The IrDA Platform

Stuart Williams and Iain Millar
HP Laboratories, Bristol

Abstract

For almost the past two years the Infrared Data Association has been working to establish an open standard for short range directed Infrared data communications. We are now at a time where the technologies developed within this forum are finding their way into the marketplace. Whilst there has been a high level of multivendor participation and collaboration in the establishing the base level IrDA standards to date very little overview material has been published. This paper provides an introduction to the IrDA's mission and to the technologies that its members have developed. What the IrDA has specified to date is very much a platform. As a platform it meets the key goals of low-cost and multivendor interoperability. It also provides a rich set of ease of use features that will enable multiple applications to concurrently share access to an infrared connection between a pair of devices.

1. Introduction

Since its formation in June 1993 the Infrared Data Association (IrDA) has been working to establish an open standard for short range infrared data communications. At the time of its formation there were a number of vertical, non-interoperable infrared communications technologies. Today IrDA is a strong contender for anyone considering adding infrared data communications to their product. Indeed, whilst supporting their own legacies, vendors who have been offering infrared solutions for years are embarked on the transition to an IrDA based solution.

The key goals for the IrDA are interoperability, low cost, and ease of use. Interoperability is addressed through the creation of an open standard with wide spread, multi-vendor support[1].

Low cost refers to the marginal cost of adding an IrDA interface to products in high volume manufacture. For the most part the cost of adding the digital logic required to provide an IrDA interface is regarded as negligible. The few thousand gates that it takes to implement even the recent higher speed proposals are regarded as coming for free in an environment where ASIC functionality is limited largely by pin-count rather than gate utilisation. This leaves the marginal cost of adding the optoelectronic transceiver which is estimated as \$2-\$3 and is set to fall further in future with the availability of transceiver modules from optoelectronic suppliers.

Lastly there is ease of use. The IrDA usage model is for short range directed communication link that supports ad-hoc point-and-shoot and place-and-play communications. The nominal operating envelope is a 1m cone with 15 degree half-angles. One of the IrDA frequently asked questions over the past year has been "How do I aim my printer?" The point being that it is all very well to be able to point a PDA at a printer, but it is not really tenable for a printer to sprout legs and point back. Whilst the term "directed" is used to describe an IrDA system, it would be unfair to suggest that it requires highly accurate alignment. Indeed the physical specification allows for more omni-directional behaviour at ranges of less than 1m.

The IrDA system design, which is the focus of the bulk of this paper, is also a significant factor in establishing IrDA platforms as easy to use. Users of conventional communications applications have had to deal with having the correct cables to connect a computer or terminal to a peripheral such as a printer or a modem. They have had to do battle with baud rates, and bits per character and parity. They have also had the responsibility of ensuring that the correct software was loaded at opposite ends of the communications channel.

Whilst the IrDA aims to replace the serial cable for ad-hoc peripheral connection, it also aims to add ease of use features that enable applications to identify peer entities with which they can communicate. Thus a printing subsystem; a file sharing client; a calendar management application; a business card exchange utility... can all identify and locate matching peer entities in order to make use of their services.

The IrDA chose to base its initial standards on a 115kbit/s UART based physical layer that had been developed by Hewlett-Packard (HP-SIR) [1] [2], and an HDLC based Link Access Protocol (IrLAP) originally proposed by IBM [3] [4] [5].

During the course of its first year the need to multiplex multiple application-to-application streams over a single IrLAP connection was identified and with it the need to provide a means for locating and identifying the function of application entities offerings services over an IrDA interface. These needs led to the development of the IrDA Link Management Protocol (IrLMP) [6].

This paper provides an introduction to the services provided by a IrDA platform. These are services upon which new families of Infrared aware applications will be build. End users will not be tied to either applications or platforms from a single vendor.

2. IrDA System Overview

The IrDA Architecture in Figure 2.1.

