The IEEE P1520 Standards Initiative for Programmable Network Interfaces

Jit Biswas, Kent Ridge Digital Labs
Aurel A. Lazar, Jean-François Huard, and Koonseng Lim, Xbind Inc.
Semir Mahjoub, Ericsson Infotech AB
Louis-Francois Pau, Ericsson Utvecklings AB
Masaaki Suzuki, C&C Research Laboratories, NEC USA, Inc.
Soren Torstensson, Ericsson Infotech AB
Weiguo Wang, Kent Ridge Digital Labs
Stephen Weinstein, C&C Research Laboratories, NEC USA, Inc.

ABSTRACT This article discusses the need for standard software interfaces for programming of networks, specifically for service and signaling control, through programming interfaces. The objective is to enable the development of open signaling, control, and management applications as well as higher-level multimedia services on networks. The scope of this effort includes ATM switches, circuit switches, IP routers, and hybrid switches such as those that provide for fast switching of IP packets over an ATM backbone. The basic ideas represented herein are in the process of development as a standard for application programming interfaces for networks under IEEE Standards Project IEEE P1520.

An international research and industry community known as OPENSIG [1] was formed in 1995, advancing the concept of open signaling and network programmability. Recently, several companies and laboratories started a new IEEE standards development project, IEEE P1520 [2]. This project envisions tomorrow's telecommunications network as a giant computer — a fully programmable machine — that delivers advanced voice, data, and video services globally. In the present paradigm, the key intelligence of the network, which lies in the signaling network, is built with a few fixed algorithms or programs known as standard signaling protocols and control programs. Development of richer signaling protocols and control programs has been a slow and arduous process. This is because the signaling standards for the modern telecommunications industry have become very complex and require consensus from all manufacturers and operators of switching equipment.

The alternative proposed is to provide a set of software abstractions of network resources which allow distributed access to low-level control capabilities of network devices. At higher levels there should be a progression of capabilities to express and exploit control intelligence. Thus, the earlier paradigm of fixed signaling protocols and control programs is now replaced with programmable and extensible signaling protocols, gateways between signaling systems, and meaningful interfaces to earlier systems. P1520 is developing a reference model separating end-user application semantics, value-added services, network-generic services, virtual network devices, and hardware and/or low-level software support. The project aims to establish programming interfaces between the levels of the reference model. The proposed standard specifies, in industry-standard Interface Definition Language (IDL) [3], a set of programming interfaces for distributed access to switching functionalities by service control entities, including but not restricted to signaling services entities. There is also a supporting interface component of the proposed standard. It specifies, in the form of a set of messages, the interfaces a switching unit must implement in order to provide minimal standard services supporting higher-level software. The types of switching units to be considered in this standard shall include asynchronous transfer mode (ATM) switches, IP routers, and circuit switches currently running Signaling System No. 7 (SS7). The standard lends itself to easy programming of quality of service (QoS) functionality through appropriate interfaces.

The capability to program the network has profound implications. In addition to opening up the marketplace to new and innovative control structures, not requiring a long and painful standardization process, there is a departure from the traditional signaling standards development model. In the traditional model service entities are defined in multiple layers, with functions at the same layer communicating peer to peer across a network. The exact semantics and syntax of the interaction among peer entities are standardized. This ensures that a *vertical interface* is well defined, so two devices of different vendors can communicate with each other as long as they conform to the same vertical interface (Fig. 1a). This is seen in both the intelligent network (IN) architecture as well as proposals that have emerged from broadband integrated services digital network (B-ISDN) signaling. Examples are traditional call routing, call setup, proposals for leaf-initiated join (LIJ) of multicast applications, and dynamic QoS control and management. This approach of hard-coding algorithms into standards inhibits programmability and extensibility. As signaling functionality increases, it is harder and harder for a consensus to be reached in the standardization process. There are simply many ways of doing the same thing, and different network conditions require different solutions. Fixing on a particular algorithm will lead to suboptimal solutions.

A different approach is possible — one that is not new. For example, an operating system in a personal computer gives the abstraction of a virtual machine to the programmer, thereby providing programming interfaces to the physical resources of the computer. Open and standard interfaces

