Provision of Signalling for Programmable Services

M. Banfield, S. Simpson, D. Hutchison

Distributed Multimedia Research Group,
Computing Department,
Lancaster University,
Lancaster, LA1 4YR,
United Kingdom
e-mail: [banfield, ss, dh]@comp.lancs.ac.uk

Key words: Programmable and Active Networks, Caspian, ReTINA, IEEE P.1520, SOAP

Abstract: Additional network complexity driven by the demand for new broadband services increases the need for network control and management signalling. This paper takes stock of this trend and suggests an approach within the context of IEEE P.1520 to separate signalling and associated broadband intelligent services from multimedia data transport so that each may be developed to their full potential. The authors draw on their experience of development of two signalling systems, one a TINA NRA inspired Connection Management System, and the second based on the P.1520.3 Programmability Architecture, in proposing a new Signalling Transport Service Provider rôle.

1. INTRODUCTION

The emergence of Intelligent Networks (IN) and other advances in telecommunications services such as the Telecommunications Management Network (TMN) have been built on extensions of the once simple call control signalling models. Even the Internet, whose connection-less IP protocol lacks an identifiable unique call set-up signalling stage, is adopting signalling through protocols such as RSVP and SIP.

This paper anticipates new initiatives within the rapidly expanding telecommunications, networking and information technologies industries. The trend is to introduce additional new functionality (programmable

services) into the network while consolidating the move towards deregulation and imposition of open free market conditions in the provision of communication services. It is well understood that, coupled with these market condition changes, the next generation of multimedia applications will generate new demands for Quality of Service (QoS), multicast and broadband services, which will in turn produce both economic and technological challenges.

The separate demands of future signalling and multimedia transport are analysed. The evolution of the Internet to provide a broadband multi-service transport medium will require the adoption of a sophisticated signalling environment. As opposed to traditional data services, multimedia traffic requires network resource management. The various uses of signalling include, end to end QoS management, and gathering of IP-POTS gateway resources. Far from replacing signalling, emerging technologies such as MPLS and DiffServ are introducing more control messaging into the Internet.

In heterogeneous services networks, where the current approach of dedicated service provision signalling protocols are inappropriate, a common signalling framework needs to be developed. Common strategies need to be adopted to provide consistent and reliable control messaging. The paper considers new open signalling and open market models (such as TINA and IEEE P.1520), and a new rôle, namely the Signalling Transport Service Provider (STSP), is proposed.

In making a preliminary evaluation of STSP we show results obtained on next generation signalling performance, with measurements made of two Open Signalling solutions. These are placed into context against alternative active technology signalling approaches.

2. THE EMERGENCE OF INTELLIGENT BROADBAND NETWORKS

In recent years, demand for advanced network services coupled with moves to deregulate the telecommunication market place have led to a number of important initiatives within the domain of open signalling. Currently network operators are forced to use multiple network control and management applications each tied to a particular technology (or worse still to a specific manufacturer's equipment). Such an approach is costly since it requires the high overhead of installing, training, running, and maintaining multiple management systems and reduces competition by tying an operator to a particular vendor's equipment.

This has resulted in a number of initiatives typified by TINA-C and IEEE P.1520 to develop architectures based on the specification of open interfaces. This should allow a single control and management system to be deployed that is independent of the underlying network technologies and the selection of vendor's switching equipment. Furthermore, the establishment of open interfaces enables the further introduction of competition into the telecommunications marketplace, enabling third party access to the underlying network resources. Of these many programmes, IEEE P.1520 is considered further in the paper; it is one of the most significant activities covering a broad spectrum of technologies including ATM switch control, SS7, MPLS, and Differentiated Services.

2.1 IEEE P.1520

P.1520 is an IEEE standards development project created in early 1998 by the OPENSIG community. Its aim [PIN-api] is to establish an open architecture for network control and define the interface between network control and management functions. Rather than specify static protocols these interfaces are designed to provide a powerful programmable API for the network infrastructure just as operating system APIs (e.g. Microsoft's Win32) currently provide to the application programmer.

The concept of programmable interfaces in P.1520 is an extension of the use of reference points in TINA; however, P.1520 does not prescribe the use of RM-ODP as a means of specification. Currently, interfaces have been described using a combination of OMG IDL (e.g. the 'L' interface for a DiffServ Router [PIN-IP007]) and in traditional protocols (e.g. the qGSMP representation of the CCM interface for ATM switches [PIN-ATM019]).

Central to P.1520 is the development of the four-layer P.1520 Reference Model inspired by the principle of opening the telecommunications infrastructure to the free market (Fig 1).

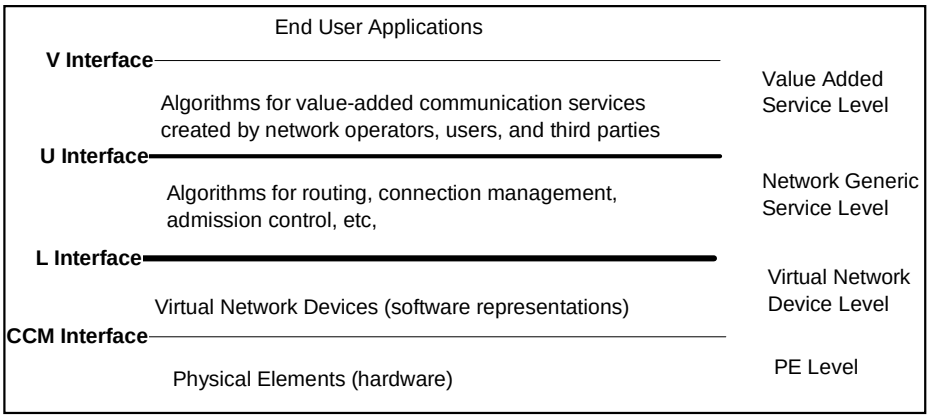

Figure 1. The P.1520 Reference Model

The model introduces four entities and four network API interfaces. The differentiation between the VASL (value-added service level), NGSL (network generic service level), and VNDL (virtual network device level) are made to separate the functionality into the classic three roles of hardware owner, network operator, and value-added-service provider.