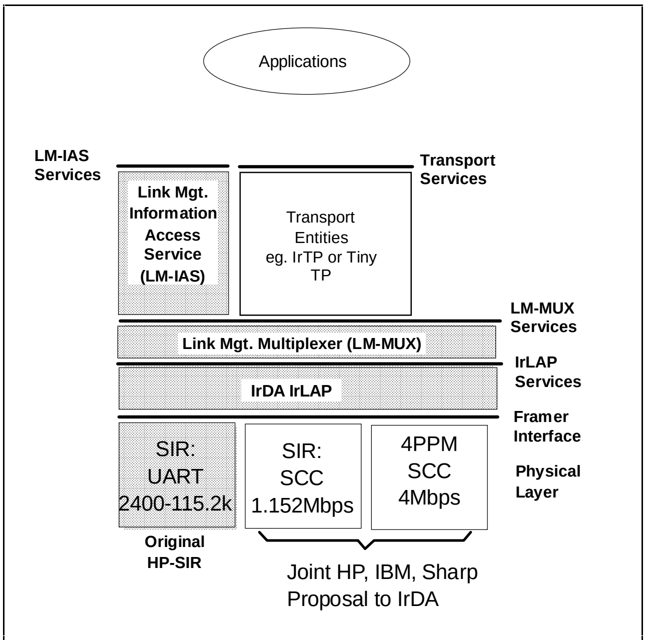

Figure 2.1 IrDA Architecture

There are now three components to the physical IrDA layer[2]:

[1] Including several major manufacturers of computers, PDAs, printers, modem and mobile phones; computer software companies; PTTs and component vendors

[2] The 1.152Mbps and 4Mbps physical layers are currently part of a proposal to the IrDA by HP, IBM and Sharp. It is anticipated that this proposal will be adopted in early 1995.

1. The original 2400bps-115.2kbps HP-SIR [1] based scheme using a conventional UART with character stuffed packet framing

2. A 1.152Mbps scheme that retains the same modulation scheme, but uses a synchronous communications controller and conventional HDLC bit stuffing [7].

3. A 4Mbps scheme that uses a 4PPM modulation scheme and frames packets with a sequence of code violations [7].

From the point of view of the Link Access Protocol (IrLAP) [5], the recent 1.152Mbps and 4Mbps extensions are regarded merely as extra speeds that may be negotiated when a device-to-device connection is established. All three physical layer schemes are designed to have a range of 1m at off axis angle of up to ±15 degrees. In practice, due to component tolerancing, on-axis ranges can be substantially greater, and satisfactory operation can be achieved at off-axis angles of 30 degrees or more.

The Link Access Protocol (IrLAP) is a variation of multi-drop HDLC [3]. It provides facilities for:

1. Controlling Hidden Terminal problems

2. Device Discovery

3. Device-to-device connect/disconnect and QoS negotiation

4. Data Transfer.

IrLAP is an asymmetric protocol and uses HDLC in its normal response mode (NRM). This means that once an IrLAP connection has been established, one station becomes a primary whilst the other becomes a secondary. In the context of a point-to-point connection there is very little difference between the behaviour of primary and secondary stations. However, as we shall see, IrLAP has a the potential to be extended to provide point-to-multipoint device-to-device connectivity. In this case a single primary device would be able to communicate with several secondary devices, but the secondary devices will not be able to communicate directly with each other.

The Link Management Protocol (IrLMP) [6] consists of two parts, a connection oriented multiplexer (LM-MUX) and a directory service (LM-IAS). With the exception of the directory service itself, there are no fixed addresses within the IrDA architecture. Device addresses are chosen at random and exchanged during IrLAP discovery. Address space collisions are resolved by the device that initiates discovery. Likewise 'port' space above LM-MUX is dynamically assigned. The LM-IAS directory service then serves as a means to identify the application services present within a device and the addressing information required establish contact between application peers.

2.1 Addressing

Within the basic IrLAP/IrLMP IrDA platform there are three levels of addressing:

1. Device Addresses: 32-bit randomly chosen identifiers exchanged between devices during IrLAP/IrLMP device discovery.

2. IrLAP Connection addresses: 7 bit HDLC secondary addresses assigned to a secondary device by the primary during IrLAP connection establishment and used for the duration of that connection.

3. IrLMP Multiplexer connection addresses: Logically an LM-MUX service access point is addressed by the concatenation of a 32-bit device address and an 8 bit multiplexer port selector. Once an IrLAP connection is established the IrLAP connection address serves as a synonym for the device address. A multiplexer connection is labelled by the addresses of the LM-MUX service access points at either end of the connection[3]