0163-6804/98/$10.00 © 1998 IEEE

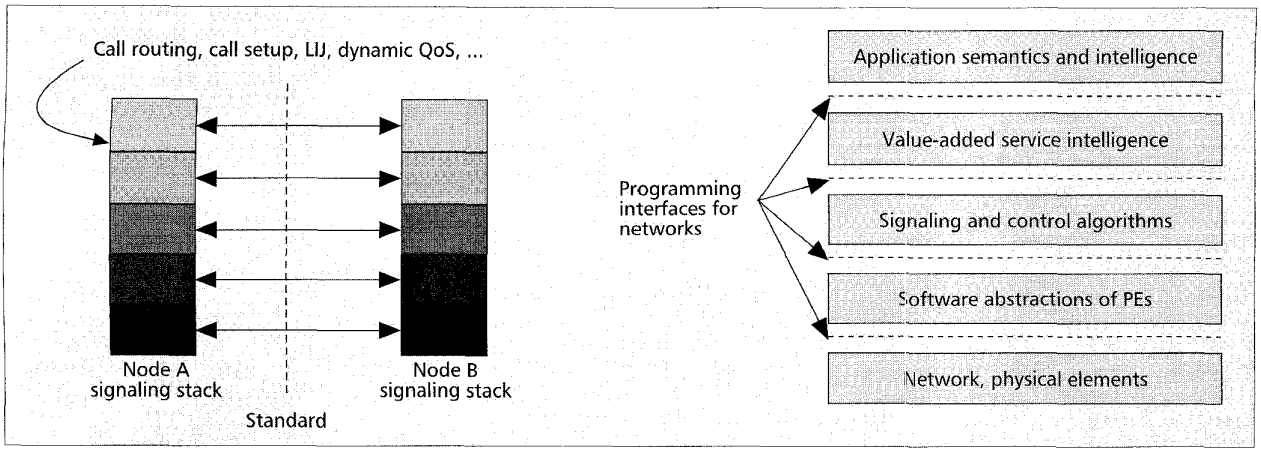


Figure 1. *(a) The traditional approach to the development of signaling applications; (b) the proposed approach, providing programmable interfaces.*

between the hardware and operating system, and between the operating system and applications, have allowed the marketplace for peripherals and applications to develop in myriad ways. Adopting a similar approach in the domain of network management has led to standardized management information and protocols for access to such information. Management applications, however, have not been standardized, and have been developed by third parties in innovative ways.

The advantages of opening up application programming interfaces (APIs) for networks (Fig. 1b) are many. First, there are the benefits associated with leveraging distributed object-oriented technology and modeling (see "Distributed Object Technology for Networking," this issue). These include benefits of object-oriented software engineering such as modularity, reusability, scalability, and reliability, which go a long way in reducing the service deployment cycle. Second, there are benefits of distributed computing, such as location-transparent remote access and dynamic binding, which make it possible for third-party service developers to write applications that perform third-party call setup and management. Third, incorporating programming interfaces in networks for developing control and management applications allows unprecedented separation of software and hardware. This, in turn, ensures that the end user gets the full benefit of competition in the marketplace. Fourth, there is now a separation of the signaling business from the transport business, permitting rich and flexible ways of dividing and segmenting the market. Finally, through gateways one can ensure that legacy interoperability will always be kept in the forefront.

We believe that a combination of the models of ATM and the Internet may be best suited for the backbone of the multimedia network of tomorrow, enhancing, and in some cases even replacing, existing signaling mechanisms with the use of parallel IP and ATM protocol stacks for information transfers within the same session. In such an infrastructure, it is best to use the IP stack with its fairly inexpensive, albeit unreliable, UDP for interobject communication (which we term *signaling*), and the ATM/ATM adaptation layer (AAL) stack for media stream transport with guaranteed QoS.

Considering the installed base of the telecommunications infrastructure, SS7 is a well-accepted standard for network signaling in circuit-switched networks, and is being implemented for ATM networks. However, since SS7 was designed to meet the signaling requirements of the past, it does not completely satisfy the requirements of today's cell- and packet-switched networks. Consequently, new and more suitable architectures (e.g., based on distributed object technology) need to be developed. It must, though, be ensured that the enormous installed base of SS7 based networks is smoothly migrated toward these new architectures. Interfaces between SS7 and new "signaling" systems (based on object-oriented technology) have to be developed. By "opening up" legacy signaling architectures (including SS7), new software markets and markets for value-added service providers will emerge.

In the following section we present the P1520 reference model. We then look at programming interfaces for networks of ATM switches, circuit switches, and IP routers/switches, respectively. We outline the initial direction being taken in the P1520 standards project, and present related standards and implementation agreements with which P1520 has liaison.

