically blocks connection attempts from the other side.



s

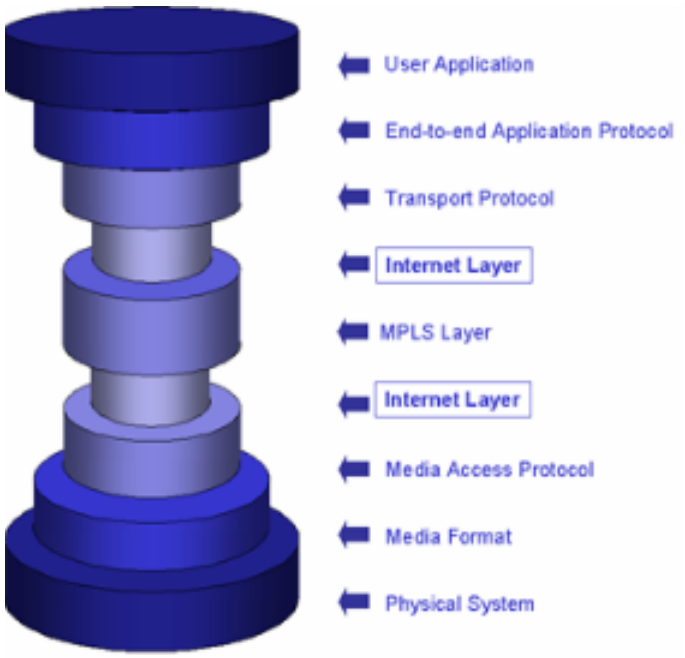
All this ruins predictability and simplicity, and pushes the problem to the application level, making applications more complex. To achieve functionality across a NAT there is now a set of restrictions placed on the upper levels of the architecture. Applications may choose to use a strict client server architecture, where the relative locations of the server is critical given that it needs to be on the "right" side of the NAT units with respect to its potential clients. Alternatively applications may choose to invent their own end-to-end identity space, and populate the Internet with 'helpers' and 'agents' placed at locations where the identifier to IP address resolution can be performed. In ether case applications can no longer assume fully transparent end to end visibility and connectivity at the IP layer, and have to perform additional functions.

Maybe we are transforming the hourglass into an hourglass with a handle.



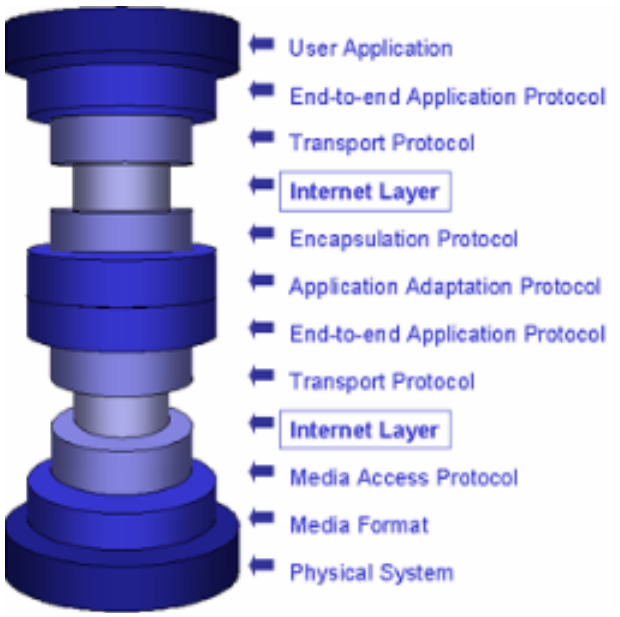
Upper layer functions are needing to establish a means of operating over specific low layer functionality. In the world of radio and mobile IP low level events devices enter or leave a radio realm, devices move between base stations in handoffs, and there are micro- movements in an ad-hoc wireless networked environment. In such situations there is often a need for higher level functions, such as access authentication, to be performed prior to IP configuration, Architecturally what is often the case is that the application needs to be able to operate in modes that are not conformant to the strict IP model. It is also the case that TCP is a general purpose transport control protocol, and the particular characteristics of some media, such as rapid bandwidth changes in the 3GPP model find problems in mapping the TCP behaviour to media behaviour. A recent approach has been in considering 'triggers', where media events are signalled directly into TCP, or even directly into higher level applications in order to allow the application to make effective use of the media.