3. DEFINITION OF THE SIGNALLING TRANSPORT SERVICE PROVIDER

The next generation of advanced telecommunications service architectures, such as TINA and P.1520 as well as the current IN approach, share in common a high degree of utilisation of inter-component signalling independent of the multimedia transport. This identifies a separation between the transport of multimedia data and the signalling data. The separation may be only logical, where signalling and multimedia data share a common physical network, or may be physical with the provision of an isolated network dedicated to transport of signalling messages.

The TINA approach for example, relies upon a distributed processing environment (DPE) to provide an object-oriented model that enables inter-component communication independent of location. This is likely to be closely related to the OMG's CORBA standards with extensions to suit the specific requirements posed within the telecommunications domain. A Kernel Transport Network (kTN) is provided to interconnect DPE nodes and provide inter-node communications. No assumptions are made as to the type of service provided by the kTN, which could be connection-less or

connection-oriented. The kTN communication is based upon a specialisation of GIOP, such as IIOP for an IP based kTN or an SS7 ESIOP (Environment Specific Interoperability Protocol) for a Signalling System 7 based kTN.

While much work has been performed on identifying the main business rôles and interfaces, little thought has been given to the provision of signalling functionality or the relationship established between this and the communicating entities. We add the concept of a Signalling Transport Service Provider (STSP) whose function is to provide the transport of signalling messages (Fig 2). This applies equally to TINA, P.1520 and IN architectures but, in this paper, P.1520 is used as the reference model for further exploration.

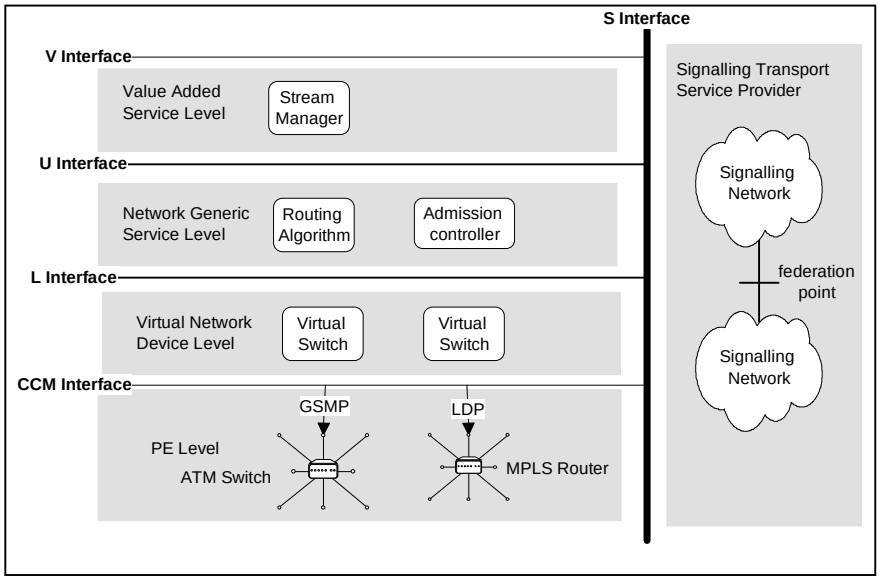

Figure 2. Introduction of the STSP and 'S' Interface into the IEEE P.1520 Reference Model

The isolation of this rôle extends the open market principles embraced in Open Signalling architectures. It provides for the outsourcing of signalling transport by the connectivity provider. This thereby allowing both the connectivity provider and the STSP to concentrate on building networks for efficient provision of their particular type of connectivity. This separation is, of course, a logical one. While in some circumstances the separation of STSP and multimedia transport network provider is made, in many cases it will be more appropriate in terms of cost for a single provider to combine both functionality sets. This detail is hidden from the control and management software, which should be unaware of how the signalling is transported, or of the ownership of the transport network.

Nevertheless, there are circumstances where this separation may be realised physically. The connectivity provider may not own a network appropriate for the transport of signalling (e.g. a wireless provider), or perhaps such high demands on network service quality are made that increased reliability through the use of an isolated signalling network provider may be appropriate. Another very important issue is that there can be no guarantee that value added service providers (located at the VASL) are physically connected to the multimedia network on which they compose their "value-added services". For example, a US based value added service provider may retail a VPN service over a network owned by British Telecommunications plc and located in the United Kingdom. A third party STSP, perhaps an ISP, can act as a communication enabler between these geographically disconnected providers.

4. REQUIREMENTS ON THE SIGNALLING TRANSPORT REFERENCE POINT

The IEEE P.1520 model is based upon a hierarchy of horizontal interfaces. In order to achieve our goal, we add to the model a new vertical interface (the 'S' interface) through which the VASL, NGSL, and VNDL establish a relationship with the signalling transport service provider (STSP). Two or more entities that wish to communicate must both form a binding with an STSP to enable them to do so. Since communication may be both intra-level (between entities located at the same service level) and inter-level (entities located at different service levels), the specification of the interface may be extended to service specific requirements noted for each signalling relationship.

The 'S' interface is used to negotiate the service requirements of the communicating entities much in the same way as a Service Level Agreement (SLA) is negotiated between an end customer and network operator over the UNI. This paper does not attempt to fully specify the interface down to IDL; this is a matter for further work and the authors believe there is much to be gained from utilising the techniques from RM-ODP[ODPspec]. Instead, we recognise five key areas of functionality that should be exposed over the 'S' interface:

1) Reservation of Service Quality – an agreed binding level of service quality must be negotiated between the parties. This could include high-level criteria such as "mean time to failure" and "percentage packet loss" or more detailed QoS characteristics such as bandwidth, delay, error rate and jitter.

2) Composition of Virtual Signalling Groups – although the signalling network provider may have many concurrent users of the signalling network, each user should be kept isolated from one another. Virtual signalling groups should be supported in a similar way to the provisioning of virtual private networks (VPN) to end-users. These groups may be used to constrain both intra and inter service level communication but should also enable federation between co-operating service providers.

3) Timescale and priority of interaction – a distinction is often drawn between the priority and timescale of connection control signalling compared to FCAPS functionality [ITU-T-M.3400] (Fault, Configuration, Accounting, Performance, and Security Management).