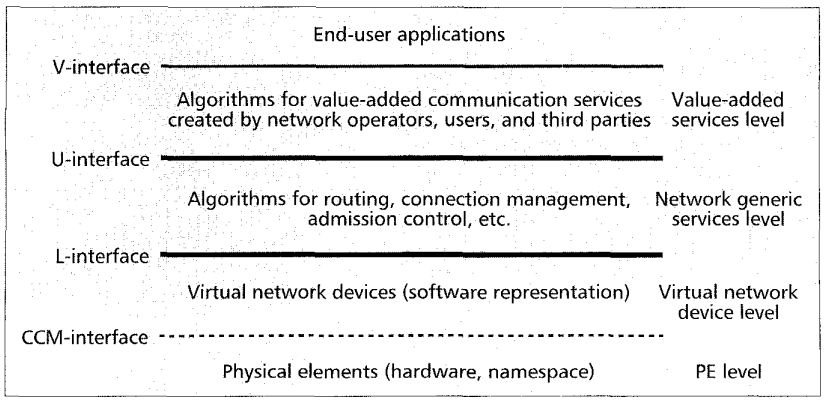


Figure 2. *The P1520 reference model.*

THE P1520 REFERENCE MODEL

Figure 2 illustrates different levels of the P1520 Reference Model for APIs for networks. In this model, there are *levels*, *entities* at each level, and *interfaces* between levels. At the value-added services level (VASL) the entities are end-to-end algorithms that add value to services provided by the third and lower levels by means of

user-oriented features and capabilities, such as real-time stream management, synchronization of multimedia streams, and other capabilities beneficial to value-added service providers and end users. At the network-generic services level (NGSL) the entities are algorithms that deal primarily with the functioning of the network. For example, the algorithms may be virtual connection/virtual path (VC/VP) configuration algorithms, or routing algorithms. This level also contains algorithms that have a global view of subnetworks of narrowband circuit switches through their individual local interfaces at the lower level. Thus, the interface to this level includes distributed object interfaces to service control points (SCPs) for use in IN applications using INAP. Standard means of carrying out such distributed algorithms are currently being investigated for standardization by the Object Management Group (OMG) Telecommunications Domain Task Force (TelDTF). Separate control entities associated with different virtual private networks (VPNs) may also exist at this level.

At the virtual network device level (VNDL), the entities are logical representations (objects) of certain state variables of these entities in the first level. The above encompasses software interfaces that are abstractions of resources of the physical elements in the PE level. An example of this is the binding interface base (BIB), which is an abstraction of resources of an ATM switch, specifically the VC/VP name-space and capacity region resources (see "Open Signaling: An Overview" and "The XBIND Implementation," this issue). This has been proposed for standardization within the P1520 ATM Sub Working Group.

Finally, as the name suggests, the entities at the PE level are physical elements of the network, such as ATM and IP switches, and local exchanges in narrowband circuit-switched telephone networks. The above encompasses hardware such as ATM switches, time-division multiplexers (TDMs), VP switches, and cross-connects; and also the physical elements of a narrowband circuit-switched telephone network. These elements can be accessed by means of control protocols such as General Switch Management Protocol (GSMP) [4] or other proprietary means for accessing information at this level. In this model four types of interfaces have been identified:

- The V interface provides access to the value-added services level.
- The U interface exposes functionality of the NGSL to the VASL.
- The L interface exposes functionality of software abstractions of the VNDL to the NGSL.
- Lastly, open protocols to access the state of physical elements are collectively called the connection control and management (CCM) interface.

The V interface provides a rich set of APIs to write highly personalized end-user software, often in the form of value-added services. These services generally do not deal directly with the communications process per se, but provide convenient features that enhance the value and experience of using the basic communication services. Many of today's advanced IN (AIN) services fall into this category, and are considered to be the precursor of things to come. Value-added services may be realized as "virtual service" objects invoked from Web browsers or other client applications.

The U interface deals with generic network services and allows its user to make requests for connections. These connections may take the form of simple point-to-point connections, point-to-multipoint trees, or any general graph (as in the case of a VPN). The power of this interface comes from its generality, which in essence allows parameterization of connection setup requests independent of the algorithm used in the connection setup procedure. This separation between interface and implementation in principle would allow multiple connection management schemes to coexist in a single network. The L interface defines the API to directly access and manipulate local network resource states. These states could, for example, take the form of VC/VP lookup tables in the case of ATM networks or routing tables controlling IP forwarding in the case of IP routers. Finally, the CCM interface, indicated by a broken line in Fig. 2, is not a programming interface, but a collection of protocols that enable the exchange of state and control information at a very low level between a switch and an external agent.

The reference model described above provides a high-level framework for positioning programming interfaces for networks. It is necessary to map this high-level model into existing networking technologies to be able to recognize the points at which "useful" programming interfaces may be obtained. In the following three sections, we consider three types of networks, namely, networks of ATM switches, circuit-switched networks, and IP routed/switched networks; and consider how the above reference model applies to each type of network. This presents an initial direction in which the work of standardization of open interfaces is carried out within the framework of each type of network. We then present the initial directions of the P1520 standards project.

