Abstract

The core Internet technologies were in the hands of the research community 10 or more years before the World Wide Web happened and popularized the Internet as a place to find information, access service, and trade. The Infrared Data Association has been in existence for over six years. Products embedding the communication technology IrDA defines have been around for over five years, starting with printers and portable PCs. IrDA is cheap to embed, uses unregulated spectrum, and is increasingly pervasive in a wide range of devices. From its roots in portable PCs and printers, IrDA technology is present in virtually all new PDAs, it is emerging in mobile phones, pagers, digital cameras, and image capture devices. We are sitting on the cusp of the information appliance age, and IrDA is playing a significant role in enabling the interaction between information appliances, between information appliances and the information infrastructure, and between appliances communicating across the information infrastructure. This article discusses IrDA's communications model. It charts the evolution of the IrDA-Data (1.x) platform architecture, and the early applications and application services now in common use. It considers the present day and the explosion in device categories embedding the IrDA platform. It broadens its horizons to consider other emerging appliances technologies and to consider communications models that might arise from a blend of IrDA short-range wireless communications and mobile object technologies. Finally, it briefly considers future directions for the IrDA platform itself.

IrDA: Past, Present and Future

Stuart Williams, HP Laboratories

The Infrared Data Association (IrDA) was formed in June 1993 and has worked steadily to establish specifications for a low-cost, interoperable, and easy-to-use wireless communications technology. Today, the infrared data communication technologies defined by the IrDA ship in over 40 million new devices each year ranging from personal computers, personal digital assistants (PDAs), digital cameras, mobile phones, pagers, portable information gathering appliances, and printers.

It is a remarkable achievement for a new communications technology to establish such widespread deployment in such a wide range of devices in such a relatively short time. The core Internet platform technologies existed for a full 10 years prior to the explosive growth brought about by the introduction of the Web.

IrDA is a communication technology for the appliance era. This is an era that, while not excluding the PC, liberates devices that have long been viewed as peripherals. It enables them to engage in useful interactions with each other without having to mediate their communications through some common control point.

End users have remarkably high expectations of wireless communications. In the wired world there is general acceptance of the mechanical constraints imposed by the various plugs and sockets that, at least in part, avoid mismatched connections. There is acceptance of the cognitive load required to sort out the connectivity and clutter of cabling at the rear of a hi-fi setup or the back of a PC. However, in the wireless world, there is an expectation that communications and connectivity will just work, and work simply. In the wired world short-range connectivity between devices is established by explicit actions on the part of the end user. In the wireless world there is an expectation that connectivity between devices will be established as required without explicit intervention by the end user. The expectation is that if the user attempts to print, the "system" will seek out and establish connectivity to a nearby printer.

The author regularly finds it remarkable that he can use the same infrared port to:

- Simply "squirt" files between devices
- Connect to the local LAN
- Dial in from a portable PC or PDA via an IrDA-enabled cell phone
- Print to an IrDA-enabled printer

All of this is achieved without reconfiguring between actions and in most cases merely by placing the appropriate devices in proximity to one another.

The work of IrDA has sought to go far beyond mere cable replacement, and provide a communications platform and application services fit for the era of information appliances and which excel in the area of ease of use.

A Brief History of IrDA-Data

The IrDA was formed in June 1993 to develop an interoperable, low-cost and easy-to-use, short-range, infrared, wireless communications technology. The inaugural meeting was attended by 70+ companies which recognized the considerable value of defining a single family of specifications for the communication of data over infrared.

Prior to June 1993, a number of noninteroperable single-vendor proprietary schemes for infrared data communications existed. There was considerable risk that the marketplace for short-range wireless infrared communications would fragment around a number of proprietary schemes, all of which would individually fail to achieve critical mass. For system and peripheral vendors eager to deploy short-range wireless solutions in their information appliances, the absence of a dominant, common connectivity technology represented a void. Without a dominant technology, the risk of choosing the wrong proprietary technology was significant. Thus, there was considerable shared interest in the generation of common specifications, and this set the tone for the early years of the IrDA.

The original requirements can be summarized as:

- Marginal cost to add infrared to a product , under $5
- Data rates of up to 115 kb/s
- Range from contact (0 m) through at least 1 m
- Angular coverage defined by a 15–30 degree half-angle cone

By the end of September, IrDA had selected one of 3 proposed approaches for defining its physical layer [1] defined by

 1070-9916/00/$10.00 © 2000 IEEE

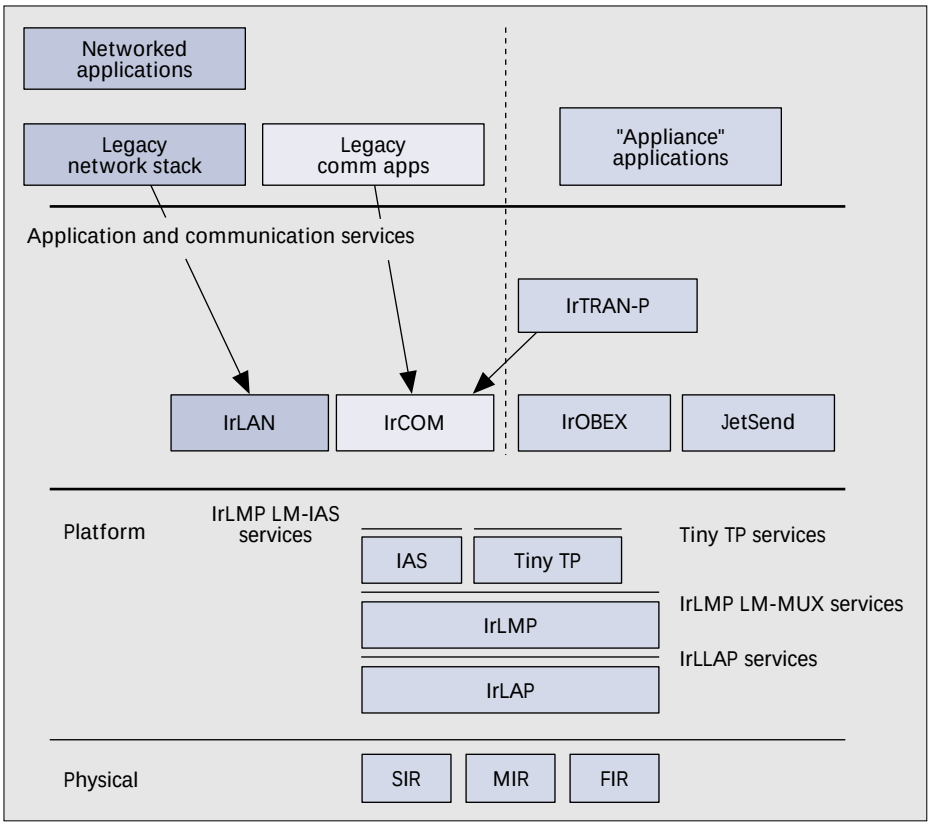

Figure 1. *The IrDA protocol architecture.*

Hewlett-Packard. All three approaches assumed the presence of a UART that could be used to modulate the infrared transmissions. The silicon cost of UART devices was well understood, and in many cases the system design of many products included redundant UARTs; thus, the marginal cost of adding IrDA could amount to just the components of the infrared transceiver.