4) Quality of Protection – signalling messages may have very high security requirements, since the interception or spoofing of signalling may provide a loophole to compromise the integrity of the multimedia data.

5) Charging – given that the rôle of the Signalling Transport Service Provider may be outsourced to an alternative operator, provision should be made for a transparent charging system. Even where both signalling and multimedia transport are kept in-house it may be useful to know the associated cost of the signalling created by new intelligent and value-added services in order to charge correctly.

The next point to consider is the set of communications primitives provided to the users of the signalling network. Here, four possible options have been considered:

1) Network layer access – an OSI layer 3 service is provided to the customer (i.e. the Connectivity Provider of the multimedia network). The user can select their own protocols by either using those pre-provided (e.g. TCP installed in the DPE workstation's Operating System protocol stack) or by building customised protocols to their own needs.
2) Network and protocol selection – many high level protocols (OSI layer 4 and above) are designed solely for specific lower layer network infrastructures (OSI layer 3 and below). An example of this is IIOP, a specialisation of GIOP that applies only to IP networks. It is therefore not always appropriate to provide simple layer 3 access to the signalling network; instead the full protocol stack must be provided. This could be in the form of TCP/IP, IIOP/TCP/IP, or SAAL/ATM.
3) Universal communications primitives – given the previous option, it is indeed possible for the selection of a number of different protocols to be in concurrent use on a given signalling network (e.g. IIOP and RMI). It remains the job of the applications to agree, prior to communications, on choice of signalling protocols. Instead this option, the network exposes a set of basic universal communication primitives that can be used to

usefully express any communication requirement. The signalling network then maps these primitives down on to the most suited protocol for that particular communication session.

4) Full DPE Services – the final option is for the STSP to provide a full range of DPE functionality that the users can utilise to communicate. This would include RPC services, messaging services, naming services, streaming services, etc.

The third option of providing universal communication primitives seems the most appropriate option given the heterogeneity of types of network and service required. It hides the network provision from the customer while enabling the customers to construct a DPE environment suited to their own specific requirements. This could mean mapping the universal primitives to a DPE personality such as CORBA or RMI, or perhaps a Microsoft Message Queue (MSMQ) [MQref].

5. REALISATION OF OPEN SIGNALLING SYSTEMS

To investigate further the distinct requirements of signalling traffic, we have made an analysis of next-generation open signalling protocols. Recently, there have been a number of research projects creating consortiums large enough to build platforms to demonstrate the potential future developments in the management of advanced telecommunications services. Lancaster University has participated in a number of these, with the resulting platforms being established in our Distributed Multimedia Research Group (DMRG) laboratories.

5.1 The ReTINA Connectivity Service

ReTINA was until its completion during 1999 a TINA auxiliary project working to validate and develop the ideas encompassed in the TINA architecture. The goal of ReTINA was to develop and demonstrate an industrial-strength open distributed processing environment (DPE). This DPE supports distributed real-time multimedia applications over emerging broadband networks. The project embraced many areas of distributed system research including end system control, network management, DPE services, and software engineering tools. In this paper, we consider the results obtained from the evaluation of the Connectivity Service platform [ReTINA-cm] developed by Alcatel, Siemens, Broadcom, and Lancaster University.

The ReTINA Connectivity Service platform is based upon the principles of the TINA Network Resource Architecture [TINA-nra], and the Network Resource Information Model [TINA-nrim]. The platform is composed of two components, the TINA Connection Manger and an optional additional QoS Manager.

The network is represented by the Connection Manager through the composition of a hierarchy of Connection Performers (CPs) acting over child subnetworks (Fig 3) until, at the lowest level, the open interfaces to switch resources is reached. A separate signalling network known as a kTN is used to transport this inter-component DPE signalling.

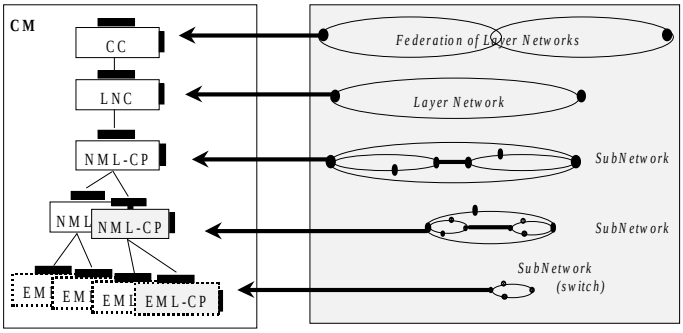

Figure 3. TINA Connection Management Architecture

Only basic call and QoS management functionality is provided by the Connection Manager. A QoS Manager can be optionally installed to provide advanced levels of QoS monitoring and estimation. However, this feature is not considered further as is a complex subject worth a publishable paper in its own right, and provides a more advanced QoS model than that exposed by other approaches such as ATM-F UNI v3.1.

Each Connection Performer (CP) exposes a CORBA interface to allow others to invoke requests to establish and release connections at the given subnetwork level. In addition, there are other more complex CP functions, many specific to the functionality appropriate for that particular hierarchical layer.

5.1.1 Implementation and results of the Connectivity Service

An implementation of this architecture was made on a Sun SPARC Solaris 2.5 platform using the Orbix v2.3 CORBA 2 compliant ORB from Iona. The Connection Manager was deployed over a cluster of three Sun Workstations all connected via an IP (over Ethernet) based kTN, so to simulate the distributed component nature of the TINA DPE. Element Layer mapping agents (using TINA terminology, the EML_CP) were then

developed to enable the control of AAL5/ATM encoded video streams across two ATM switches (Fore ASX100 and ASX200).

Measurements were made of the bandwidth used on the IP based signalling network by "snooping" and analysing packets in transit across the signalling network. The implementation is based upon Orbix generated IIOP v1.0 messages; this is a specialisation of GIOP v1.0 designed to work specifically over IP-based networks. The IIOP protocol operates over TCP/IP, which gives a low error rate but higher delay characteristics. These properties mean that the IIOP protocol is well suited to the QoS requirements in this signalling domain.