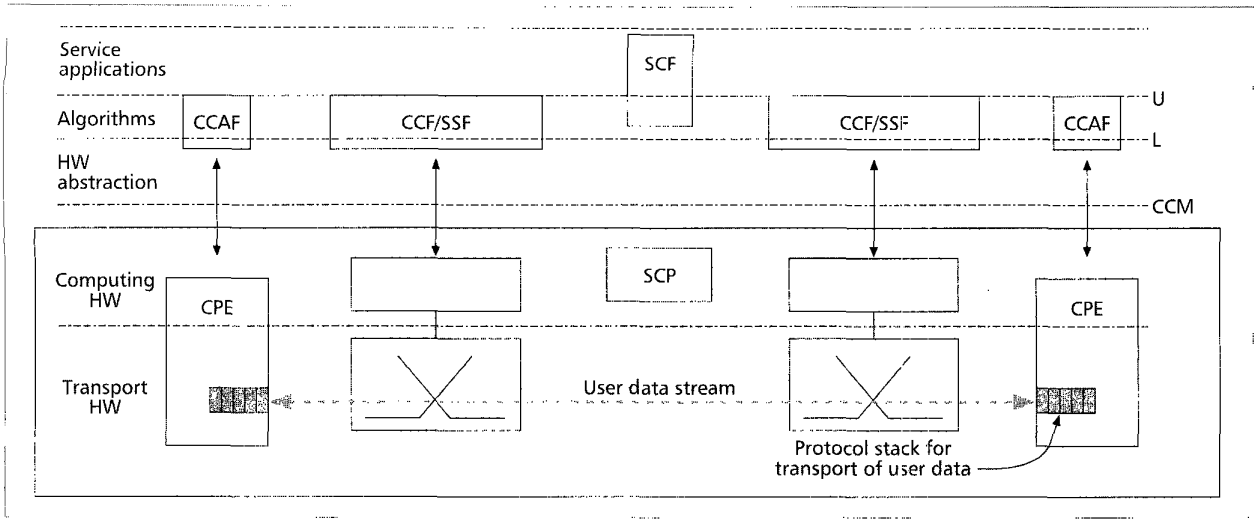
Programming Interfaces for ATM Networks

The connection-oriented nature of ATM switching allows, in principle, a fairly straightforward way to implement multimedia networking services. The primary challenge in defining programming interfaces for ATM networks lies in the task of developing generic QoS abstractions that capture the notion of capacity and allow specification of QoS requirements and constraints. These abstractions represent a subset of the network resources as defined in the L-interface of the P1520 reference model, and make the ATM network programmable in the sense that any signaling and control procedure can be specified in terms of a sequence of remote operations over these abstractions. As such, a set of open distributed APIs need to be specified as the L-interface in the P1520 reference model. An example of such an interface is the BIB [5]. BIB interfaces fall into two broad categories: APIs that provide the abstraction of fundamental networking resources, and APIs to support the creation and subsequent control and management of end-to-end multimedia streams.

In the first category of interfaces, two fundamental forms of resources are identified: name space and bandwidth. Name space resources represent addressing constraints, while bandwidth resources characterize buffer limitations. In the case of ATM networks, name space resources manifest themselves as the virtual path and virtual circuit identifiers (VPI, VCI) of an ATM channel, while bandwidth resources are manifested as the buffer sizes at the multiplexing points of an ATM cross-connect.

Similarly, interfaces in the second category can be further subdivided into sub-APIs that support:

- Control of generic multimedia devices
- Control of general ATM switch elements, including the switch fabric and output multiplexers
- Control and management of formatted multimedia flows
- Control of end-to-end transport protocol elements



■ **Figure 3.** *Mapping of the P1520 reference model to the SS7 architecture. HW: hardware.*

Multimedia device control refers to the APIs for enabling the setup and subsequent changing of multimedia device parameters that affect the generation or consumption of multimedia data. These typically include control of sampling parameters, including rate, format, encoding scheme, and so on. The control of general ATM switch elements refers to the task of controlling the allocation of name space and bandwidth resources of switches. These are typically related to the manipulation of VC/VP lookup tables in an ATM switch fabric and the control of scheduling policies at the output ports (assuming an output buffered switch model).

The control and management of formatted multimedia flows refer to the task of providing VCR-like controls on top of an encoded multimedia stream. Typically, operations here involve pausing and resuming the stream, changing the temporal direction and speed of playout, and direct access to indexed points in a sequentially recorded stream. The control of end-to-end transport elements refers to the task of selecting and instantiating an appropriate transport protocol stack in accordance with QoS requirements, and the subsequent tracking and monitoring of the level of QoS achieved. Interfaces at this level are extremely important because they provide the only means to assess the end-to-end quality of a stream in a connection.

The implementation of the L-interface in ATM networks will in some cases require direct access to hardware resources, especially in the case of network element resources. As such, there have been three proposals for defining a separate generic vendor-neutral interface for remote control and monitoring of ATM switches. Two of the proposals are actually extensions to [4] with added support for QoS parameters. All three proposed APIs specify a bit-level protocol exchange between a switch controller and a remote switch for connection setup of unicast and multicast trees, cell-level statistics monitoring, and alarm reporting.