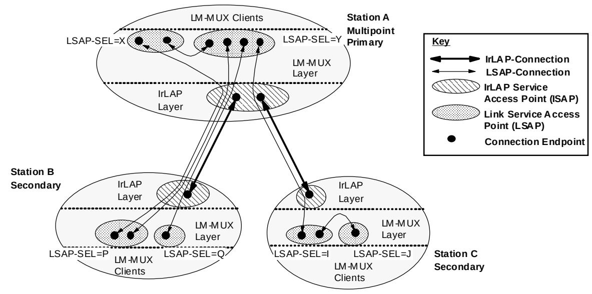

Figure 2.2 IrLAP and IrLMP Connections

The relationship between IrLAP connections, LM-MUX connections/connection end-points and LM-MUX service access points is shown in Figure 2.2

2.2 Link Access Protocol (IrLAP)

IrLAP, the IrDA Link Access Protocol [3], is based on HDLC operating in the Normal Response Mode (NRM) [3]. Typically this mode of operation has been used on multi-drop serial lines between say a terminal controller and a group of terminals sharing the serial line. The terminal controller acts as a primary and regularly polls each of the attached terminals. There are two attractive artefacts to this behaviour in the context of directed short range infrared communications:

1. Once the device-to-device connection has been established then in the absence of aberrant behaviour, access to the shared media is contention free.

2. The constant reversal of the 'line' due to the polling mechanism acts as a beacon to indicate to other devices that approach and active link that the media is in use.

In today's world of peer-to-peer communication a master/slave protocol may seem something of an odd choice. However, it is reasoned that for directed communication the majority of real life scenarios can be addressed by the provision of a single point-to-point link between a pair of devices. In this context the difference between primaries (masters) and secondaries (slaves) becomes moot. Indeed, all differences between primary and secondary are masked by the IrLMP layer above IrLAP so that applications need not be aware of this minor asymmetry.

IrLAP operates in two main modes:

1. Contention Mode: Procedures that occur in contention mode are device discovery, address conflict resolution and connection establishment. All contention mode traffic occurs at 9600bps over the HP-SIR/UART physical layer.

2. Connection mode is entered at the point that an IrLAP connection is established. The communication speed is changed to the rate negotiated in the connection setup messages.

Connection mode traffic has priority over contention mode traffic.

Contention Mode MAC Rules

Once the IrDA stack in a device has been enabled it must sense the media for a minimum of 500ms. Also, a device must sense the media as idle for at least a further 500ms prior to repeating a

[3] This is similar to a TCP/IP connection being labelled by the concatenation of IP address and port number at each end of the connection. Also this leads to the restriction that there may be at most one TCP/IP connection between the same pair of TCP ports. A similar restriction applies to LM-MUX connections.

contention mode procedure. 500ms is absolute upper bound on the time that either a primary or secondary station may retain the right to transmit frames. Shorter intervals may be negotiated during connection establishment.

Connection Mode MAC Rules

Once and IrLAP connection has been established access to the media is mediated by the exchange of a token (the P/F bit in the HDLC control field). Both primary and secondary stations monitor the exchange of this token and provide status indications upward in the event of deteriorating link quality or loss of connection. A station may transmit a number of frames, up to a limit bounded by the negotiated window size, maximum data packet size and the overriding turnaround time for returning the token to its peer.

2.2.1 Hidden Terminal Management

We have already touched on the hidden terminal management capabilities of IrLAP. By placing an absolute upper bound on the link turn around time it is possible to ensure that each end of an IrLAP connection makes regular transmissions that act as a beacon to indicate that there is an established IrLAP connection in the vicinity. Hidden terminals (hidden from one end of the connection) remain silent in the presence of an active IrLAP connection.

2.2.2 Device Discovery and Address Conflict Resolution

An IrDA device address is a 32-bit identifier that a station randomly assigns to itself. Within the relatively small extend of the 'reachspace' the probability of two or more devices choosing the same address is relatively small[4]. IrLAP provides the facility for its client (IrLMP) to instruct devices with colliding addresses to select new device addresses. IrLMP drives the address resolution process by making a single attempt to resolve each address conflict.

Device discovery takes place in contention mode. Device discovery is used to retrieve *<DeviceAddress><DeviceInfo>* tuples from devices in the vicinity.

<DeviceInfo>=<ServiceHints><DeviceNickName>

Service hints is an extensible bit map that provides for a very coarse characterisation of the services offered by the device: currently defined hints bits can specify a device as offering the services of a PDA, a Computer, a Printer, a Modem, a Fax, LAN Access, Telephony, or a File Servers. It is incumbent on the designers of an application service to state what hints bits will be set if an instance of that service is available with a station.