So far, these requirements have little to say about the functional model of communication. There was an implicit requirement that the infrared medium serve as a cable replacement, but, as we shall see later, the question of which cable remained.

The natural abstraction of a half-duplex, asynchronous character-oriented transmission was too poor an abstraction for building interactions that were self-organizing and easy to use. In addition, there were frequent discussions of how to select data rate, how media access control was to function, and how, in the context of a 115 kb/s link, reasonably efficient use could be made of the available bandwidth.

By November 1993 IrDA had settled on a token-passing approach, originated by IBM [2] and derived from high-level data link control (HDLC) [3] operating in normal response mode (NRM). As with other proposals, this was a packetized scheme. However, in contrast to contention-based schemes that were also considered, the HDLC-SIR (later renamed Infrared Link Access Protocol, IrLAP [3]) approach yielded contention-free access to the medium once initial communication had been established. IrLAP defines a fixed-rate slotted contention-mode device discovery scheme that enables initial contact to be established. Critical communication parameters such as connection data rate, maximum packet sizes, and certain minimum and maximum gap timings are negotiated during connection establishment. Following IrLAP connection establishment, the two devices engaged in communication are deemed to "own" the spatial region which they both illuminate — nominally the union of two overlapping 1 m cones, each with a 15–30 degree half-angle.

It soon became apparent that the definition of IrLAP would not be sufficient to meet IrDA's ease-of-use goals. Certainly, IrLAP would provide a reliable connection- oriented communication service between two devices, but it provided no means to identify prospective clients of the IrLAP communication services. The year 1993 was a "hot" period with the emergence of numerous PDAs, notebook, and sub-notebook PCs. It was apparent that a model which turned over the infrared communication facilities to a single application would be inadequate. The emerging multithreaded consumer computing platforms required a multiplexing communications model that enabled several applications to share access to the infrared communications resources within a device. In this way, multiple applications could passively listen for appropriate peer application entities to connect. Thus, in December 1993 the activity to define the Infrared Link Management Protocol (IrLMP) [5] was born.

IrLMP provides a connection-oriented multiplexer, LM-MUX, and a lookup service, LM-IAS, that enables multiple IrLMP clients to claim a "port" above the multiplexer and advertise their availability by placing critical contact information into the lookup service. The namespace for the lookup service is designed to be self-administering in order to avoid the bureaucracy of maintaining administrative records about namespace registrations and to ensure "fair access" to make use of the namespace.

By June 1994, just 12 months after the inaugural IrDA meeting, version 1.0 of the core IrDA platform specifications, IrPHY, IrLAP, and IrLMP, was released [4–6].

Work continued to define a per-connection flow control scheme to operate within IrLMP connections. When multiplexing above a reliable connection, unless there is a means of independent flow control for each derived channel, the delivery property of the derived channel is reduced to "best-effort." Per-channel flow control restores a "reliable" delivery property. This work led to the definition of the Tiny Transport Protocol (Tiny TP or TTP) [7].

IrPHY, IrLAP, IrLMP, and TinyTP are the currently accepted specifications that define the core of the IrDA platform, often referred to as the IrDA-Data or 1.x platform. The platform has been extended three times to accommodate:

- The addition of 1.152 Mb/s and 4 Mb/s data rates
- The inclusion of a short-range, low-power option primarily for use in devices such as mobile phones where battery life is paramount
- The addition of a 16 Mb/s data rate

It was not enough merely to define a communications platform. In order to promote interoperability between applications, it was essential to develop specifications for the application services and the application protocols that support them. Hence, work has also progressed to define application

protocols and services that reside above the IrDA 1.x platform, most notably:

- **IrCOMM** [8], which provides for serial and parallel port emulation over the IrDA platform. This allows legacy communications applications to operate unchanged over IrDA and also provides for wireless access to external modems. The most novel example of the latter is NTT's deployment of IrDA-enabled integrated services digital network (ISDN) payphones.
- **IrLAN** [9], which provides wireless access to IEEE 802 style LANs.
- **IrOBEX** [10], which provides for the exchange of simple data objects and could be considered the IrDA analog of HTTP. IrOBEX delivers on the notion of "squirting" information objects such as business cards, phone lists, calendar entries, and binary files between devices.
- **IrTRAN-P** [11]: which provides for the exchange of images between digital still image cameras, photo printers, and PCs.
- **IrMC** [12], which defines a profile of relevant IrDA specifications for inclusion in cell phones. Much of this work is being leveraged by the Bluetooth community. IrMC provides for vendor independent interactions with common cell phone features such as phone list synchronization, calendar synchronization, and wireless modem access. It also provides for third-generation smart phones.
- **IrJetSend** [13, 14]: which describes how to bind Hewlett-Packards JetSend protocol for networked appliance interaction to the IrDA platform.

Figure 1 below summarizes the IrDA-Data platform and application services defined to date.

The discussion so far has focused on the history of the standards development process. Table 1 below shows key milestones in terms of the introduction of classes of products implementing various mixes of applications services.

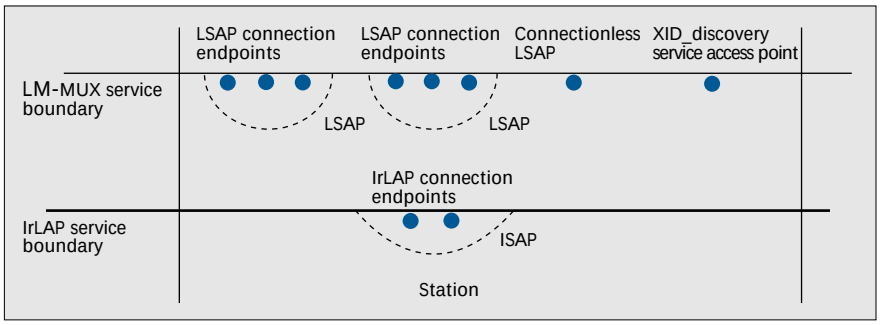

Figure 2. *Service access pointsand connection endpoints.*

IrDA 1.x Platform Architecture

In this section we describe the layered protocol architecture of the IrDA-Data 1.x platform, the services provided at its layer boundaries, its connection model, and the information model and philosophy of its device and service discovery processes.