In summary, control and management of the switches may be carried out by external agents that use the CCM interfaces. In the P1520 architecture, the external agent consists of a software representation of the switch, which presents a programmable interface, the L interface, at the VNDL. Algorithms at the NGSL manipulate the local states of switches by means of the L interface. Entities in the VASL make requests of the NGSL algorithms via parameterized requests at the U interface. The NGSL algorithms then translate the parameters of each request into individual steps carried out on the local states of switches by means of L interfaces. Thus, for example, a value-added service provider providing a QoS-controlled service at the VASL may make a request for a point-to-multipoint connection setup request to a connection manager at the NGSL. The connection manager might then utilize a routing algorithm to obtain a preferred route or path for the communication, and then make a tentative route booking at each switch in the path via the L interface for that switch. This is illustrated in greater detail by means of an example in the white paper which is available at the P1520 Web site [2].

Programming Interfaces for SS7-Based Networks

A number of activities around the world are investigating issues related to the development of an open (signaling- and technology-transparent) network architecture that enables creation of distributed multimedia and IN-oriented services.

The Telecommunications Information Networking Architecture (TINA) is probably one of the most widely known conceptual architectures in this context. The TINA Consortium (TINA-C) [6] approach is to separate *service* and *delivery*, to integrate control (signaling) and management, and to adopt distributed object technology for the creation of a generic computing and communication platform. This platform is defined as the distributed processing environment (DPE); it relies on the kernel transport network (kTN), which encapsulates network signaling. TINA also identifies several reference points (RPs) that constitute sets of application- or DPE-related interfaces. The number of currently specified RPs is, however, reduced,[1] and DPE-related RPs are not described in the current work. P1520 has identified a number of interfaces that in some cases correspond to TINA application-related RPs, but in general are on a different level.

In contrast to TINA-C, which is defining a complete new architecture, others are aiming at the creation of high-level object-oriented programming interfaces toward "traditional" IN functions. One of these initiatives is the OMG TelDTF which is defining a Common Object Request Broker Architecture (CORBA)-based IN service creation environment. A

[1] *Currently, TINA-C has specified two application-related RPs, the Consumer/Retailer and the Retailer/Connectivity Provider RPs.*

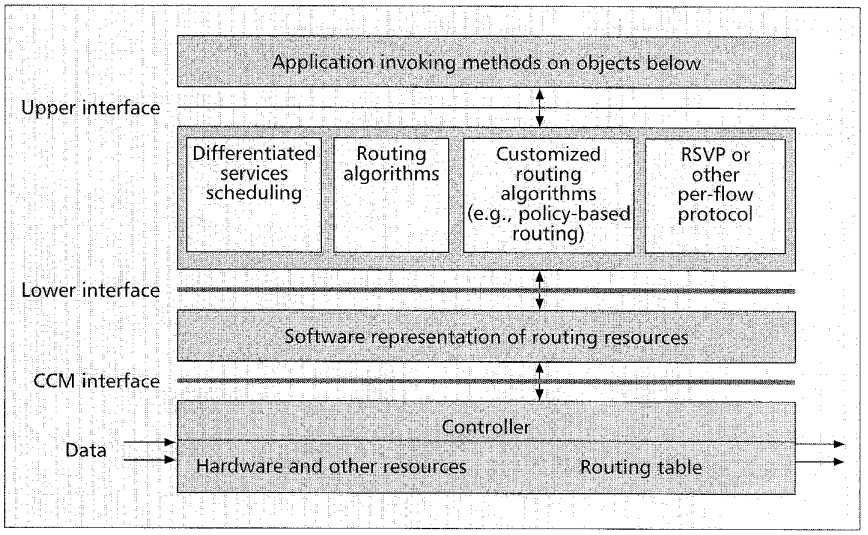


Figure 4. *Mapping of the P1520 architecture to IP routers/switches.*

couple of proposals dealing with interworking between CORBA and IN have been submitted. Another initiative is Java AIN (JAIN), which is driven by Sun Microsystems in collaboration with leading SS7 stack providers. JAIN offers Java-defined APIs toward the SS7 functions. The architecture takes advantage of Java's multiplatform capabilities, which let developers write an application that can run on a variety of platforms.

Thus, the industry is developing interfaces close to services and applications. These interfaces have more in common with the U interface of the P1520 reference model.

Figure 3 is an attempt to map the P1520 interfaces to the SS7 architecture. Hardware consisting of service switching points (SSPs) and SCPs forms the basis of the physical level in this architecture. The hypothetical placement of the interfaces of the P1520 reference model is shown as dashed lines in Fig. 3. It has been concluded that for circuit-switched networks there is little value in opening up the interface to the hardware since the QoS abstraction for a physical circuit does not provide added value. The QoS provided by a circuit switch is well defined and invariable.