The Device Nickname is a short name that may be presented to the user in order to select between two otherwise identical devices.

A slotted discipline is used for discovery. The station initiating discovery issues a request that specifies the use of 1, 6, 8 or 16 slots. Stations receiving this request randomly select a slot between 0 and the specified upper bound minus 1 in which to make their response. The initiating station then 'calls' out each slot in turn, marking its start with a packet that encodes the slot number slot being polled. Finally the initiating station marks the end of discovery with a final packet that includes the stations own discovery information. Device discovery is illustrated in Figure 2.3.

[4] Actually the generation of good quality random numbers is an issue here. If the device address of two devices do collide and they both employ the same pseudo random sequence then their next choice of device address may very well collide as well. Some genuinely random process must be incorporated in the process of assigning a device address.

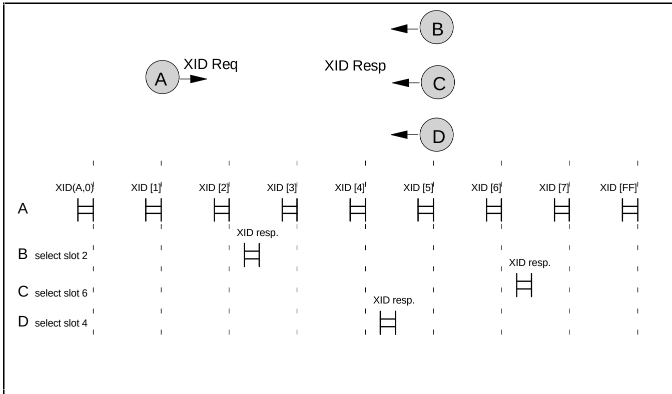

Figure 2.3 Eight Slot Device Discovery

2.2.3 Connection Establishment

An IrLAP connection is initiated by the transmission of a Set Normal Response Mode (SNRM) frame, using Contention mode MAC rules, by the station that will initially become the primary station. The SNRM frame contains a number of negotiable QoS parameters, including: data rate capabilities; turnaround requirements negotiable from 500ms down to 50ms; maximum data packet size; receiver window size, 1 through 7; Link disconnect and threshold times to deal with packet loss. The SNRM also contains a device address (retrieved by a recent discovery operation) and assigns a 7-bit connection address for use during the connection.

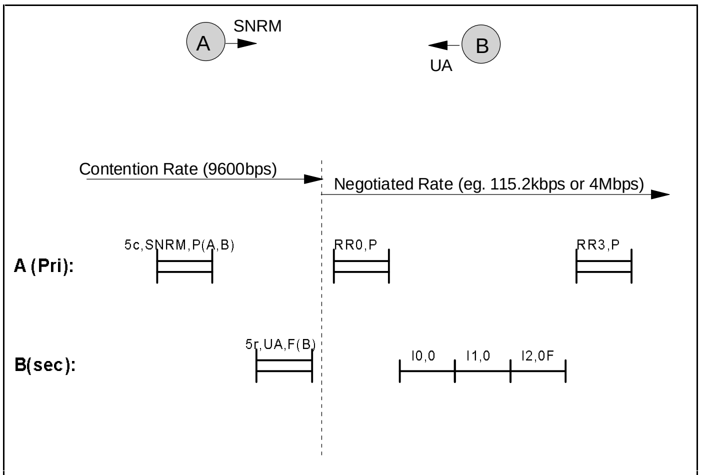

Figure 2.4 IrLAP Connection Establishment

The station addressed by the SNRM responds to the SNRM with an Unnumbered Acknowledge frame, also at the 9600bps contention rate, that contains the results of the negotiation process. At this point both stations apply the newly negotiated parameters. The primary immediately sends an Receiver Ready frame to indicate to the secondary that it is now using the negotiated communication parameters[5].

2.2.4 IrLAP Data Transfer Services

Once an IrLAP connection has been established then the data transfer service between devices is similar to that provided by HDLC operating in Normal Response Mode. It provides for the reliable sequenced exchange of packets. Provision is also made for the transmission of Unnumbered Information (UI) frames. These frames are sent at the negotiated data rate and have

[5] Careful timing is needed here to ensure that both the primary and secondary have applied the new parameters prior to the transmisson of the RR.

priority over the exchanged of sequence information (I) frames, but they are also subject to loss without recovery.