Figure 1 shows the layering of the IrDA protocol architecture and many of the application services mentioned in the previous section. The upper boundary of each of the boxes represents an interface where the services of that layer are abstracted.

The segmented physical layer provides packet transmission and reception service for individual packets, and the means to determine when the infrared medium is busy.

The IrLAP layer provides for the discovery of devices within range and the establishment of reliable connections between devices.

The IrLMP layer provides connection-oriented multiplexing services with both sequenced and unsequenced delivery properties (LM-MUX services) and the service information access service (LM-IAS). LM-MUX provides for multiple logically independent channels between application entities within the communicating devices. Note that the absence of per-channel flow control in LM-MUX channels means that they may only safely be regarded as best-effort delivery channels.

Tiny TP mirrors the LM-MUX services; however, it augments them with the inclusion of per-connection flow control. This restores the reliable delivery properties for sequenced data. Tiny TP provides a null pass-through for unsequenced data whose delivery properties remain best-effort.

LM-IAS provides query/response services on an information base that contains essential contact information that enables prospective service users (clients) to identify and bind to service providers (servers).

These four protocol layers, IrPHY, IrLAP, IrLMP, and Tiny TP, form the core of the IrDA platform.

IrDA Connection Model

The IrDA 1.x connection model is established primarily by the IrLAP and IrLMP layers. There is a 1:1 correspondence between IrLMP LM-MUX service access points (LSAPs) and Tiny TP service access points (TSAPs). Thus, the Tiny TP layer does not contribute to the connection model, it merely alters the delivery properties of the channel from best-effort to reliable.

Within each IrDA device (or station) (Fig. 2), IrLAP services are accessed via a single IrLAP service access point (ISAP). The architecture allows multiple IrLAP connection endpoints to exist within the ISAP; however, in practice the IrLAP protocol defines only single point-to-point connectivity. There are no known research or

Approximate Introduction Date	Device Category
Late1994	115.2 kb/s Optical Transceiver Components
Early 1995	115.2 kb/s Personal Laser Printers Serial Port Adaptors Printer Adaptors
Mid 1995	115. kb/s Portable PCs Windows '95 Portable Ink Printers
Late 1995	4 Mb/s Optical Transceivers
Mid 1996	4 Mb/s Portable PCs 4 Mb/s LAN (Ethernet) Access Devices Nokia 9000 Communicator Windows CE 4 Mb/s Personal Laser Printers
Late 1997	Digital Cameras Mobile Phones
Mid 1998	Palm Compting Platform (Palm III) Casual Capture and Share Information Appliance
Early 1999	IrDA/Linux Implementation Sony PocketStation

Table 1. *Product categroy introductions.*

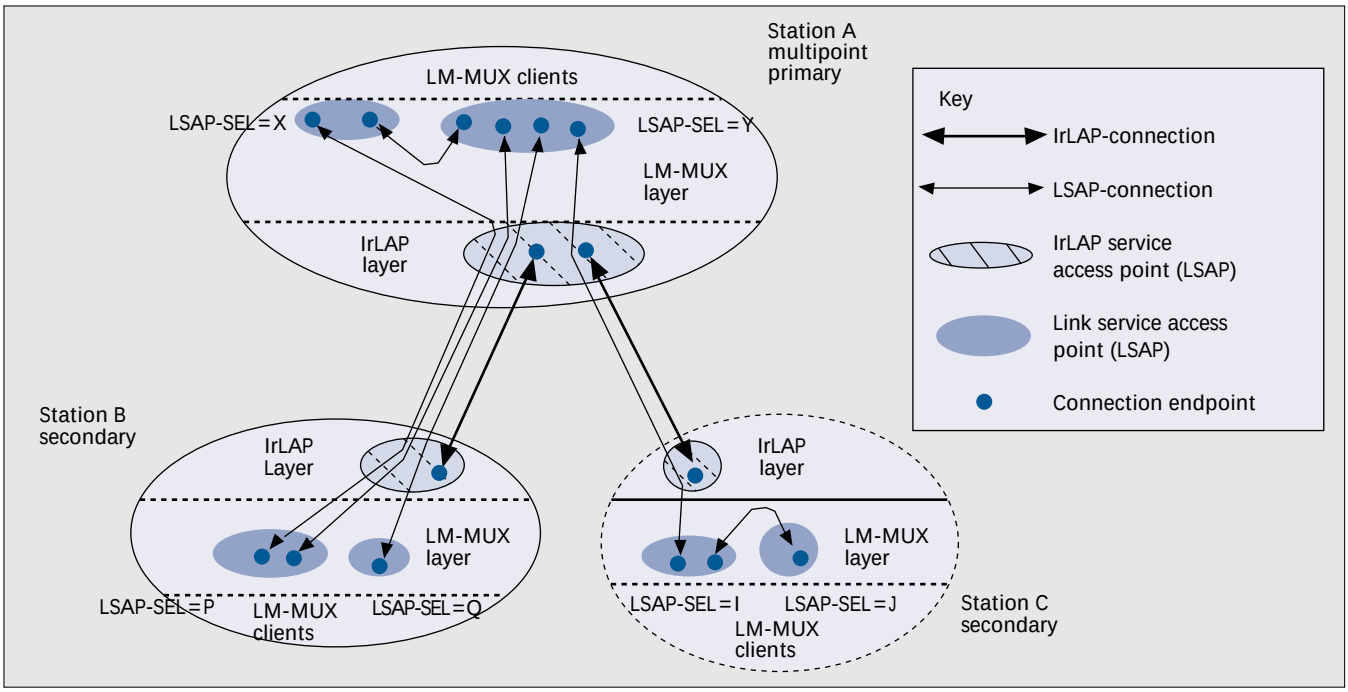

Figure 3. *Connection model.*

commercial IrDA stacks that support point-to-multipoint connectivity. However, one commercially available implementation supports multiple IrLAP interfaces and gives the impression of multipoint operation through multiple independent instances of IrLAP and IrPHY.

Likewise, IrLMP LM-MUX services are accessible via multiple LSAPs. Typically, an application entity will bind to an LSAP and, in general, will support multiple IrLMP LM-MUX connections (or Tiny TP connections). Thus, each LSAP may contain multiple LM-MUX connection endpoints. LSAP addresses are formed by the concatenation of an 8-bit LSAP selector and the device address of the device where the LSAP resides.

