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.

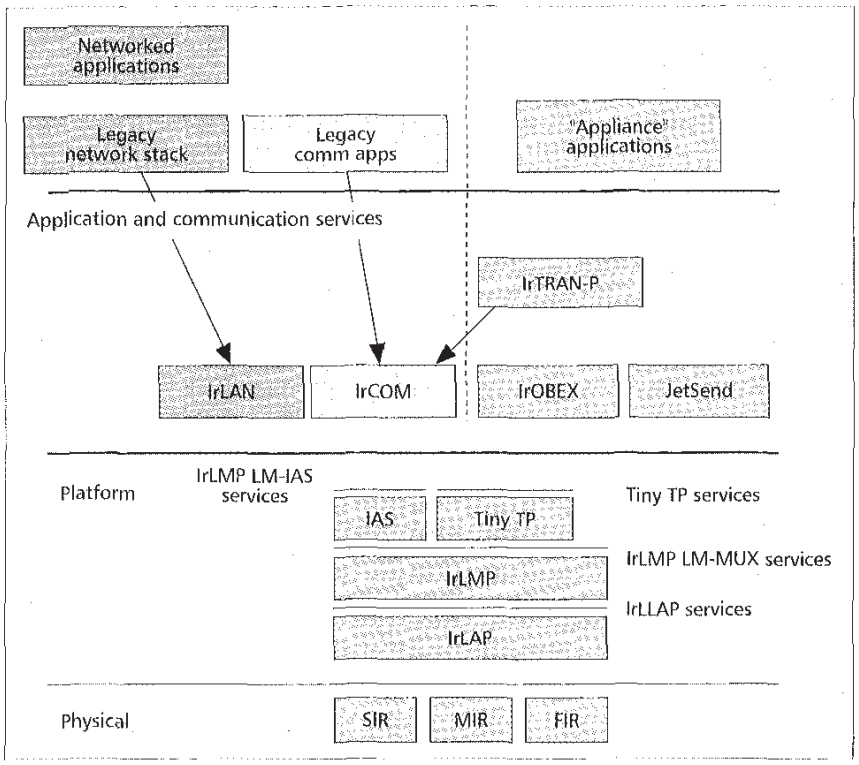
Prior to June 1993, a number of noninteroperable single-vendor proprietary schemes for infrared data communications existed. There was considerable risk that the marketplace for short-range wireless infrared communications would fragment around a number of proprietary schemes, all of which would individually 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

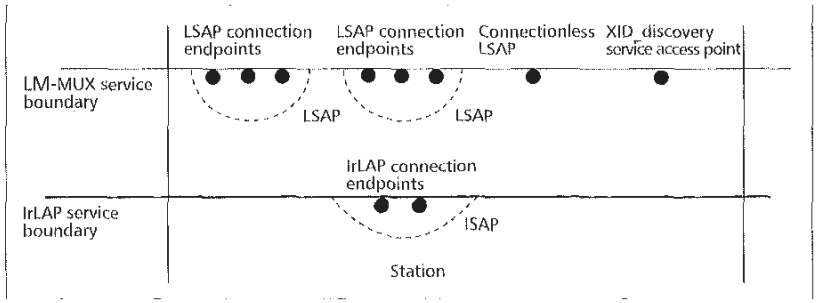
It was not enough merely to define a communications platform. In order to promote interoperability between applications, it was essential to develop specifications for the application services and the application protocols that support them. Hence, work has also progressed to define application

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.

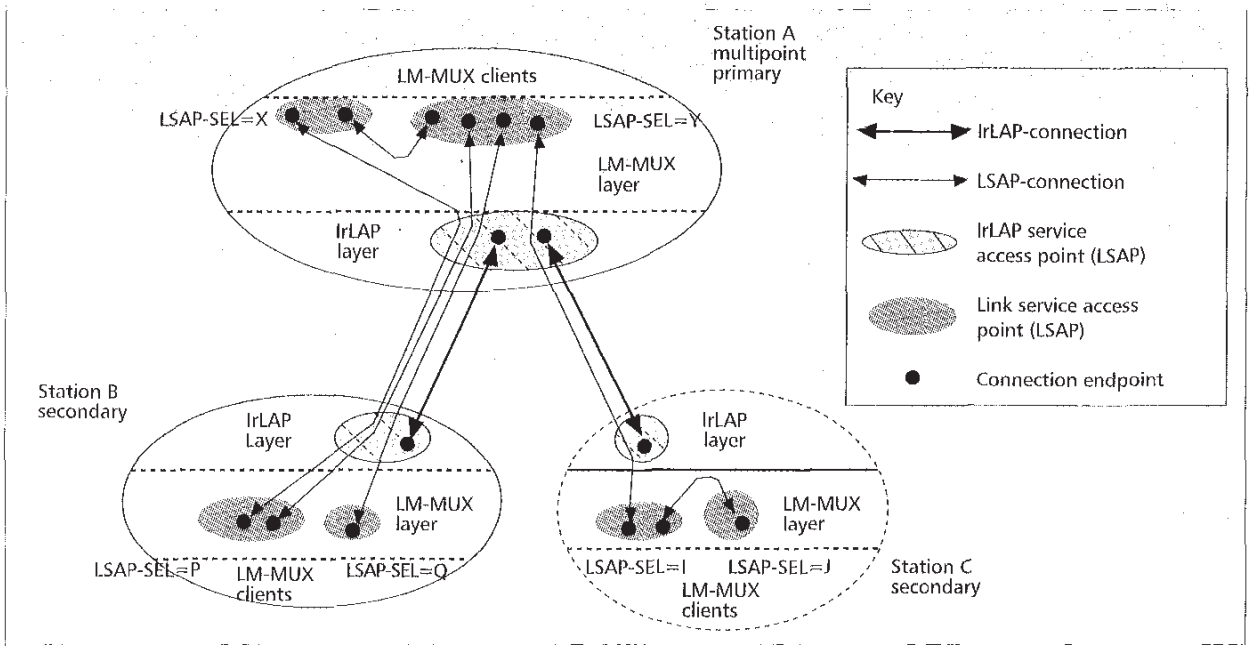
IrDA Connection Model

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

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.