Various forms of encapsulation and tunnelling also yield surprising results from an architectural perspective.



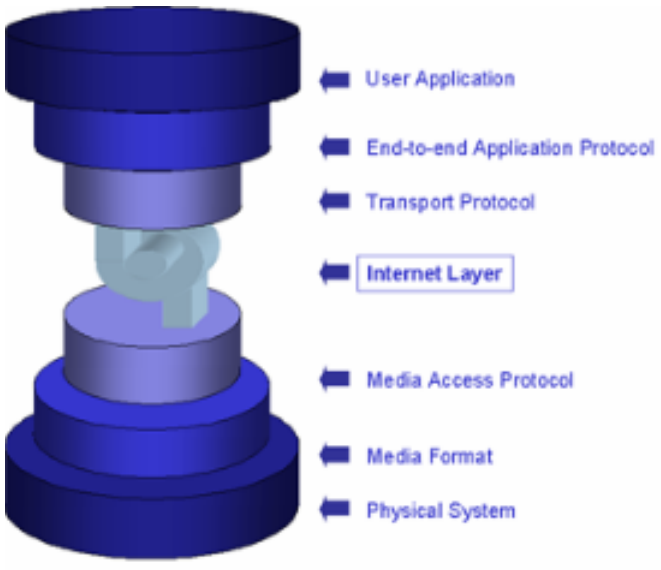
One of my favourites is using an IP network to also support the MPLS architecture to create end-to-end label switched tunnels, across which is operated ATM, which in turn is used to emulate a Frame Relay interface which is used to run, predictably, IP. While things work an overlay using encapsulation is often a powerful method to abstract yourself away from the complexities of one layer by creating a simple overlay. The problems are often exposed only when things did not operate as expected. A disruption in the network may be caused at any of the levels, and understanding the operational behaviours of such networks is no mean feat. Part of the problem here is that an outage at one level may cause upper levels to attempt an automatic repair, which, in turn causes disruptions further up the encapsulation chain and so on. There is probably some rule about the number of simultaneous encapsulations that is safe to deploy, but my guess that the number is small.

While IP over X over IP, where X is some link level protocol, is a relatively common service provider solution in the VPN market, other solutions have proliferated from the end user. Where a user sites within private address space on a VPN, and the network is visible other thought a firewall-filtered view of the world. then its not surprising to see more esoteric forms of encapsulation where IP is layered on top of the application level, as the application is about the only means of supporting a relatively coherent view to the external world. Imagine, if you will that you are behind one or more NAT units, using a private address block, and that the only function supported by the associated gateway is web access. Stuck? Possibly. Or maybe not. If you have a helper agent on the Internet side you can use of the observation that secure HTTP, or HTTPS is uncacheable, and therefore is normally passed through unaltered. This observation can be used to place IP packets inside the encrypted payload of an HTTPS session, thereby creating a tunnel between the endpoint and the HTTPS agent. The resultant architecture looks somewhat like the modified hourglass below.