Figure 3 illustrates the IrDA 1.x connection model in the case of point-to-multipoint connectivity.

IrLAP connections are labeled by the (unordered) pair of 32-bit device addresses of the devices involved in the connection. Following connection establishment, a temporary 7-bit connection address is used in the packets as an alias for this concatenated device address.

Likewise, IrLMP LM-MUX connections are labeled by the (unordered) pair of LSAP addresses at each end of the LM-MUX connection. A corollary of this is that at most only a single LM-MUX connection may be established between any two LSAPs.

This connection model is identical to that offered by TCP/IP where, semantically, IP addresses may be substituted for IrDA device addresses and TCP/IP port numbers are substituted for IrLMP LSAP selectors.

Device and Service Discovery

IrLAP provides a basic device discovery mechanism. Functionally, the result of invoking the IrLAP discovery process is a list of records that encode:

- Device Address: A 32-bit semi-permanent device identifier of the discovered device.
- Nickname: A short multilingual name for the discovered device that may be presented in user interfaces to aid in selection.
- Hints: A bit mask giving nonauthoritative hints as to the services that may be available on the discovered device. This may be used to order "deeper" queries into the IAS to authoritatively establish the presence or absence of a particular service.

The device discovery process is further abstracted through IrLMP by defining procedures for the resolution of conflicting device addresses, and "hiding" such issues from the LM-MUX user.

Device discovery enables entities within one device to establish the presence of other devices. However, for a system to be largely autoconfiguring and to operate with minimum unnecessary intervention from the end user, it is essential that application entities within one device be capable of identifying and establishing contact with peer entities. These peer entities share a common interface (or application protocol) that enables them to interact. Contrast this with the situation where an end user is faced with the problem of ensuring that the right applications are bound to the right serial ports, or that the correct serial ports are connected together and the appropriate pin-pin mappings have been installed in the cable depending on whether the connection is DTE-DCE or DTE-DTE and on particular idiosyncrasies in the device's serial port implementation.

LM-IAS defines:

- A set of operations that an IAS client may invoke on an IAS server
- The behavior of an IAS server
- An information model for representing the application services accessible at a given device

Starting with the information model, each application service is represented by a named IAS object class. The name of the object class reflects the name of the service and may be up to 60 octets in length. A hierarchical naming convention is used to avoid name space clashes and to minimize the administrative burden on the IrDA office. It also in effect provides open and equitable access to the class namespace. Thus, classnames that start "IrDA:" are defined by IrDA, while classnames that start "Hewlett-Packard:" are defined by the Hewlett-Packard Company, and so forth.

An object class acts as a container for a list of attribute/value pairs. Attributes are named, and in general the attribute namespace is scoped by the enclosing class. Howev-

er, by convention some attributes are of such global utility that they are deemed to have the same semantics in all scopes. Such attributes carry hierarchically structured names that follow the same syntactic conventions as the IAS classname. Thus, IrDA:IrLMP:LsapSel and IrDA:TinyTP:LsapSel are the names of globally scoped attributes that carry the LSAP selector portion of the address of the entity represented by an instance of the object. IrDA:IrLMP:InstanceName is a globally scoped attribute used to carry a distinguishing name that may be used in user interfaces to aid in selection when multiple instances of a given service are found on a single device.

There are three attribute value types:

- Integer: A 32-bit signed integer.
- User strings: Intended for presentation via a user interface; up to 255 octets in length with multilingual support.
- Octet sequence: An opaque sequence of up to 1024 octets of information. The attribute may impose further structure on the contents of the sequence. This is a good way to cluster a body of information under one attribute.

IrLMP defines a number of operations for traversing and retrieving information from an IAS information base; however, only the GetValueByClass operation is mandatory. A possible C function prototype for the client operation would be:

```
AttributeValueList
        GetValueByClass(ClassName class,
                AttributeName attribute);
```

where the result type, AttributeList, encapsulates a possibly empty list of object instance ids and attribute values from objects that match given object and attribute names. Thus, a single invocation may result in responses for multiple object instances, and further attributes, such as instance names, may need to be sought in order for an appropriate choice to be made.

The IrDA platform provides a space for the definition of new applications and application services above the platform. In defining new services it places three obligations on the service designer:

- The definition of an IAS object class
- The definition of a hints mask that indicates the strong likelihood that an instance of that service exists on the discovered device
- The definition of the semantics of the application level interaction and the communication stack profile(s) that provides the channel for the interaction

IrDA-Data, 1.x Platform Summary

Before moving on to consider some of the application services defined above the IrDA platform, a brief recap of what we have described so far is whorthwhile.

The IrDA platform provides a connection model identical to that provided by TCP/IP. The semantics of the Tiny TP transport service are sufficiently close to those of TCP that in practical implementations they can be provided through an application programming interface (API) based on Berkeley sockets.

Naming and addressing in IrDA differs from TCP/IP naming and addressing. Device addresses are flat and dynamically assigned. While device addresses change infrequently, automated processes do force change when conflicts arise. Both the names and addresses of devices are explicitly discovered. Neither are assumed to be known a priori. Services in the IrDA environment are named using IAS classnames. These names are dynamically mapped to IrLMP LSAPs and/or Tiny TP TSAPs through IAS queries. This dynamic mapping reduces the administrative burden imposed on the IrDA office. With the limited 7-bit "port" address space of the LM-MUX, it also removes the problem of organizations making unfair claims on address space real estate.

Device discovery and LM-IAS provide the pivotal ease-of-use features in the platform that enable application entities to locate and establish contact with peer entities which support a given interaction protocol (i.e., the semantics of the message set exchanged between application entities via the channel established through the IrDA platform).

Advanced Infrared

The IrDA-Data 1.x architecture has some obvious limitations.