A smooth evolution must take place in order to migrate from old signaling architectures (e.g., SS7) toward new object-oriented ones. The SS7 Sub Working Group of P1520 will investigate the ongoing activities within the industry regarding network programming interfaces to get a clear picture of the state of the art. It will also be necessary to analyze "problem areas" in order to identify *real needs* for standard U-level programming interfaces for network services. Based on the findings and if there is a consensus on it, a standard will be developed.

Such an interface may include functions that are application-specific (distributed application API) and/or more general functions (e.g., signaling transport API). It is also believed that such an interface can be made more technology-independent, and be usable not only for SS7-based circuit-switched networks but also for IP networks and ATM networks.

Programming Interfaces for Networks of IP Routers/Switches

The P1520 reference model maps to IP routers/switches (Fig. 4), in a manner very similar to ATM switches. A GSMP-like interface is desirable at the CCM interface. Such an interface would be capable of triggering events in a routine way to notify higher-level algorithms of the occurrence of certain types of events such as flow identification and other exceptional conditions. As may be expected, the issue of polling versus interrupts becomes an important performance criterion for the management of these events, so the interrupt (notification) capability of a distributed object system becomes significant.

The programmable interface for a router supports traffic control needs such as dynamic adjustment of resource allocations among differentiated service classes, alternative routing algorithms, and customized treatments for packets associated with a user, destination, or application, described further below. As multimedia communication becomes increasingly prevalent, it becomes important to deal with special conditions of multimedia traffic and provide for these special conditions within the architecture of an IP router/switch and its management and control.

The P1520 guiding principle is to dissociate the maintenance of state information from the algorithms that manipulate state in a network. This is the same guiding principle that applies to determining appropriate programming interfaces for IP routers/switches. Routing table lookup and manipulation may be an important ingredient of several classes of algorithms which operate at the NGSL; for example, policy-based routing, differentiated services scheduling, Resource Reservation Protocol (RSVP) or other flow-based protocols, and so on. These algorithms, and others that may be proposed in future would benefit from programming interfaces at the CCM and L levels for IP routers/switches.

As a rule, IP routers are designed to do fast routing of a large number of packets without the selective communication session control that could be done through programmable interfaces of the kind envisioned by the P1520 Working Group. In particular, the IP community believes it is impractical to maintain per-flow state in large networks. The thrust of this community is currently toward supporting a very few differentiated services [7], with different treatments for aggregations of traffic in each service class. One research direction is to use Multi-Protocol Label Switching (MPLS) [8] in a trunk dedicated to a particular service class, and the RSVP [9] for setting up and maintaining these trunks. It would seem, at first glance, that P1520 interfaces are neither necessary nor desired by router manufacturers and users. A closer examination reveals, however, that such interfaces may be necessary for IP routers and switches.

Even though programmable interfaces are more associated with ATM network control than IP network control, the need for dynamic modification of policies and configurations within IP routers is likely to emerge in the near future. The applications could include:

- Adjustment of resource allocation, such as output port rates, per differentiated service class
- Traffic grooming by source or destination address
- Configuration of line cards
- Facilitating active network policies

In addition, service integration of the Internet and public networks is likely to occur more expeditiously and efficiently if IP and ATM networks share identical or similar programming interfaces. IP plain old telephony service (POTS) (public network) telephony interworking, for example, will be

facilitated even without per-flow QoS control in the IP portion if routers can be programmed with policy-based routing and service class treatments. Learning from the ATM model indicates that P1520 interfaces for IP routers and switches should provide:

- A GSMP equivalent protocol for opening the hardware and low-level software and exposing their functionalities to the resources control elements of a router
- An L interface exposed by the control and management elements in the upper section of Fig. 4

Toward a Standard for Open Programming Interfaces

We have looked at three technologies, namely, ATM networks, circuit-switched networks based on SS7, and networks of IP routers/switches. These technologies differ, not only in their technical content and orientation, as noted in the previous sections, but also in the extent to which they have penetrated the market. ATM is a technology that has found its foothold in the backbone networks and trunks; IP is a technology that is growing rapidly in its deployment for various types of networks from data to voice and even wireless; and SS7 is a technology that is well entrenched in the telecommunications marketplace, mainly for telephony applications.

What can be said about these three technologies is that they are technically very divergent, and that they are likely to coexist in the foreseeable future. Thus, none of them can be ignored. Given this diversity in technical content and market orientation it is not our intent to realize a unified mapping for all three technologies as an immediate goal, or even to allow interoperation between the three at this stage.