Obviously, the amount of signalling bandwidth consumed is dependent upon the number of switches used in a route and the height of the hierarchical decomposition performed. Indeed, it is unlikely that there will be a one to one mapping between DPE node and Connection Performer (CP) element (e.g. NML_CP); many CP's may be co-located thereby constraining communication to use of the operating system IPC mechanisms. Given this factor, it is therefore appropriate to measure the bandwidth consumed between two dislocated CP components. Since both NML_CP and EML_CP expose the same generic interface, there is no differentiation between the signalling bandwidth requirements of either component. Analysis of the signalling network traffic shows that the bandwidth used to request the set-up/release of a connection between a parent and child node in the hierarchy is dependent on the type of connection requests (e.g. uni/multicast). It is also dependent on whether the operation was successful or if it generated an exception. Table 1 shows the size of IIOP messages used to request the successful set-up/release of a unicast, unidirectional connection between two intermediate nodes (NML_CP).

Operation	Request	Reply	Total
setup_snc	287 bytes	198 bytes	485 bytes
release_snc	248 bytes	12 bytes	260 bytes

Table 1. Size of IIOP messages for connection establishment and release

These figures only express the bandwidth consumption point to point between two CPs. To obtain a realistic figure as to the total bandwidth used to establish/release a connection it is necessary to sum the multitude of individual CP to CP signalling. This is dependant on height and breadth of the hierarchical topology, and for the remainder of the paper we simplify this by assuming the construction of hierarchies with a regular branching factor (an example of this being a binary tree).

Given this hierarchical signalling relationship it is obvious that kTN bandwidth consumption may be reduced by the construction of broad flat

hierarchies. This infers a high branching factor, increased complexity routing performed at a given CP, and limited potential for computation distribution (at least at the granularity of the CP). Such properties restrict the potential scalability for concurrent use by introducing the bottleneck of nodes of high computation. This can be overcome by utilising the total combined power of the DPE nodes instead through the deployment of a less flat hierarchy of distributed CPs. Inevitably, there is a trade-off to be made against limiting kTN usage and exploiting DPE distributed concurrent processing. This is a complex problem that must take into account the requirements posed by the particular network deployment. However, for our purposes we consider the trade-off made on a simple network of 8 switches.

A network of 8 switches can be constructed in four regular hierarchies, 1x8, 1x2x4, 1x4x2, and 1x2x2x2. For each of these four hierarchies, measurements were made of kTN usage and processing time to establish a connection. Processing time is given as two values; average processing time per DPE node, and total DPE computational time. Per DPE node-processing time is a measure of the time from the reception by the CP (at level n) of a *setup_snc* operation to the time the CP invokes the set-up operation on the next CP at level n+1. Consequently the time includes both marshalling/de-marshalling computational time, plus the running of the routing algorithm on the topology modelled by the CP. Clearly this time is a factor of the DPE node platform (Orbix 2.3 on Solaris v2.5) and the efficiency of the ReTINA routing algorithm. However this is meant to illustrate the kTN and DPE processing trade-off and the figures should not be read as precise figures for all TINA Connection Management systems.

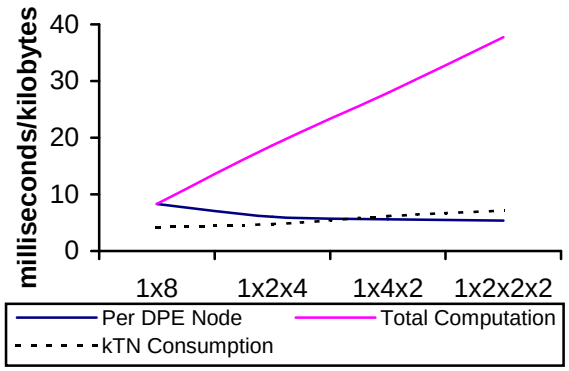

Figure 4. Trade-off between DPE computation and kTN usage

Observing the result (Fig 4), there is visibly a trade-off between per node processing, and kTN usage when choosing the appropriate hierarchy, the choice depending upon power of DPE nodes and scalability requirements of the system. If a flat hierarchy (1x8) is chosen, all connection establishment

requests must pass through a relatively slow root CP. When the 1x2x2x2 topology is used, processing time in the root CP per setup/release is less, enabling more connection requests to be processed by this CP node.

Total Computational time appears though to suggest that broad hierarchies are considerably more efficient in terms of DPE usage. This is true, but the figure is simply a summation of the per node DPE processing time for all nodes within the environment. Much of this computation is the marshalling/de-marshalling of DPE requests between nodes, and will be performed in parallel between DPE nodes, and does not represent latency of connection establishment to the customer. The Connectivity Provider when deploying their topology must, therefore, make a judgement between the costs of deploying many low powered parallel DPE nodes or a limited number high-powered nodes.

Extra bandwidth will, of course, be consumed on the kTN due to the overhead of the network and transport protocols. This must be coupled with any necessary fragmentation of the IIOP message, which is dependent upon the underlying kTN MTU (Maximum Transmission Unit) size. The scenario established within our laboratory utilised a kTN provided by an STSP based upon a 10Mbit/s Ethernet infrastructure.

Since IIOP utilises TCP as its transport protocol, it was possible to discover the likelihood of IIOP message fragmentation through "snooping" the TCP MSS (Maximum Segment Size) advertised by the TCP stack. Analysis with *tcpdump* showed the MSS to be 1460 bytes, consistent with the 40 bytes of TCP/IP header (both IP and TCP headers are by default 20 bytes) and the 1500 bytes maximum capacity of an Ethernet frame. In many legacy systems, X.25 is another likely kTN medium. Here too, fragmentation is unlikely to be necessary, given the 576 byte X.25 MTU.

In addition to this, there may be further signalling through the use of legacy switch control protocols. An example of this within early ReTINA demonstrators was the deployment of the EML_CP component on a Solaris workstation on a different kTN subnet than the switch. The ATM switch was not itself a DPE node, and set-up/release, QoS monitoring, and configuration requests where passed between the EML_CP and switch using the SNMP protocol. Later ReTINA Connection Managers incorporated the switch as a full DPE node, and part of the EML_CP was deployed on the switch itself, therefore not requiring extra SNMP communication over the kTN signalling network.

5.2 The PINE Approach

An alternative approach to heavily structured solutions typified by ReTINA is PINE (Programmable Interfaces for Network Elements), a

programmable network framework currently under development by Lancaster University within the EURESCOM funded Caspian (P926) project. Within this paper, PINE is chosen to highlight an alternative signalling model.