First, although the architecture can accommodate a point-to-multipoint mode of operation, the IrLAP specification has never been extended to define the protocol machinery to enable that functionality. From an end-user point of view it is also questionable whether such extension of the 1.x platform is even desirable. Viewed as a single point-to-point link, the behavior of an IrLAP connection is largely symmetric, and the differences in behavior between an IrLAP primary station and an IrLAP secondary station are largely moot. However, the introduction of point-to-multipoint operation would significantly disturb this symmetry in ways that would become inconvenient for the end user. Consider a portable computer that needs to access both a LAN access point and a desktop printer. It would be natural for the portable computer to become the IrLAP primary and establish IrLAP connections with the LAN access point and the printer, each of which acts as an IrLAP secondary station. However, it is also reasonable that the LAN access point (or the printer for that matter) is capable of "serving" multiple "clients," but in order for it to do that it would itself have to take on the IrLAP primary role. If the connection to the LAN access point were established first and the access point were to cease the primary role (possibly through role reversal), the portable computer would be unable to establish a second IrLAP connection to the printer. If the portable computer retained the primary role, it could establish that second connection, but the LAN access point (and the printer) would be prevented from establishing connections to other potential "clients." What the user could achieve would not only depend on the set of concurrent interactions they were attempting to initiate, but also on the order in which those interactions were initiated. This would lead to inconsistent behavior which would become frustrating for end users. Thus, IrDA so far has chosen not to expand the IrLAP definition to encompass point-to-multipoint operation.

Second, within some given field of view, the establishment of an IrLAP connection between a single pair of devices inhibits the establishment of connections between other independent devices whose fields of view intersect that of the established connection. Thus, use of the medium becomes dedicated to a single pair of devices. An important subclass of general multipoint communication in a shared medium is to enable multiple independent pairs of devices to establish independent communication relationships. If two devices are in view of each other, it is reasonable that they should be able to establish communications and share access to the medium with other users of the space.

Thus, members of the IrDA community sought to extend the IrDA-Data architecture to enable true multipoint connectivity while at the same time preserving the investment in upper layer applications and services by ensuring that the semantics of the service definitions at the upper layers of the platform are maintained.

It is important to be aware of a few differences between the goals of the IrDA community and the goals of those defining wireless LAN specifications. The IEEE 802 medium access control (MAC) service defines a best-effort ordered delivery service with *at most once* delivery semantics. It also

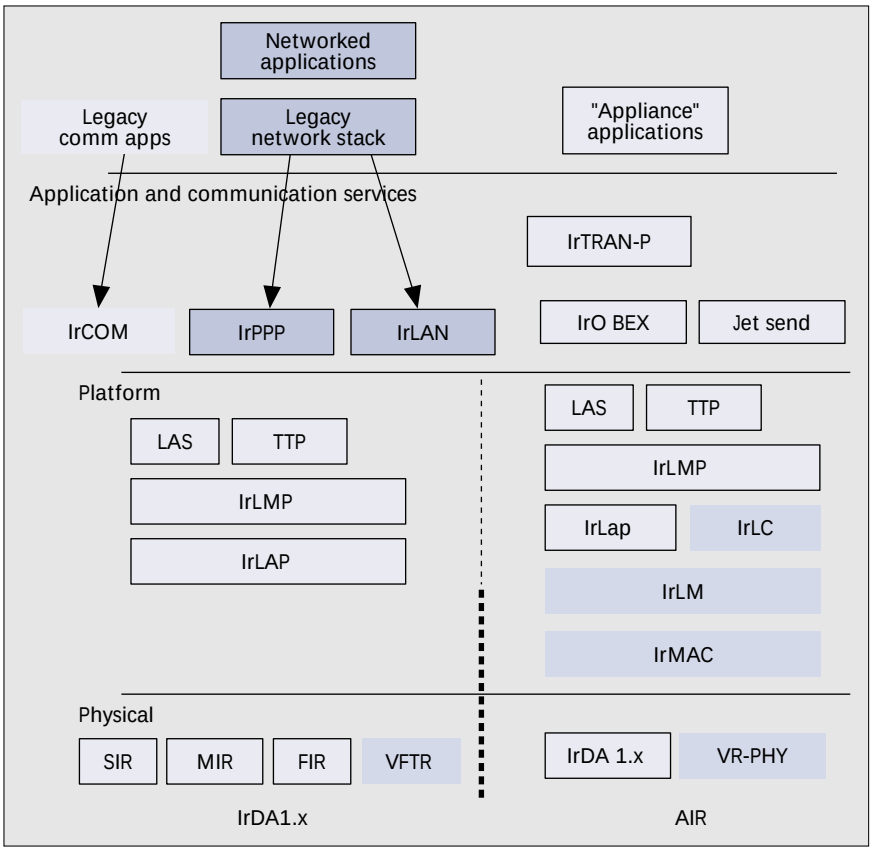

Figure 4. *IrDA data advanceIR protocol architecture.*

assumes a transitive communications relationship. At the MAC layer:

IF A can communicate with B
AND B can communicate with C
THEN A can communicate with C.

In the world of short-range infrared wireless communication this is not the case. It may even be the case that the communication relationship is asymmetric: A may be "heard" by B, but B cannot be "heard" by A. With the wired LAN medium the notion of "belonging" to a particular LAN segment is strongly associated with a physical attachment to that segment. Arrivals and departures, infrequent in most wired cases, can be noted by both the arriving/departing node, and potentially the wired infrastructure and the other devices attached to that infrastructure. In the wireless case, the bounds of a given LAN segment are less well defined, and arrival and departure are much more the norm.

Thus, the primary goals in extending IrDA-Data's connection model were:

- To enable devices in view of one another to establish communication relationships uninhibited by the connection state of nearby devices.
- To enable an advanced infrared (AIR) device to establish communications with at most one IrDA 1.x device. This enables AIR devices to interoperate with legacy 1.x devices in a way that is well understood by users of legacy 1.x devices.
- For AIR devices to respect established IrDA 1.x connections with which they could interfere. This is a coexistence requirement intended to ensure that AIR devices do not disrupt active IrDA 1.x connections

From an architectural point of view it is relatively simple to introduce multi-access communication. It requires that IrLAP be partitioned into a MAC layer (IrMAC) [14] and a link control layer (IrLC) [16]. In fact, as illustrated in Fig. 4, the AIR protocol architecture adds an IrLC protocol entity, which provides multipoint link layer connectivity alongside an IrLAP protocol entity, which provides legacy connectivity to IrDA 1.x devices. The link manager (IrLM) layer [17] is a "thin" layer that multiplexes the use of IrLAP and IrLC over their respective physical layers. At the upper bound of the link control layer, IrLC and IrLAP provide identical services to the IrLMP layer. Thus, within an AIR device, the IrLAP, IrLC, and IrLM protocol entities may be regarded as a single logical entity, with a single device address which supports an integrated discovery process, and both IrLAP and IrLC connections. The particular use of IrDA 1.x IrLAP or AIR IrLC procedures for establishing device-to-device connections becomes transparent to IrLMP entities and above.