From the perspective of open programming interfaces, the first priority at this stage is to reengineer each of these technologies separately, in such a way as to naturally expose points at which such interfaces may be beneficial. The signaling protocols for handling multimedia traffic on the Internet, for example, must be reexamined from the point of view of QoS guarantees, as indicated above. If programming interfaces in these protocol stacks are desired, these should then be specified clearly. A similar approach would apply to ATM networks and SS7 networks. Thus, in our approach we would start where we are with each technology, get into the details, and propose interfaces in line with those motivated by the reference model. Once these interfaces have been defined, specified, and implemented and used, we shall be in a better position to integrate or interoperate between the different technologies.

With this in mind, the P1520 standards initiative was started with the IEEE Standards Board's approval. The title of the proposed standard is "Application Programming Interfaces for Networks."

Relations with Other Standards and Implementation Agreements

Other standards and implementation agreements that are relevant to this project include ATM Forum UNI, PNNI, MPOA, and IETF RSVP, GSMP, "Integrated Services," and MPLS. The proposed standard defines part of the software environment for programmable networks. As such, existing and evolving standards (e.g., for ATM signaling or IP-related signaling) can be implemented within the standard framework as special cases.

Since the OMG Telecommunications Domain Task Force's current work on IIOP/SS7 for IN is addressing implementing IN over a CORBA platform, this project shall form close liaison with the above activity at OMG. Similarly, the work of the ANSI T1S1 subcommittee on Services Architectures and Signaling [10], has similarities to this project, and close liaison will be maintained with that group as well, and other related groups such as ETSI and QSIG. Liaison shall also be maintained with ITU-T Study Group 11, which is responsible for studies relating to signaling requirements and protocols for telecommunications. The work done in this project will complement the protocols and standards already being worked on within ANSI T1S1 and ITU-T SG11 and other bodies, by making it possible for these protocols and specifications to be programmed in an open manner, by means of open programming interfaces and control entities.

TINA-C [6] has developed a modular open software architecture for the provision of telecommunications and information services, encompassing both management and control. Although programming of control functions has not been the focus of that initiative, the interfaces defined by TINA-C will provide precedents and experience useful in the specification of interfaces for this proposed standard.

Summary of the P1520 Standard Project

In summary, the IEEE P1520 standard project aims to establish an open architecture in network control, and the interface between network control and management functions. In this approach a reference model is defined which separates control intelligence from control mechanisms. The control mechanisms reside on top of the hardware or physical element level of the network. Within the initial timeframe of two years from December 1997, the P1520 project shall establish for ATM switches a programming interface, namely the L interface, and a supporting interface, the CCM interface, also for ATM switches. Thus, the immediate scope of the deliverables of the P1520 project is restricted to L and CCM interfaces, although U interfaces are be examined for SS7 networks. Appropriate L and CCM interfaces for IP routers and switches are also currently under examination.

References

[1] OPENSIG, http://comet.columbia.edu/opensig

[2] P1520 Web site: http://www.ieee-pin.org; see also http://stdsbbs.ieee.org/groups/index.html

[3] ISO/IEC JTC1/SC21 WG7 N14750, "ISO/OMG IDL — Interface Definition Language."

[4] Ipsilon Networks, "GSMP: General Switch Management Protocol," IETF RFC 1987.

[5] A. A. Lazar, K. S. Lim, and F. Marconcini, "Realizing a Foundation for Programmability of ATM Networks with the Binding Architecture," *IEEE JSAC*, Special Issue on Distributed Multimedia Systems, vol. 14, no. 7, Sept. 1996.

[6] TINA-C, http://www.tinac.com

[7] IETF, "Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)," Internet draft draft-li-paste-oo.txt, Jan. 1998.

[8] IETF, "A Proposed Architecture for MPLS," Internet draftdraft-ietf-mpls-arch-00.txt, Aug., 1997.

[9] IETF, "Resource Reservation Protocol (RSVP) — Version 1 Functional Specification," RFC 22054, Sept. 1997.

[10] T1S1, ANSI T1S1 subcommittee on Services, Architectures and Signaling; docs. include T1.110, T1.111 (on MTP), T1.112 (on SCCP), T1.113 (ISDN-UP), T1.114 (on TC), T1.116 (on OMAP), T1.118 (on ISNI), and other documents in the T1.6xx series concerning BISUP, DSS2, additional traffic parameters, network call correlation, security, multipoint call control, and BISDN NNI; see http://www.t1.org

Additional Reading

[1] A. A. Lazar, "Programming Telecommunication Networks," *IEEE Network*, Sept./Oct. 1997, pp. 2–12.