The PINE framework builds upon the wealth of background in Open Signalling architectures combined with the developments within the Internet community to provide a multiservice environment for dynamic service provision. PINE combines the IEEE P.1520 programmable architecture with dynamic injection of active code into network elements, two previously opposing approaches to advanced service provision.

The IEEE P.1520 IP Subworking group (now established as a separate project denoted P.1520.3) has identified the provision of the 'L' interface to be inadequate for the modelling of the control characteristics of IP network elements (e.g. routers). A two-layer abstraction was defined [PIN-IWAN] distinguishing between the "L+" service-specific interface and the "L-" service-independent interface, the latter being an abstract representation of an IP network element independent of whether the device in question is for example a DiffServ Router, or an MPLS switch. This interface, however, opens up the router's internals, something that many vendors may not be keen to do, and provides a very complex abstraction to program. The "L+" reference point is a service-specific interface (e.g. DiffServ Router Control Interface) providing an easy to program API that hides the underlying router complexity from the programmer. The result being that there is no single L+ interface, but a set of interfaces providing the programming of specialised services. Recently this model has been extended [PIN-IP013] to a three-tier abstraction model; "L+" interfaces now referred to as the "service-specific building block", the "L-" interface being the "resource building block". Additionally a "base building block" layer has been added that has no service or resource significance from a behavioural or packet-processing perspective. This division is currently under review and is unstable, and therefore PINE is built upon the preceding two-tier abstraction.

Our implementation of PINE is based upon the deployment of LANode (Lancaster Active Node) [LARA] routers at the Element Layer. These router nodes allow code providing control interfaces (at the "L+" reference point) to dynamically be uploaded and instantiated. The router can therefore be easily configured to provide different service abstractions of the routing resources, and specific service interfaces can quickly be deployed, tied to individual customers' requirements.

LANode offers a programmable API providing base abstractions of router resources (e.g. the routing table) on which the dynamically uploaded code can provision its service-specific control interfaces. As such, this

LANode programmable API corresponds to the "L-" reference point in the IEEE P.1520.3 model.

At the Network Management Layer (NML) subnetworks have a peer-to-peer relationship rather than the client/server relationship of the ReTINA Connection Management Architecture. Subnetworks are opaque in that they hide their underlying routing element infrastructure. From the SML viewpoint, each subnetwork is a virtual router, and just as with the physical routers, the virtual NML router supports the dynamic instantiation of new control interfaces. These control interfaces exposed by the virtual NML routers are equivalent to the "U" interface reference point (Fig 5).

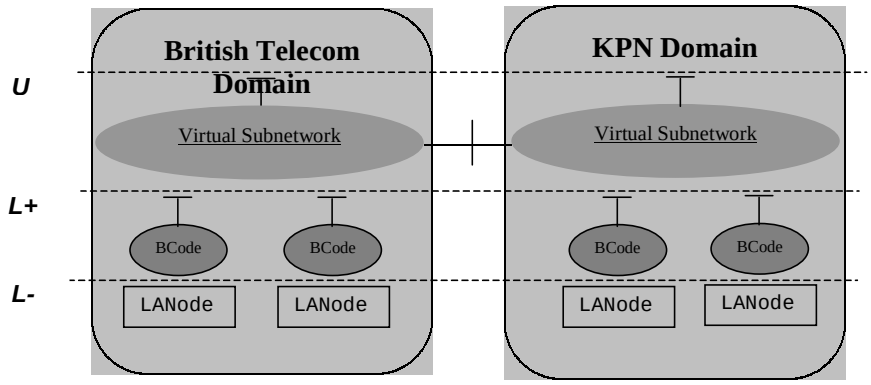

Figure 5. The PINE Architecture

5.2.1 PINE Signalling

Signalling within the PINE framework follows a distinctly different model to the ReTINA Connection Manager, with the use of the same physical IP network for carriage of both signalling and multimedia data. However, the distinction between signalling control data and the multimedia data are still maintained; this contrasts with most "active network" solutions which utilise in-band signalling: data and control are combined into the same stream, an example of such an approach being SwitchWare [Sware98].

Within PINE, two distinct types of control signalling exist. In the most coarse-grained approach, new algorithms can be uploaded to the active node that drastically alter the characteristics of the network element. For example, new packet scheduling algorithms could alter the type of Quality of Service (QoS) available, perhaps changing from Differentiated Service QoS to per flow provision. Coarse-grained signalling is therefore performed through the uploading of Java 2 bytecode that accesses the platforms "L-" API. The transport of this bytecode signalling data is carried over the conventional HTTP protocol. This has considerable advantages over the in-band approaches used by many conventional "active network" platforms and

typified by the ANEP protocol. Firstly, the HTTP protocol is well accepted today for transport of HTML documents; all firewalls already support the conveyance of HTTP data. PINE therefore supports the control of nodes over multiple security domains as our bytecode signalling traffic can pass freely through today's firewalls. Alternative protocols such as ANEP [ANEP97] cannot boast this advantage. Secondly, as HTTP is built upon TCP/IP which is supported by most modern OSes, many nodes can easily be extended to support PINE, either as a network node or as a control node inaugurating the uploading of bytecode. Alternative protocols (e.g. ANEP) are not supported by current protocol stacks, and require the addition of extra modules into the OS to capture, process and deliver (to corresponding process) the packet contents.

A finer-grained control is granted through the bytecode algorithms exposing L+ control interfaces. It is not appropriate to attempt to specify the method by which interfaces are instantiated since this can be tailored to the customer's requirements. It is, however, likely that a common RPC (remote procedural call) standard would be adopted, possibilities including CORBA/IIOP, DCOM, and RMI. In our implementation we have chosen a further alternative through the development of components exposing interfaces using the SOAP (Simple Object Access Protocol) [SOAPspec] protocol.

SOAP is a distributed RPC protocol based upon the common HTTP and XML standards. Existing RPC protocols (e.g. DCOM and IIOP) are poorly suited to the Internet environment. Wide-scale firewall deployment is often incompatible with current RPC mechanisms, many firewalls supporting only a few well established services (HTTP, SMTP, etc). Organisations have already invested great resources into HTTP security mechanisms (e.g. SSL) that can be readily utilised by SOAP, whereas by contrast other secure distributed mechanisms (e.g. DCE) require considerable investment.