IrMAC defines a burst reservation carrier sense multiple access with collision avoidance (CSMA/CA) MAC protocol. Such MAC protocols rely on the exchange of Request-To-Send, Clear-to-Send MAC protocol data units (PDUs). For such a mechanism to function properly, it is important that the reservation MAC PDUs be decoded, not only by devices capable of engaging in communications, but also by devices capable of interfering with communications. In some radio frequency (RF) systems using this style of MAC protocol, the range of the reservation messages is extended by boosting the transmission power for the related MAC PDUs. AIR makes use of a variable rate (VR) coding technique known as *repetition coding* to robustly code the headers of all AIR MAC PDUs. The use of this technique was pioneered by IBM Research in Zurich [18], and full details are given in the AIR physical layer specification [19]. Repetition coding trades signal-to-noise ratio (SNR) (range) for transmission rate by repetition of physical layer symbols. Repetition decoders "average" the repeated symbols received prior to making a decision on what symbol was encoded. In theory, successive halving of the data rate by successively doubling the number of symbol repetitions yields approximately a 19 percent range increase at each reduction step. Cumulatively, the effect of a 16-fold rate reduction (four doublings of the repetition rate) yields a doubling of the effective range of transmission. Key fields of the AIR IrMAC [17] PDUs are coded with 16x repetition.

This VR physical layer delivers two benefits. First, it provides a means of robustly coding the media reservation messages defined by the CSMA/CA burst reservation MAC protocol defined in the IrMAC specification so that they reach more potential sources of interference.

Second, by actively monitoring the symbol error rates at an AIR decoder, it is possible it estimate the SNR for the channel between the source and sink of a packet. This SNR estimate can then be used to compute a rate recommendation that can be fed back to the sending station in order to maintain a good-quality channel. AIR's base rate is 4 Mb/s, reducing through successive halving of data rates to 256 kb/s to yield a doubling of range. AIR prototypes have been demonstrated that have a 4 Mb/s range of 5 m doubling to 10 m at 256 kb/s.

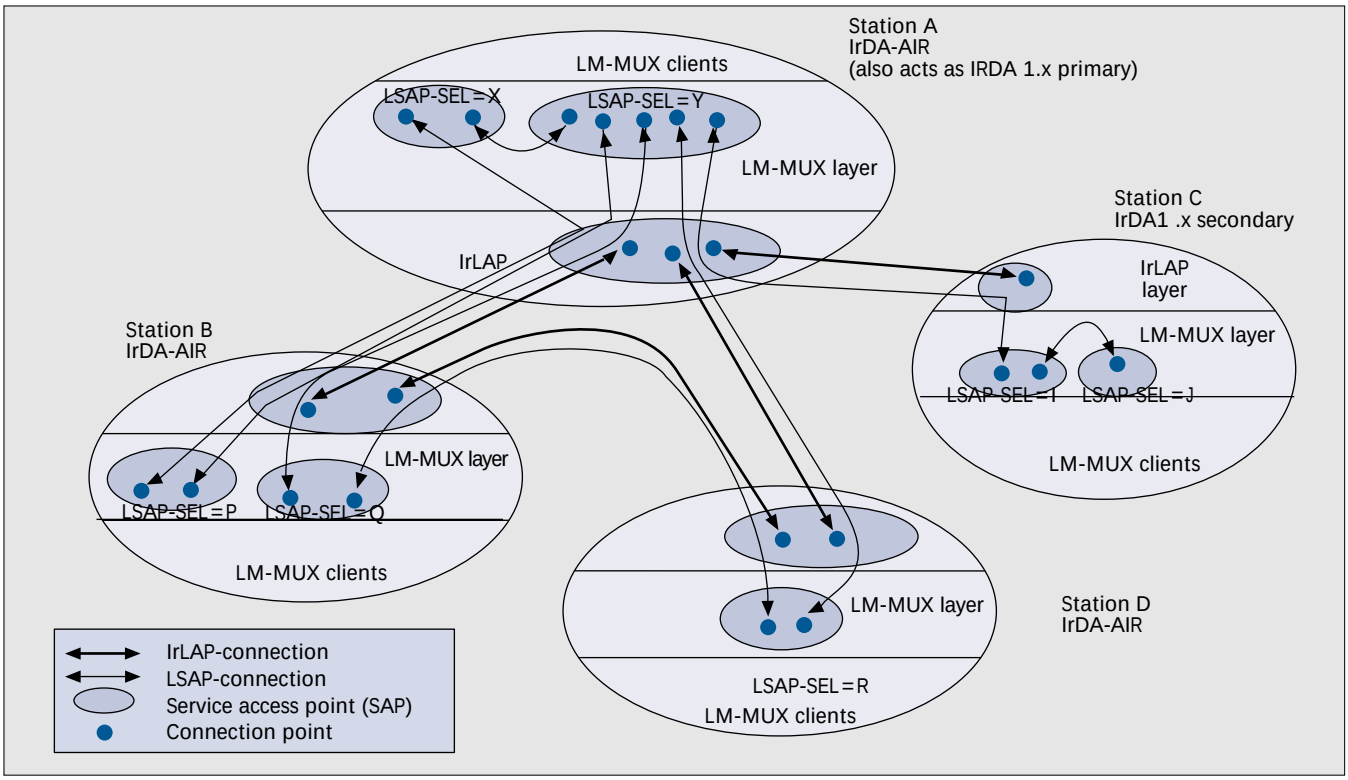

Figure 5. *IrDA-data AI connection model.*

The AIR Connection Model

Figure 5 illustrates the extended AIR connection model regarding the IrLAP, IrLC, and IrLM entities within a station as single protocol entity, with a single service access point (ISAP), capable of supporting multiple connection endpoints. LSAP addresses, as before, are formed by the concatenation of an IrLAP/IrLC device address and an IrLMP LM-MUX LSAP selector. IrLMP LM-MUX connections, as before, are labeled by pairing the LSAP addresses from each end of the LM-MUX connection. As before, there is a 1:1 correspondence between Tiny TP connections and LM-MUX connections.

IrDA Legacy Communications Services

Serial port emulation and LAN access were regarded as two forms of cabled connection that IrDA could replace. While serial port communication is acceptable at link rates of up to 115.2 kb/s, LAN access provided one of the primary motivations to push the data rate to 4 Mb/s and beyond.

IrCOMM

There were two key motivator behind the definition of IrCOMM [8].

First, there was the perceived need to provide a serial cable replacement for legacy applications. Thus, traditional communications applications, such as Kermit, could be used over an emulated serial port connection. At first glance this may appear a little odd. However, the IR medium, at least as used by the IrDA, is a half-duplex medium. Data can only flow in a single direction at a time. The IrDA platform imposes a media access discipline that shields platform clients from the need to be aware of the limitations of the medium and abstracts multiple duplex channels. In addition, many communications applications make use of flow control and modem control lines. There is a need to communicate the "virtual" state of these signals in addition to the character data exchanged between devices.