Biographies

JIT BISWAS (biswas@krdl.org.sg) is a senior member, research staff at the Kent Ridge Digital Labs (KRDL), Singapore. His areas of work include distributed computing and computer networking and telecommunications. Previously he has worked in network management in a collaboration with ITRI, Taiwan, and also provided knowhow and expertise in the management of services and networks for the national high-speed network research testbed in Singapore. His current work is in the area of standardization of programming interfaces for networks. He has a Bachelor's degree in electrical and electronics engineering from Birla Institute of Technology and Science, India, a Master's degree in computer science from Southern Methodist University, and a Ph.D. in computer science from the University of Texas at Austin.

AUREL A. LAZAR [F '93] (http://comet.columbia.edu/~aurel) has been a professor of electrical engineering at Columbia University since 1988. His theoretical research focuses on networking games and pricing. His experimental work focuses on building open programmable networks that support the rapid creation, deployment, and management of networked multimedia services. This work led to the establishment of the IEEE standards working group on Programming Interfaces for Networks (http://www.ieee-pin.org).

JEAN-FRANÇOIS HUARD (http://www.xbind.com/~jfhuard) received the B.Eng (EE) in 1990 from Ecole Polytechnique de Montreal, the M.A.Sc. (EE) in 1992 from Concordia University (Montreal), and the M.Phil. in 1994 from Columbia University, New York. He his currently a Ph.D. candidate in the Department of Electrical Engineering at Columbia University. His current research interests are in the area of open middleware transport architecture and high-performance QoS-aware transport protocols. He is leader of the Transport Group of Xbind, Inc. He is also technical editor for the IEEE P1520 standard initiative and chair of the IEEE P1520 ATM SWG (http://www.ieee-pin.org).

KOONSENG LIM (koonseng@xbind.com) received a B.Sc (Hons) from the National University of Singapore in 1991 and an M.S. (EE) from Columbia University in 1997. He was the recipient of the Halbretch book prize for his undergraduate dissertation on the performance analysis of FDDI networks in 1991. In 1994 he was awarded a full scholarship to pursue a Ph.D at Columbia University by National University of Singapore. He served on the program committee and organizing committees of IEEE OPENARCH '98 and IFIP IWQoS '97, respectively, and is a contributor to the ISO MPEG-4 DMIF and IEEE P1520 standards working groups.

SEMIR MAHJOUB (semir.mahjoub@ein.ericsson.se) received his M.Sc. from Chalmers University of Technology, Sweden. He has a software engineering background, mainly within distributed systems, real-time OS, SDH communication platforms, and SS7. During 1995–1997 he was responsible for the activities related to WDM network management within the European research projects RACE-MWTN and then ACTS-METON. Currently, he is working as a technical product manager at Ericsson Infotech, Network Interworking division. He is chair of the SS7 sub-working group within IEEE P1520.

LOUIS-FRANCOIS PAU is general manager, Ericsson Utveckling AB (the core engineering company of Ericsson group); he was formerly technical director of Digital Equipment Europe. He holds M.Sc., Ph.D., and D.Sc. degrees from the Université de Paris, an M.Sc. degree from ENSAE, and an M.B.A. from IEP. Has been on the faculties of Danish Technical University, École Nationale Supérieure de Télécommunications, Paris, MIT, and the University of Tokyo.

MASAAKI SUZUKI (masa@ccrl.nj.nec.com) received his B.S. degree in electrical engineering from Tokyo University, Japan, in 1989. He was with the Transmission System Group of NEC, Kawasaki, Japan, from 1989 to 1996, working on hardware design of packet-based monitoring systems and system design of network management systems for SDH-based networks. Since April 1996 he has been with C&C Research Laboratories, NEC USA, Princeton, New Jersey, as a senior research associate. His research areas are a distributed object architecture applied to telecommunications systems and related QoS issues.

SOREN TORSTENSSON (Soren.Torstensson@ks.ericsson.se) received his M.Sc. in engineering physics at Chalmers University of Technology in 1982. Since 1982 he has been working for Ericsson, currently as a specialist in data and telecommunication.

WEIGUO WANG (wwang@krdl.org.sg), deputy director of Ubiquity Lab at Kent Ridge Digital Labs (KRDL), received his M.A. and Ph.D. in computer science from Boston University in 1985 and 1990, respectively. In 1990 he joined the Institute of Systems Science, Singapore, which recently became KRDL. He was responsible for networking research at ISS. AT KRDL he is leading a group on home networking and information appliances. He chairs the IEEE P1520 working group.

STEPHEN WEINSTEIN [F] (sbw@ccrl.nj.nec.com), a Fellow in NEC's Princeton C&C Research Laboratory, explores communications software architecture, access networking, and multimedia systems and applications. He received his S.B., M.S., and Ph.D. degrees in EE from MIT, the University of Michigan, and University of California, Berkeley, respectively. He was 1996–1997 President of the IEEE Communications Society. He is coauthor of the textbook *Data Communication Principles* (Plenum, 1992).