Existing approaches (including DCOM and CORBA/IIOP) require hefty runtime support code in order to implement their rich service sets. Many of these features are rarely used, and inappropriate for this domain. Few network elements have built in support for DCOM or CORBA, where as HTTP support is now commonplace.

The SOAP protocol has the following advantages which make it appropriate for dynamic L+ interface provision:

a) HTTP supported by current firewalls – as explained earlier in regard to coarse-grained signalling, this enables support for inter-security domain signalling.
b) Lightweight – SOAP has cut down RPC functionality. Additional unused complexity is minimised so to keep interface implementations compact; important since this is carried within the signalling bytecode.

c) Object model agnostic – is independent of programming language, object model, processor, platform, and OS consistent with the heterogeneous network element infrastructure.

Figure 6 shows the signalling relationships within the element layer (LANode) of the PINE framework.

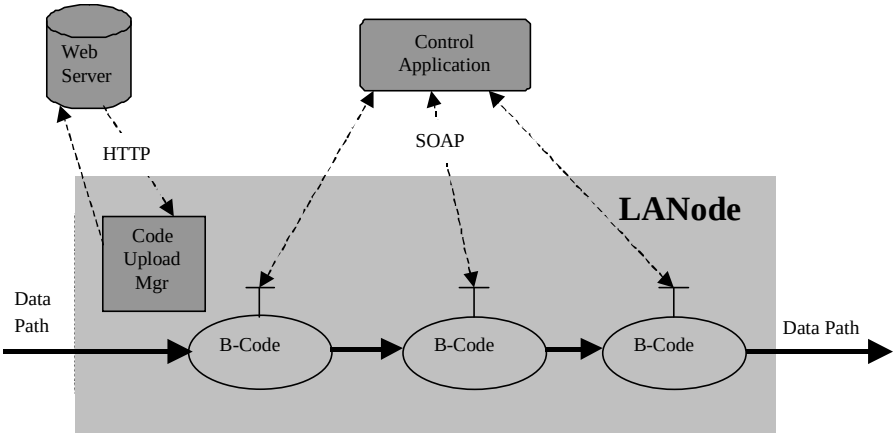

Figure 6. HTTP and SOAP signalling

Although SOAP is a new protocol it has evolved from several years of XML research [XML-RPC]. An internet draft has been submitted to the IETF, and SOAP has support from major distributed object technology vendors including Iona and Rogue Wave. Indeed Microsoft have added their voice to the SOAP community, adopting SOAP as an integral part of the Windows DNA 2000 Architecture [DNA2000].

As an example of a service deployed within the PINE framework, a simple application has been developed to expose an L+ interface to enable influence of routing tables by algorithms and software situated off-router. An example of such a scenario is the establishment of Virtual Private Networks over which the customer has direct control. The L+ interface exposes a view of the router resources tailored to the permissions of the customer.

An excerpt of this interface (defined in CORBA IDL for simplicity) is provided (Fig 7) to depict a typical L+ interface.

```
-----------------------------------------------------------------
  module routing {
    struct RouteId { string dest, mask; };
    struct RouteParams { string gateway; };
    struct Route { RouteId id; RouteParams params;};
```

```
    typedef sequence<Route> Routes;

    /* Accesses an L+ routing table. */
    interface RoutingTable {
      /* Get all routes. */
      Routes getRoutes();

      /* Get the parameters for one route. */
      RouteParams getParams(in RouteId id);

      /* Create a new route (or alter parameters of existing route). */
      void addRoute(in RouteId id, in RouteParams params);

      /* Delete a route.*/
      void delRoute(in RouteId id);
    };
  };
-----------------------------------------------------------------------
```

Figure 7. A Route Control L+ interface

Implementations have been made exposing both SOAP and IIOP implementations of this interface to run on LANode routers. This has enabled some comparison of the relative merits of both solutions and conclusions to be drawn. Three factors were considered for detailed analysis: size of SOAP/IIOP messages, time taken to process SOAP/IIOP request and response, and size of bytecode used to implement each solution.

The following series of results was obtained instantiating the routing interface on LANode router based on an Intel Pentium II 400MHz processor. The LANode software utilised the Linux 2.2.13 kernel, and a JVM based on the Blackdown JDK 1.2.2RC4 using native thread support. The IIOP instantiation of the L+ interface exploited Orbacus 3.2 (Java binding) a CORBA 2 implementation from Object Orientated Concepts [OOC]. The SOAP implementation used the SOAP/Java (version 0.3) message parser from DevelopMentor [DevM]. This also required the use of a SAX XML parser, for these testes the Xerces/Java v1.0.0 parser from Apache.

Measurements were made of SOAP and IIOP message sizes and round trip time for request/response. The “getParams()”, “addRoute()”, “delRoute()” methods detailed in the IDL in figure 7 were taken as example L+ interface calls. It was necessary to factor out the network traversal time, and any work the interface implementation may make on receipt of the call in order to achieve a true representation of the call processing time.

Table 2 shows the round trip request/response time

Method	IIOP	SOAP
getParams()	1.95ms	41.27ms
addRoute()	2.03ms	40.50ms
delRoute()	1.72ms	38.16ms

Table 2. Round trip request/response times for IIOP and SOAP interface implementations

These figures show SOAP to be considerably slower and require more processing than equivalent IIOP invocations. The "on the wire" size of the invocation and response message for SOAP was also found to be considerably larger.

Table 3 provides a comparison of the relative size of IIOP and SOAP request and response messages. Since SOAP is still relatively new and unknown, an example HTTP/SOAP message is illustrated in Fig 8.