Second, there was a need in the telecommunications community to enable infrared access to the telephony network.

IrCOMM provides for both DTE-DTE (null modem) communications and DTE-DCE communications. In addition, unlike in the wired equivalent of these scenarios, the end user is completely unaware of the difference.

IrCOMM is typically implemented as a serial port driver which, rather than interacting with real serial port hardware, implements the IrCOMM protocol to interact with a peer IrCOMM entity to transfer data between their respective clients.

IrCOMM supports 3- and 9-wire modes of operation. The 9-wire case includes modem control lines. The relative timing of modem control line state changes can be maintained to the level of the boundaries between characters.

Applications running over IrCOMM do not benefit from all the ease-of-use features the IrDA platform provides. This occurs because of the very nature of serial communications.

The open/close operations in most serial APIs are about local resource allocation, typically a local serial port resource. They are not about (physical) connection establishment. Indeed, it becomes obvious after a few moments that serial APIs cannot be expected to provide for physical connection establishment. That is something that the user does... with a cable! Therefore, there is no way at the serial port API level for an application to express anything through the API about the nature or identity of the remote peer with which it wishes to establish communication.

In the same way as it is possible for an application to open a real serial port when there is no cable plugged into it, it is possible for an application to open an IrCOMM virtual serial port when there is no device in the vicinity with which to communicate. Once opened, an IrCOMM virtual serial port will actively seek out peer entities with which to communicate. Typically, instances of IrCOMM will renew efforts to find a peer as applications continue to "pump" data into the "virtu-

al" serial port. The presence of buffered data may stimulate periodic discovery processes. Semantically, data pumped into a disconnected serial port can be lost.

As with real serial ports, the end user needs to explicitly control which applications are bound to the virtual serial port. It is possible for a given device to support multiple virtual serial ports. However, a device encountering another device with multiple virtual serial ports is faced with a problem. This problem is similar to that of encountering a device with multiple real serial ports where the labeling next to the serial ports has been rubbed out, and where the mapping between operating-system-specific logical device names (/dev/tty0, COM1, etc.) and physical ports is unknown.

This is an ugly problem for real serial communications and remains an ease-of-use challenge for legacy applications in the world of short-range wireless communications.

Nevertheless, IrCOMM has found widespread application in mobile phones, and there has been significant deployment in ISDN payphones in the Tokyo area. Such deployment allows relatively trouble-free dialup access to the internet.

IrLAN

IrLAN [9] adopts a similar philosophy to IrCOMM in order to support legacy networking applications.

Typically, IrLAN operates in "access-point" mode where the combination of the IrLAN protocol operating over the IrDA stack to an IrLAN LAN access point is equivalent to a regular LAN card and provides 802 LAN MAC service. Just as IrCOMM is typically implemented as a serial port driver, IrLAN is typically implemented as a LAN card driver.

IrLAN can operate in a peer-to-peer mode that effectively models a stub-LAN with just two devices attached. Legacy networking protocols such as classic NetBios can establish ad hoc network communications file and print sharing in such an environment.

However, peer-to-peer IrLAN operation is of little use for most TCP/IP implementations since it fails to provide the infrastructure such as DHCP for autoconfiguration, Domain Name Service (DNS) for name-address resolution, or any of the other infrastructure services whose existence is generally assumed by applications programmed for the TCP environment.

IrLAN has proved controversial. Its sustained deployment in operating systems is in question and is likely to be superceded by a model based on Point-to-Point Protocol (PPP) which leverages superior access controls, autoconfiguration, encryption, and header compression facilities from PPP.

IrDA Application Services

IrOBEX

IrOBEX [10] seeks to be regarded as the HTTP of IrDA. Like HTTP, it precedes the transfer of some data object with a set of headers that carry meta-data about the object such as its name, size, and type. IrOBEX supports both PUT and GET operations.

Currently, the dominant use of IrOBEX is for the transfer of file-like objects such as document files, electronic business cards (vCards), and appointments (vCalendar). Content typing and naming conventions can be used to launch applications appropriated for the content type being transferred. HTTP is used typically using a PULL model, whereas the dominant usage model for IrOBEX is a PUSH model. A user "squirts" information from one device to another. The premier example is that of the exchange of business cards between PDAs. The author has also used a commercially available IrOBEX implementation to transfer the accumulated 2 Gbytes of files built up on one "congested" portable PC to its successor one evening in the living room. That single instance of use alone could, for many, justify the $3–5 cost the inclusion of IrDA adds to a portable PC.

IrOBEX is defining semantics for sequences of PUT and/or GET operations to achieve such objectives as address book and diary synchronization between phones, smart phones, PDAs, portable PCs, and pagers.

IrTRAN-P

IrTRAN-P [11] is an image transfer protocol defined by members of IrDA for the exchange of images between digital cameras, PCs, PDAs, and photo-printers.

IrTRAN-P cameras can be used in conjunction with PDAs, mobile phones, and smart phones to accomplish such things as gathering data, and e-mailing or faxing it back to the office. A more social variation of this notion might be dubbed the "Instant Postcard."

IrTRAN-P saw widespread deployment in digital cameras in the Asian marketplace prior to the 1998 Nagano Winter Olympics. These cameras have been used in conjunction with the previously mentioned ISDN payphones and street kiosks. These kiosks either generate small printed address book stickers or enable the images to be uploaded to Web sites so that they can be shared more widely.

IrJetSend

JetSend [14] is an open appliance interaction protocol defined by Hewlett-Packard. JetSend-based appliance interactions embody the strong principle that the participating appliances do not model each other. JetSend devices share a model of the structure of information, e-material, and can engage in multilevel negotiation over the encoding used to exchange information. A mandatory encoding exist for ALL defined content types, ensuring a base level of interoperability between JetSend appliances. JetSend also provides a control protocol such that appliances may render UI artifacts on behalf of appliances with which they are connected.

Underlying all JetSend interactions is an object interaction model based on surface interaction. This is an unconventional object model. Consistency of object state is maintained through the application of internal rules. Some portion of an object's state is exposed at its surface. The surfaces of connected objects are impressed on one another (simple equivalence between connected surface artifacts). Dynamic behavior arises from disturbance of state and the application of the internal rules until the overall state becomes stable or cyclic.