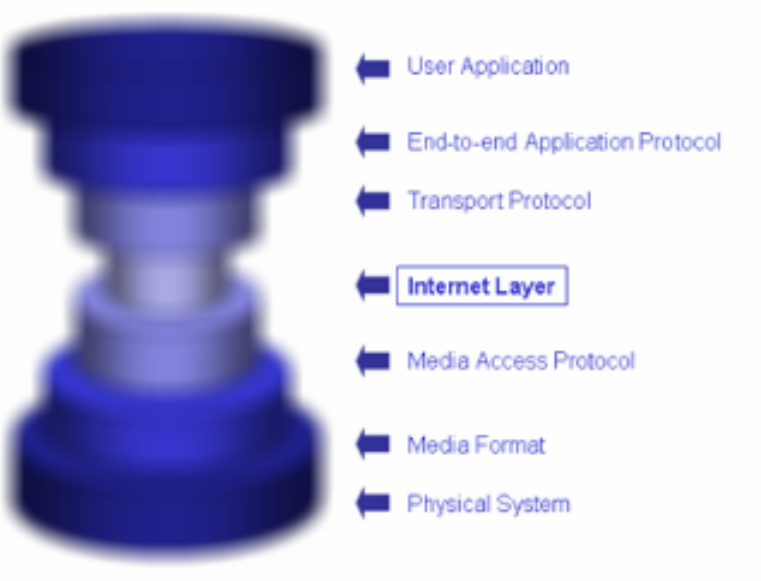
IP has been defined to work over a wide variety of applications, including most recently XML, and it is perhaps an instance of the generalization that in a layered architecture any layer can be seen to function more or less identically to any other layer. And this goes with the related observation that the practice of attempting to solve a problem simply by adding a layer of recursion is a fine example of the practice of computer science.

Despite all this IP is still remarkably supple. IP in IP solutions are one example of the use of IP to support various forms of VPN architectures and provider-selection functions for public IP service offerings, while still retaining much of the benefit of the thin adaptation function of IP. Such IP-in-IP approaches also offer a set of solutions to mobility-based issues, where discrete IP headers are used to differentiate between a mobile device's current location and its IP identity.



What is going on here? Are we seeing the normal process of decay that besets all large engineered solutions over time? Is this just entropy at work? Or is this evolution at work, where the results of the process of market-based selection define the direction of further changes to the IP model. While the results of such an evolutionary process are undesigned and perhaps unpredictable, that does not necessarily imply that they are necessarily inferior. Indeed adaptability is a key component or survival. A system that is engineered to be adaptable makes minimal demands on others in order to reduce the imposition of complex interdependencies, while at the same time creates outcomes that are valued in the environment.

So what we are looking for in IP is simplicity and modularity as a constant attribute of IP while other aspects may change over time. If you design a protocol to permit subsequent change its often the case that you design it differently than if you are after some static concept of design excellence.



Yes, some of the layers are blurring, but the layered model of protocol architecture was just that - a model. The essential attribute of the layered model was one way to describe the modularity of the components of the system and their interaction. Yes, the layers in this model are blurring as the shape of the IP architecture changes. The challenge we have at present is to come up with an evolved model that accurately captures the current IP architecture, and once again places sharp focussed lines around the component modules in order to prevent the intrusion of complex interdependencies. That way we can ensure that the evolutionary processes at work here are ones that create scaleable, simple and effective outcomes in the Internet.

This month's column was inspired by Steve Deering's inspiring presentation at the August 2001 Plenary Session of the IETF on "Waist Watching in IP". Thanks Steve.

Disclaimer

The above views do not represent the views of the Internet Society, nor do they represent the views of the author's employer, the Telstra Corporation. They were possibly the opinions of the author at the time of writing this article, but things always change, including the author's opinions!

About the Author

GEOFF HUSTON holds a B.Sc. and a M.Sc. from the Australian National University. He has been closely involved with the development of the Internet for the past decade, particularly within Australia, where he was responsible for the initial build of the Internet within the Australian academic and research sector. Huston is currently the Chief Scientist in the Internet area for Telstra. He is also the Executive Director of the Internet Architecture Board, and is a member of the APNIC Executive Committee. He was an inaugural Trustee of the Internet Society, and served as Secretary of the Board of Trustees from 1993 until 2001, with a term of service as chair of the Board of Trustees in 1999 and 2000. He is author of a number of Internet-related books.