RPC protocol	Request	Response	Total
IIOP	100 bytes	39 bytes	139 bytes
SOAP	609 bytes	520 bytes	1129 bytes

Table 3. SOAP and IIOP message sizes for getParams() method

```
-----------------------------------------------------------------------
POST /L+interface/routing HTTP/1.0
Content-Type:  text/xml
SOAPMethodName:  urn:schemas-lancs.ac.uk:caspian#delRoute
User-Agent:  Java1.2.2
Host:  wand:8888
Accept:  text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Content-length:  362

<?xml version='1.0' ?>
<s:Envelope xmlns:ns2='urn:schemas-lancs.ac.uk:caspian'
xmlns:xsi='http://www.w3.org/1999/XMLSchema/instance'
xmlns:s='urn:schemas-xmlsoap-org:soap.v1'>
     <s:Body>
              <ns2:delRoute>
                       <id s:href='#sid1' />
              </ns2:delRoute>
              <ns2:RouteId s:id='sid1'>
                       <dest>148.88.153.50</dest>
                       <mask>255.255.0.0</mask>
              </ns2:RouteId>
```

```
</s:Body>
</s:Envelope>
-------------------------------------------------------------------
```

Figure 8. Sample HTTP/SOAP message to invoke delRoute() method

Clearly the figures obtained show that SOAP performs poorly when compared to the IIOP protocol, this despite SOAP lacking the rich set of RPC features found in IIOP, and without the overhead of the CORBA object model (which could indeed by built over SOAP) in our Orbacus experiments.

However, within our chosen usage domain these issues are not the highest priority. The Java bytecode for these interface implementations are considerably larger than SOAP/IIOP messages. Given the rapid interface dynamic instantiation feature that requires the upload of bytecode onto the LANode router, it is important that the implementation bytecode be as compact as possible.

A comparison of the total bytecode size of the interface implementations is made in table 4. Class files that account for these totals have been selected based on their need to be present in code that exposes the interface of an object implementing RoutingTable. The ORB and the HTTP server have been excluded from these, since they would likely be provided locally on the system.

For both CORBA/IIOP and SOAP, this includes similarly sized parts such as classes representing types defined in the IDL, a skeletal interface implementation, and a main program to instantiate the implementation, and bind it to the relevant communication services.

In the IIOP version, a selection of other IDL-generated classes (stubs, helpers and holders) constitutes the bulk of the extra weight, since little of their function is needed for SOAP. What is needed includes a method dispatcher, SOAP body classes representing method requests and responses, and the registration of Java types to SOAP types, all of which have been hand-written.

The code-size advantage of SOAP has come from eliminating unnecessary code for simple distributed services.

RPC protocol in implementation	Size of implementation
IIOP	26129 bytes
SOAP	11241 bytes

Table 4. SOAP and IIOP implementation

This shows that although SOAP has a considerably higher overhead in terms of message size and required processing, the actual implementation is

approximately only 43% of the size of the equivalent IIOP implementation. Therefore, the choice as to which protocol to use to provide L+ interfaces is not specified within our PINE framework; the most appropriate protocol should be chosen given the usage requirements posed. This should include consideration of interface lifetime, likely interface usage, type of clients, and security restrictions.

6. RELATED WORK

The approaches to signalling identified within this paper represent only two possible solutions. In tandem with our "Open Signalling" inspired solutions, other frameworks have been developed with similar goals. Of particular interest is the Parlay Group, the Darwin project conducted at Carnegie Mellon University, and the "Active Reservation Protocol" being developed by USC/ISI.

The Parlay Group is a closed consortium formed in April 1998 by 5 partners (British Telecom, Ulticom, Microsoft, Nortel Networks, and Siemens). In 1999, the consortium was expanded further to include AT&T, Cegetel, Cisco, Ericsson, IBM and Lucent. Its rôle is to define an API [ParlayAPI] to expose network capabilities while ensuring network security and integrity. This enables the introduction of a new telecommunications business model [Parlay99], separating network ownership and service provision, similar to that of P.1520. However, the Parlay API only represents the "U" interface reference point and does not consider how these interfaces map onto the "L" interfaces (known within Parlay as Resource Interfaces) of the underlying network equipment. This is considered a matter for implementation, and therefore outside the remit of Parlay. Our PINE framework is compatible with Parlay; "U" interfaces supporting the Parlay API can be instantiated to meet customer demands.

The Darwin Project [DARWIN98][DARWIN99] aims to develop a framework for customisable resource management for the support of value-added services. Similar to our LANode routers used within PINE, limited active processing is supported on the control plane through "delegates", active code installed on network nodes. Delegates are however more autonomous when compared to PINE, where the purpose of active code is to provide control interfaces tailored to the specific customers services and the intelligence resides within the NML layer.

Naturally, delegates pose significant security concerns, which are addressed through both runtime (e.g. Java 2 JVM sandbox) and compile time mechanisms. Additionally, the concept of delegate owner is introduced to control those permitted to introduce delegates onto the router.

Other signalling is supported through Darwin's own Beagle signalling protocol. Unlike traditional signalling protocols (RSVP, and PNNI), Beagle signalling affects virtual network meshes rather than individual streams.

The "Active Reservation Protocol" (ARP) project [ARP99] at ISI is developing "a framework for implementing and deploying complex network control functions using an active network approach". It is argued that rather than the need for a long protocol standardisation process, only a single implementation is required to form a new standard (the standard is the implementation code itself). Similar to the LANode routers within PINE, a programmable interface at the "L-" reference point is defined known as the "Protocol Programming Interface" (PPI). Although the approach shares a similar design, the goals are different; ARP aims to enable rapid deployment of new (network element to network element) protocols as opposed to exposure of programming interfaces within PINE.

7. DISCUSSION AND CONCLUSION

This paper has discussed the context and the increasing rôle of signalling in the provision of telecommunication services. The signalling generated by increased broadband IN functionality spawns Quality of Service requirements quite separate and distinct from those of multimedia data.

Many of the emerging models such as TINA and IEEE P.1520 choose to push the rôle of provision of signalling traffic into the functionality provided by the Distributed Processing Environment (DPE). However, the current DPE architectures, including CORBA, DCOM, and RMI, are designed as generic DPE platforms. They do not tackle the issue of provision of a signalling network, aiming to be independent of the underlying network infrastructure. This paper therefore highlights signalling network provision, and suggests the isolation of signalling provision as a specific rôle to consider when designing an advanced-services network. A context for this rôle of signalling provision has been provided through the explanation of its relevance to the TINA and IEEE P.1520 architectures.