Interaction policies enable interactions to be structured to represent the transfer of a document, or the dynamic status of a device to be monitored.

JetSend was originally deployed in imaging and hardcopy devices connected to the Internet. More recently, a means of binding JetSend to the IrDA platform has been defined [13]. A number of commercial devices exist that both source and sink JetSend eMaterial over the IrDA platform.

IrDA Futures

IrDA has grown up late in the era of the PC and at the start of the information appliance age. Today it is exciting to see its adoption in such diverse devices as mobile phones, PDAs, pagers, digital cameras, portable scanners, access points, and printers. It is exciting to consider the device and service interaction possibilities that a pervasive short-range communication technology enables. The range of device categories that embed the IrDA communication platform is expanding. The means by which such devices can gain access to infrastructure over an initial short-range hop is expanding. This leads to an

explosion in the potential for devices to interact with one another, either in the immediate vicinity or remotely across infrastructure. It leads to an explosion in the potential for devices to interact with entities and services embedded within the infrastructure.

However, with all this potential there is massive challenge. Do we aggregate functionality into ever more complex appliances, or do we separate functionality into devices focused on doing one thing well? Can we find methods of interaction that enable the functions of a collection of single-function devices and infrastructure-based services to be composed to achieve some larger end-user goal which we might regard as a dynamically composed application? Can we do this in ways that are intuitive for end users, where the outcome of an interaction needs to be both predictable and useful? If I send an electronic business card from an organizer to a mobile phone, what is the mobile phone expected to do? It could dial the number on the card, "file" it away in its address book, fax it to some remote recipient, and in the not too distant future it might e-mail it somewhere else. What is that same phone expected to do with a photograph sent from a digital camera? Both of these interactions are possible with off-the-shelf IrDA-enabled products today.

The information exchanged between devices need not be static, either. Financial transactions between electronic wallets are being accomplished over IrDA [20]. The advent of virtual machine environments and mobile objects such as Java/RMI and distributed computing frameworks such as Jini provide the opportunity to apply these technologies in an IrDA environment. This leads to the possibility of exchanging active objects, perhaps representing goal-driven agents, between devices and between devices and infrastructure.

At some point as we go up through these layers of abstraction, it is no longer appropriate to confine those abstractions to the IrDA environment. They become of much broader utility, and another significant challenge will be to deliver these capabilities through upper-layer application services in consistent ways across a range of lower-layer short-range wireless technologies.

In IrDA AIR is in the final stages of being adopted. It offers to improve the freedom of movement available to IrDA devices that adopt the technology. They will be freed from the restrictive 1 m,15–30 degree half-angle coverage profile of IrDA 1.x, and enabled to operate over greater range and wider angle. Just as with RF wireless technologies, this will present new user interface challenges too, since, in general, the target of a given interaction can no longer be resolved merely by pointing.

IrDA offers a flexible, globally pervasive short-range wireless communications platform for appliance builders. Its widespread adoption In devices to date greatly reduces the risks associated with embedding it in new device categories. Its range of application will continue to grow well into the new millennium.

Acknowledgments

The author would like to thank the IrDA president, Mike Watson of Calibre-Inc., and Parviz Kermani of IBM Research, as well as the Hewlett-Packard Company for their support, encouragement, and review of this article.

References

[1] P. D. Brown, L. S. Moore, and D. C. York, "Low Power Optical Transceiver for Portable Computing Devices," U.S. Patent No. 5,075,792, Assignee: Hewlett-Packard Company, Dec. 24, 1991.
[2] T. F. Williams, P. D. Hortensius, and F. Novak, "Proposal for: Infrared Data Association Serial Infrared Link Protocol Specification," v. 1.0, IBM Corp., Aug. 27, 1993.
[3] ISO, "High Level Data Link Control (HDLC) Procedures - Elements of Procedures," ISO/IEC 4335, Sept. 15, 1991.
[4] IrDA, "Serial Infrared Link Access Protocol (IrLAP)," v. 1.1, June 1996.
[5] IrDA, "Link Management Protocol," v. 1.1, Jan. 1996.
[6] IrDA, "Serial Infrared Physical Layer Link Specification," v. 1.3, Oct. 1998.
[7] IrDA, "TinyTP: A Flow-Control Mechanism for Use with IrLMP," v. 1.1, Oct. 1996.
[8] IrDA, "IrCOMM: Serial and Parallel Port Emulation over IR (Wire Replacement)," v. 1.0, Nov. 1995.
[9] IrDA, "LAN Access Extensions for Link Management Protocol IrLAN," v. 1.0, July 1997.
[10] IrDA, "Object Exchange Protocol," v. 0.1a, Jan. 4, 1995.
[11] IrDA, "IrTran-P (Infrared Transfer Picture) Specification," v. 1.0, Oct. 1997.
[12] IrDA, "Specifications for Ir Mobile Communications (IrMC)," v. 1.1, Mar. 1999.
[13] IrDA, "JetSend(tm) Protocol on IrDA Application Note," v. 1.0, Sept. 1998.
[14] Hewlett-Packard, "HP JetSend Communications Technology Protocol Specification," v. 1.0, July 1997
[15] IBM Corp., "Advanced Infrared (AIr) MAC Specification," v. 0.3, Jan. 1999; available at IrDA Members FTP Site.
[16] IBM Corp.,"Advanced Infrared Logical Link Control (AirLC) Specification," v. 0.1, Jan. 1999; available at IrDA Members FTP Site.
[17] IBM Corp., "Advanced Ir Link Manager. (AirLM) Specification," v. 0.3, June 1999; available at IrDA Members FTP site.
[18] F. Gfeller *et al.*, "Wireless Infrared Transmission: How to Reach All Office Space," *Proc. VTC '96*, vol. 3 pp. 1535–9.
[19] IrDA, "IrDA Advanced Infrared Physical Layer Specification," v. 1.0, Sept. 1998.
[20] Confinity Inc., http://www.paypal.com

Biography

STUART WILLIAMS (skw@hplb.hpl.hp.com) received Bachelor of Science and Ph.D. degrees in electrical and electronic engineering from the University of Bath, United Kingdom in 1981 and 1986. He is a technical contributor at Hewlett-Packard Laboratories, Bristol, England. Between 1994 and 1998 he was responsible for managing a research team focused on the use of freespace infrared for short-range wireless data communications. During this period he made numerous technical contributions to the work of the Infrared Data Association (IrDA) and has served as co-chair of the IrDA Technical Committee. Prior to joining Hewlett-Packard in 1992, he worked for seven years on LAN and routing technologies at British Telecom's Martlesham Heath Laboratories.