2.3 Link Management Protocol (IrLMP)

The IrDA Link Management Protocol (IrLMP) [6] provides two distinctly different types of services. Firstly it provides a level of connection oriented multiplexing (LM-MUX) on top of IrLAP. Secondly it provides an Information Base that hold details of the application entities present in the local station that are current offer services to other IrDA devices. Objects in this information base carry the essential addressing information necessary establish communication with the corresponding application entities. Access to this Information Base is provided by an Information Access Server and an corresponding Client. Collectively the Information Base, the Server and the Client provide an Information Access Service (LM-IAS). Both LM-IAS Client and Server entities are LM-MUX clients.

2.3.1 The Multiplexer

The LM-MUX provides a simple level of switching over the top of an IrLAP connection. It also hides the master/slave nature of IrLAP from the application and provides a symmetrical set of services to IrLMP clients.

The key goal here is to allow multiple independent sets of application entities to share access to the underlying IrLAP connection. This is increasingly important as the ability of portable platforms to multi-task improves. Also the mix of applications running on a portable platform tends to be chosen by the end-user from a potentially varied list of vendors. With this degree of 'end-user' integration it is simply not tenable to offer a solution that does not allow applications to share access to the IR media. For example: consider a file sharing application that allows portable device to access files on a desktop machine. This may result in a relatively long lived connection between the devices and the end user may not really be conscious of its existence. It would not be acceptable to have to shut down the file sharing software in order to then gain access to say an E-mail, printing or Fax services.

The functionality provided by LM-MUX has already been presented in Section 2.1 and Figure 2.1. LM-MUX also provides a device discovery operation that combines IrLAP device discovery and address conflict resolution into a single operation.

The introduction of a simple multiplexing function above the reliable device-to-device data transfer service of IrLAP does introduce one problem:

In general, multiplexing LM-MUX channels over a single IrLAP connection can lead either to data loss or deadlock. To illustrate consider a pair of peer application entities **A** and **B** connected by two LM-MUX connections. One LM-MUX connection is used to exchange data, while the other is used to send control. **A** sends on both connections and **B** receives on both connections.

Consider what happens if **A** sends a large amount of data to **B** while **B** attempts to read first from the control channel and then from the data channel. The following code fragments illustrate this behaviour[6]:

```
/* Behaviour of sender A */
for(...) {
    .
    .
  res = send(data_fd, data, sizeof(data));
    .
    .
}
res = send(control_fd, ctrl,sizeof(ctrl);
```

[6] Assuming blocking send and recv operations

```
/* Behaviour of receiver B */
  recv(control_fd, ctrl,sizeof(ctrl));
  recv(data_fd, data, sizeof(data));
```

Data inbound for **B** is not being read. At some point buffer space holding inbound data for **B** becomes exhausted. We could allow the IrLAP flow-control mechanism to pause the sending of data by **A**. However, if we do that there is then no way that the control information from **A** to **B** can now be sent to allow **B** to progress to reading the data. The system is deadlocked. This is a general problem for any system that multiplexes channels over a reliable channel.

Alternatively, we avoid this deadlock situation by allowing IrLMP LM-MUX to discard inbound data that it cannot deliver. However this destroys the reliable delivery property that IrLAP gives us.

Allowing the deadlock possibility is by far the greater of these two evils, so the designers of IrLMP took the view that LM-MUX may discard inbound data that it is unable to deliver. The LM-MUX data transfer service is therefore best effort rather than reliable. The only possible cause of packet loss in these circumstances is inbound congestion within an LM-MUX channel. This inbound congestion may becompletely avoided by the inclusion of a flow-control mechanism within the channel between application entities.

The IrDA offers two suggestions for addressing this problem: a variant of the ISO 8073 Class 2 Transport Protocol [8] named IrTP [9]; and a bared credit based flow-control scheme dubbed Tiny TP [10].

As an alternative to flow control it should also be noted that designers of LM-MUX clients may choose instead to implement a retransmission scheme to recover from data lost due to inbound congestion; and in some circumstances the loss of data could simply be tolerated (e.g. playback of audio data).

Since these three alternatives exist, the IrDA has not mandated the use of any one particular.

There is one last facet of the multiplexer that is worthy of note. The negotiation present within IrLAP means that there is a deterministic upper bound to the time between the reversals of the underlying IrLAP connection. Some application designers may wish to exploit this deterministic behaviour however the presence of multiplexed streams provided by IrLMP obscures any guarantees provided by IrLAP. LM-MUX provides a mode of operation that grants exclusive access to the underlying IrLAP connection to just one LM-MUX connection.