These next-generation approaches to the provision of advanced network services are realistic and realisable. Prototype realisations have been established in research labs such as ours. As has been shown, although signalling has distinct characteristics from multimedia data, signalling itself is quite diverse. ReTINA and PINE characterise two distinct approaches to the provision of signalling transport; ReTINA adopting a separate signalling network (known as the kTN), while PINE uses the same network for signalling as for data transport.

While minimising signalling bandwidth is obviously an important consideration, it is not the only aim of an intelligent broadband signalling protocol; other properties such as extensibility and computation distribution may be considered. For example, although considerably more efficient, the ATM Forum's UNI3.1 protocol lacks the extensibility of the open signalling approach. The efficient location of signalling nodes need not be made so as to minimise signalling but to maximise distribution of processing. Such a decision is unique to each particular situation, taking account of costs of signalling bandwidth against cost of computational nodes.

Despite the claims made by some about the Internet's lightweight approach to signalling, the provision of adequate signalling functionality is of increasing importance. Recently there has been considerable interest on network security following attacks on prominent Internet sites. Governments fear terrorist attacks on our networking infrastructure at a time when we are increasingly reliant on an on-line society. Much of the interest has gone into securing end-to-end data communications on an individual stream through adoption of encryption protocols. Yet security of signalling provision is of even greater importance since, if compromised, it has the potential to bring down or violate the entire network. Recently the important rôle of intelligent signalling technology was highlighted in the UK with the outage of several IN services. While we do not know if this was simply a software fault or a malicious attack, the unavailability of 0800 (freephone) numbers to key services caused significant disruption to many people.

Signalling security is of great importance and Quality of Protection (QoP) has been identified as a key goal of the identification of the separate rôle of signalling provision. Both ReTINA and PINE attempt to address this issue, ReTINA with the separate signalling network, and PINE by using regular secure HTTP technologies (e.g. SSL). However, this issue is far from solved, and we expect ever more advanced approaches to be developed to match the increasing sophistication of network hackers.

In conclusion, those involved in the development of advanced network services are concerned with the construction of ever more complex architectures. It is clear that the signalling network is the foundation of an advanced services network, and without adequate provision of the signalling network, the whole architecture cannot be effectively realised.

ACKNOWLEDGEMENTS

We gratefully acknowledge the EURESCOM funded Caspian project (P926) and ACTS programme ReTINA (AC048) project for the sponsorship of M. Banfield and S. Simpson.

REFERENCES

[ANEP97] Alexander, D.S., Braden, B., et al., "Active Network Encapsulation Protocol (ANEP)", July 1997, http://www.cis.upenn.edu/~switchware/ANEP/

[ANON99] Tschudin, C., "An Active Networks Overlay Network (ANON)", IWAN'99, Berlin, June/July 1999.

[ARP99] Braden, B., "ASP EE: An Active Execution Environment for Network Control Protocols", http://www.isi.edu/div7/ARP, December 1999.

[DARWIN98] Chandra. P., et al. "Darwin: Resource Management for Value-Added Customisable Network Service", IEEE ICNP, Austin (USA), October 1998.

[DARWIN99] Takahashi, E., et al. "A Programmable Interface for Network Resource Management", IEEE OPENARCH, New York (USA), March 1999.

[DevM] SOAP/Java version 0.3 message parser, DevelopMentor Inc., http://www.develop.com/soap/

[DNA2000] "Windows DNA 2000", Microsoft Developer Network Journal, Issue 16, January/February 2000, Pages 18-22

[LARA] Cardoe. R., et al. "LARA: A Prototype System for Supporting High Performance Active Networking", IWAN'99, Berlin, June/July 1999.

[MQref] Dellinger, C., "Codename Falcon: The Microsoft Messaging Queue", Microsoft Interactive Developer Journal, March 1998.

[ODPspec] ISO/IEC, "Open Distributed Processing", ITU Recommendation X.901 – X.904, 1995

[OOC] jOrbacus v3.2, Object Orientated Concepts Inc., http://www.ooc.com

[ParlayAPI] Parlay Group, "Parlay API Specification 1.2", The Parlay Technical Team, September 1999, http://www.parlay.org/

[Parlay99] Parlay Group, "Parlay API Business Benefits White Paper", Issue 1.0, June 1999, http://www.parlay.org/

[PIN-api] Biswas, J., et al. "Application Programming Interfaces for Networks", IEEE P.1520 Whitepaper, January 1999.

[PIN-ATM019] Adam, C., Huard, J-F., Lazar, A., Nandikesan, M., "The qGSMP Protocol", [P1520/TS/ATM-019], IEEE-P.1520 ATM subworking group, July 1999

[PIN-IP007] Raguparan, M., Biswas, J., Wang, W., "P1520-L Interface requirements for IP routers that support differentiated services" [P1520/TS/IP007], IEEE-P.1520 IP Subworking group, May 1999.

[PIN-IP013] Vicente, J., et al. "Proposed L- Interface Architecture", [P1520/TS/IP013], IEEE-P.1520.3 IP Subworking Group, January 2000

[ReTINA-cm] Banfield, M., Edwards, C., Charton, N., Hutchison, D., "Providing Scaleable QoS-based Connectivity Services", TINA'99 Conference, April 1999.

[SOAPspec] Box, D., et al. “SOAP: Simple Object Access Protocol”, INTERNET-DRAFT <draft-box-http-soap-01.txt>, Internet Engineering Task Force, November 1999.

[Sware98] Alexander, D.S., et al. “The SwitchWare active network architecture”, IEEE Network, 12(3):P29-36 May/June 1998.

[TINA-bm] TINA-C, “Business Model and Reference Points Version 4.0”, May 1997

[TINA-nra] TINA-C, “Network Resource Architecture v3.0”, Deliverable no: NRA_v3.0_97_02_10, February 1997.

[TINA-nrim] TINA-C, “Network Resource Information Model Version 3.0”, Deliverable no: NRIM_v3.0_97_12_17, December 1997.

[PIN-IWAN] Denazis, S., et al. “Designing Interfaces for Open Programmable Routers”, IWAN’99, Berlin, June/July 1999.

[XML-RPC] UserLand Software Inc, http://www.xml-rpc.com/