2.3.2 Information Access Services

So far we have described a relatively straight forward communications mechanism that supports multiplexed communication channels between a pair of devices. A key goal for the IrDA has been ease of use. Previous IR solutions have had a tendency to disappoint users because the burden of ensuring that compatible peers application entities are active at each end of the link has fallen on the end-user. The LM-IAS within IrLMP provides the means for an application entity to identify and locate a compatible peer entity.

The LM-IAS Information Base contains a number of simple objects. Each object is an instance of a given class and contains a number of named attributes.

The class of an object implies the nature of the application entity that it represents, the data transfer method(s), the semantics of the information stream exchanged between peer entities etc. It also scopes the semantics of the attributes contained within instances of the given object class.

Both class names and attribute names my be up to 60 octets long. Since the meaning of an attribute is scoped by the class of the enclosing object there is no strict requirement to administer the attribute name space. Object class names do need to be administered, however it is intended that with such a large name space some sensible conventions will ensure that class name collisions do not occur. For example, Object Class definitions defined by the IrDA all start with the root “IrDA:”.

Whilst in general attributes are scoped by the class of the enclosing object some attributes are of such general utility that they may be regarded as having global scope. In a formal sense this requires that they are identically defined in all object classes that adopt the use of such global attributes. Typically global attributes arise in order to express an address within the IrDA environment. For example, IrLMP defines the attribute “IrDA:IrLMP:LsapSel” to identify the LM-MUX service access point of a directly attached application entity. IrTP defines the attribute “IrDA:IrTP:TsapSuffix” to carry the portion of the Transport service access point address that extends the 32-bit device address. Likewise Tiny TP defines the attribute “IrDA:TinyTp:LsapSel”. Application entities advertise their accessibility via these mechanisms by the inclusion of the corresponding attribute.

There are three attribute value types:

- Integer: A 32 bit 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.

The IrDA requires that every IrDA compliant device provides a “Device” object that carries a long form of the device name and an indication of the version of IrLMP implemented on the device and the optional features that have been implemented. The long device name is useful as it allows names of up to 255 octets in length whereas the nickname exchanged during device discovery is restricted to 19 octets (<10 characters if Unicode is used to encode the nickname).

Access to a remote IAS Information Base is provided by a local IAS client entity that communicates with an IAS server entity on the remote device. The IAS server is statically bound to LSAP 0x00 on the multiplexer. This is the ONLY fixed address in the IrDA environment. All other application services are located by inspection of the Information Base. The IAS Client and IAS Server entities provide a number of querying operations on the Information Base. Get Value By Class is the only mandatory operation that both must support. This provides a ‘shot-in-the-dark’ mode of retrieving attributes from objects. The notion is that a client application entity knows what application service it seeks to make use of. For example a file sharing entity would be looking to make contact with a matching file sharing entity. It therefore knows the object class name of Information Base objects that represent such an entity and implicitly it knows the name and semantics of attributes attached to such objects. There is therefore little value in the application entity browsing the Information Base, it merely needs to attempt to retrieve known attributes from an instance of a known object class. This is precisely what Get Value By Class does.

The remaining optional IAS operations: Get Information Base Details; Get Objects; Get Value; Get Object Info and Get Attribute Names provide for richer interactions with the Information Base including the ability to browse the Information Base.

2.3.3 IrLMP Client Example

Consider a Fax Modem that offers independent data and control channels. The Fax modem advertises its data service by installing the following object in its local Information Base:

```
object 1 class FaxModemData {
  attribute IrDA:TinyTP:LsapSel =
        Integer(0x05);
}
```

The FaxModemData service makes use of Tiny TP and is accessible with an LM-MUX service access point selector of 5.

An client application entity that wishes to make use of the FaxModemData service performs the following operations:

```
IasValue *lsapSel;
DiscoverList *dl;
DeviceAddress *da;
FILE *fp;

/* Device Discovery */
dl = LM_DiscoverDevices(slots);

fp = NULL;

while (fp == NULL) {

/* Search Hints for Fax Device */
  while (dl != NULL) {
    if(dl->deviceInfo.hints & FAX_MASK){
      da = dl->deviceAddress;
      break;
     }
     dl = dl->next;
    }

/* Check for end of Discovery List */
  if(dl == NULL)
    break;

/* Read the LM SAP Selector */
  lsapSel = LM_GetValueByClass(
            da,"FaxModemData",
            “IrDA:TinyTP:LsapSel");

/* Connect if we got an LM-MUX SAP Sel */
  if(lsapSel != NULL) {
    fp = TinyTP_Connect(lsapSel,...)
  }
  dl = dl->next;
}
```

2.4 Upper Layers

We have already made mention of both IrTP and Tiny TP. Both of these may still be regarded as part of the plumbing as they form part of the conduit between peer application entities.

- IrTP [9] is based on the ISO 8073 Transport Protocol Class 2 [8]. The main task of the IrTP specification is to describe the interpretation of Transport Service Access Point Addresses in an IrDA context and to provide a mapping from the ISO Network Service primitives used by 8073 to IrLMP LM-MUX service primitives.

  IrTP provides:

  Per transport connection flow control

  Segmentation and reassembly of arbitrary sized PDUs

  Graceful disconnect

  More multiplexing.

- Tiny TP [10] defines a credit based flow-control scheme and relies on LM-MUX for multiplexing.

  Tiny TP provides:

  Per transport connection flow control

  Segmentation and reassembly of arbitrary sized PDUs

There is also scope for the development of transport protocols that exploit the deterministic behaviour provided by exclusive mode.

Members of the IrDA are currently working to define an mechanism for application level object exchange, OBEX [11]. Objects may be records from Personal Information Managers, diary entries, business cards etc.; Word processor, spreadsheet other traditional types of file; or other parcel of information e.g. an information hunting robot launched into a network from say a PDA attached via IR to a payphone in an airport lounge.

2.5 Application Interfaces

Widespread implementation of the IrDA platform services described previously and the availability of consistent application programming interfaces (APIs) is the key to creating a market for IrDA aware applications.

The IrDA community is also currently working on two types Application Programming Interfaces (APIs) for IrDA. Legacy communications APIs and native IrDA APIs.

Legacy APIs

There is a general perception of Infrared as merely a cable replacement technology. From the preceding discussion it should be apparent that the inclusion of IrLMP, particularly LM-IAS, makes it much more. Nevertheless this perception persists and there is a desire to be able to run legacy serial and parallel port communications applications over Infrared in much the same way that terminal emulation applications were transitioned onto LANs from RS232 cables.

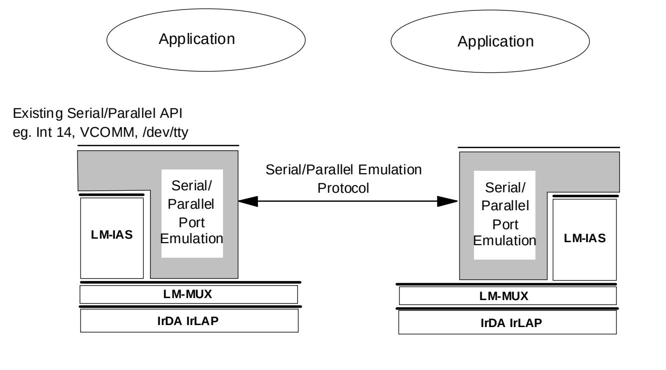

Figure 2.5 Legacy Communications API Support

The IrDA has a Working Group known as IrCOM that is working on the emulation of the legacy communication interfaces typically provided by serial/parallel device drivers.

Native APIs

New applications will take full advantage of the potential that IrLMP offers to enable compatible application peers to identify and locate each other. This requires a native API for IrDA that exposes the full functionality of IrLMP, IrTP and Tiny TP to the application programmer.

IrDA members are interested defining a Winsock 2 Service Provider and API semantics [12][13]. Mapping the LM-IAS services into Winsock 2 is likely to prove a particular challenge!

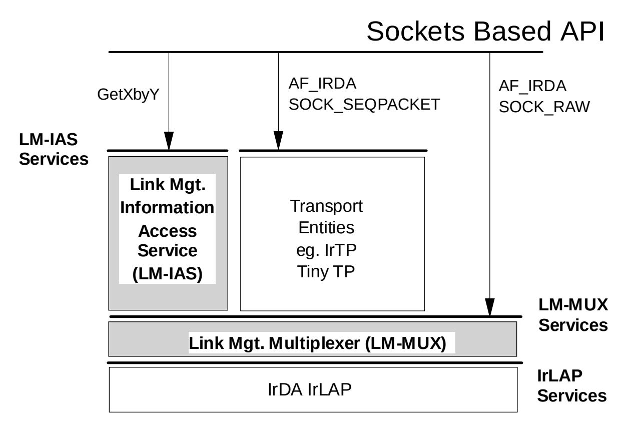

Figure 2.6 Sockets Based API

3. Application Services

With a stable platform on which to build application and application service designers can rapidly populate the space above the base platform. Some services will come about as the result of either open or closed collaborations, others will be the work of an individual or a single organisation. It is incumbent on application service designers to specify:

1. The IrLMP service hints that will be set if an instance of the service is being advertised.
2. An object class to carry the parameters essential to establish communication between peers.
3. Data transfer methods: i.e. raw, over IrTP, over Tiny TP or some other method specified in the service definition.
4. The application level protocol.

The degree to which the application service designer makes this information open is a matter of judgement for them. They may choose to seek endorsement from the IrDA; they may simply publish the specification of their service. Alternatively it may be regarded as proprietary and closed.

4. Summary

The future of short range directed infrared data communications looks bright!

With some 80 member companies the technology developed within the IrDA will soon be available on just about every significant mobile computing platform. End-users will be able to casually exchange, print and share information from whole documents to snippets from a diary or a business card. They will be able to do this without the hassle of needing to have the right cables and without having to do combat with pages of configuration and setup dialogues commonplace with serial and networked communication. Within the next year products with high speed IrDA interfaces will be readily available in the marketplace.

The ease of use that will be characteristic of IrDA applications is largely due to the device discovery and QoS negotiation facilities in IrLAP and the Information Access Services specified in IrLMP.

5. 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, December 24, 1991.

[2] Infrared Data Association, "Serial Infrared (SIR) Physical Layer Link Specification", Version 1.0, April 27, 1994.

[3] International Organisation for Standardisation (ISO), "High Level Data Link Control (HDLC) Procedures - Elements of Procedures", ISO/IEC 4335, September 15, 1991.

[4] T. F. Williams, P. D. Hortensius and F. Novak, "Proposal for: Infrared Data Association Serial Infrared Link Protocol Specification", Version 1.0, IBM Corporation, August 27, 1993.

[5] Infrared Data Association, "Serial Infrared Link Access Protocol (IrLAP)", Version 1.0, June 23, 1994.

[6] Infrared Data Association, "Link Management Protocol", Version 1.0, August 11, 1994.

[7] HP/IBM/Sharp, "PROPOSAL: Fast Serial Infrared (FIR) Physical Layer Link Specification", Proposal to the Infrared Data Association, September 13, 1994.

[8] International Organisation for Standardisation (ISO), “Information technology - Telecommunications and information exchange between systems - Open Systems Interconnection - Protocol for providing the connection-mode transport service”, ISO/IEC 8073, December 15, 1992.

[9] Infrared Data Association, “Use of ISO 8073 as an IrDA Transport Protocol”, Version 1.0, August 12, 1994.

[10] Infrared Data Association, “TinyTP: A Flow-Control Mechanism for use with IrLMP”, Version 0.1a, January 16, 1995.

[11] Infrared Data Association, “Object Exchange Protocol”, Version 0.1a, January 4, 1995.

[12] Winsock 2, “Windows Sockets 2 Application Programming Interface: An Interface for Transparent Network Programming Under Microsoft Windows”, Revision 2.0.6, February 1, 1995.

[13] Winsock 2, “Windows Sockets 2 Service Provider Interface: A Service Provider Interface for Transparent Network Programming Under Microsoft Windows”, Revision 2.0.6, February 1, 1995.

6. IrDA Contact Information

Further information about the IrDA and copies of the IrDA Standards may be obtained from:

John Laroche
Executive Director
Infrared Data Association
P.O. Box 3883,
Walnut Creek
Califonia
USA 94598

Tel: (510) 943 6546
Fax: (510) 943 5600

E-mail: jlaroche@netcom.com

7. Author Information

The authors may be contacted at

Hewlett-Packard Laboratories, Bristol
Filton Road
Stoke Gifford
Bristol, BS12 6QZ
United Kingdom

Tel: +44 (117) 979 9910
Fax: +44 (177) 922 8920

E-mail: skw@hplb.hpl.hp.com
im@hplb.hpl.hp.com