HP-JetSend Network ARS

Version 3.30

Allen Baker

Hewlett-Packard Company
INSO R&D Lab

November 11, 1996

Table of Contents

1. Revisions 4
   1.1 Version 3.30 (11nov96) 4
   1.2 Version 3.21 (15aug96) 4
   1.3 Version 3.20 (23may96) 4
   1.4 Version 3.10 4
   1.5 Version 3.03 4
   1.6 Version 3.02 4
   1.7 Version 3.01 5
   1.8 Version 3.00 5
   1.9 Version 2.10 5
   1.10 Version 2.00 5
      1.10.1 What's New? 5
2. Patents 6
3. Introduction 7
4. References 8
5. Architecture 9
   5.1 Generalized Architecture 10
      5.1.1 HP-JetSend Session Protocol layer (OSP) 10
         5.1.1.1 HP-JetSend Session Protocol 11
            5.1.1.1.1 HP-JetSend Addresses 11
               5.1.1.1.1.1 HP-JetSend Multi-Network form (OMN form) 11
                  5.1.1.1.1.1.1 Syntax 11
                  5.1.1.1.1.1.2 Examples 12
               5.1.1.1.1.2 Universal Resource Locator form (URL form) 12
                  5.1.1.1.1.2.1 Syntax 12
                  5.1.1.1.1.2.2 Examples 12
               5.1.1.1.1.3 Multi-Port Network-Parallel Gateway Addressing 12
            5.1.1.1.2 HP-JetSend Session 13
            5.1.1.1.3 HP-JetSend Session Channels 15
            5.1.1.1.4 Shared OSP 15
      5.1.2 HP-JetSend Transport Independent API (TIAPI) 16
      5.1.3 Host Name Resolver (HNR) 17
      5.1.4 Reliable Message Transport Protocol (RMTP) 18
         5.1.4.1 Service Objectives 18
         5.1.4.2 Relation to other protocols 18
      5.1.5 Supported Transport and Network Protocols 19
   5.2 Fall 97 Implementation Architecture 19
      5.2.1 HP Scanner Platforms (Verona) 20
      5.2.2 HP Printer Platforms (MoBY, Monarch) 21
      5.2.3 Desktop Platforms (Windows 95 and Windows NT) 22
6. Appendix A (Addressing Considerations) 23
   6.1 TCP/IP 23
      6.1.1 IP Address 23
      6.1.2 Host Name 23
      6.1.3 Recommendation 23
   6.2 SPX/IPX 23
      6.2.1 IPX Address 23
      6.2.2 User Names 23
      6.2.3 Real Names 23
      6.2.4 Recommendation 23
   6.3 IrDA (Infrared) 24
   6.4 FAX 24

6.4.1 FAX Gateway 24
6.4.2 FAX Appliance 24
6.4.3 E-mail Gateway 24
6.5 Sub-Addressing 24
7. Appendix B (Dynamic Address Assignment) 25
7.1 TCP/IP 25
7.1.1 Bootp 25
7.1.2 DHCP (Dynamic Host Configuration Protocol) 25
7.1.3 IPCP (Internet Protocol Control Protocol) 25
7.2 SPX/IPX 25
7.2.1 NetWare 25
7.3 IrDA (Infrared) 25
8. Appendix C (Name Services) 26
8.1 TCP/IP 26
8.1.1 DNS (Domain Name Service) 26
8.1.2 WINS (Windows DNS) 26
8.2 SPX/IPX 26
8.2.1 Bindery 26
8.2.2 NDS (NetWare Directory Services) 26
8.3 IrDA (Infrared) 26
9. Appendix D (Device Browsing / Discovery) 27
9.1 Browsing and Discovery Methods 27
9.1.1 Using Existing Name Services 27
9.1.1.1 Pros 27
9.1.1.2 Cons 27
9.1.2 Using Broadcast Messages 27
9.1.2.1 Scope 27
9.1.2.2 Pros 27
9.1.2.3 Cons 27
9.2 Browsing and Discovery Uses 28
9.2.1 FAX Gateway Discovery 28
9.2.2 Address Book Sharing 28

1. Revisions

1.1 Version 3.30 (11nov96)

- Changed IANA registered default port number from 9120 to the actually assigned 1782.
- Updated diagrams.
- Added context diagram.
- Separated scanner and printer implementation architecture sections.

1.2 Version 3.21 (15aug96)

- Improved consistency of terminology and improved some wording.
- Expanded HP-JetSend Address section.
- Added mention of multiple sessions to an OSP gateway.

1.3 Version 3.20 (23may96)

- Removed mention of a control channel within OSP.
- Updated overview of how OSP sets up a session.
- Removed details of how OSP sets up a session; they are covered in the OSP ERS.
- Replaced “OSP address” with “oipHandle” to bring terminology into line with OSP ERS.
- Changed title from HP-JetSend Network Architecture to HP-JetSend Network ARS.
- Added introduction to describe scope of document.
- Added references section.

1.4 Version 3.10

- Removed references to MLC as a network.
- Removed SPX and IPX.
- Replaced Message Protocols section with Reliable Message Transport Protocol (RMTP).
- Updated figures.
- Increased font size.

1.5 Version 3.03

- Removed paragraph that talks about TIAPI running on a separate thread.

1.6 Version 3.02

- Removed Issues, Requirements, and Goals sections. Any information that was in those sections that is part of an architecture description is moved to an appropriate architecture subsection.
- Finished architectural description of OSP.
- Changed DNS component to Host Name Resolver and made it include WINS along with DNS.
- Expanded TIAPI section.
- Changed “Reliable Datagram Protocol” section to “Message Protocol” section.
- Reformatted document.

1.7 Version 3.01

- Added discussion of OSP.

1.8 Version 3.00

- Removed ERS information. We will create separate ERSs for both the TIAPI and the OSP.

1.9 Version 2.10

- Added back datagram calls.
- Removed idea of connection type to select between TCP and R-UDP or between SPX and R_IPX etc. TIAPI now does best effort STREAM setup.

1.10 Version 2.00

- This document makes obsolete HP-JetSend Network Architecture, version 1.0; Randy Sartin; February 1, 1996.

1.10.1 What's New?

- The Connection Manager (CM) is now called the HP-JetSend Session Protocol layer (OSP)
- The Link Manager (LM) is now called the Transport Independent API (TIAPI).

2. Patents

The architecture described in this document is protected under the following international patents which are owned by the Hewlett-Packard Company:

1. DE69802792D1
2. DE69802792T2
3. DE69835314D1
4. DE69835314T2
5. EP0872991A2
6. EP0872991A3
7. EP0872991B1
8. EP0886411A2
9. EP0886411A3
10. EP0976058A1
11. EP0976058B1
12. US6202096 B1
13. US6502000
14. US6721286
15. WO1998047076A1

3. Introduction

An Architecture Reference Specification (ARS) specifies its target object from an internal perspective. The internal perspective is a view of the target object from the inside and does not include requirements specifications or the functional and interface specifications of the target object as a whole (the target object as a whole is described by its ERS). The internal perspective includes all of the target object's components and how they communicate with one another. Each component is identified and its functionality and interfaces are specified (to avoid redundancy, functionality and interfaces should be specified by a reference to the component's ERS if one exists). All communications among components are identified and explained. The data and signals being communicated as well as their purpose and the physical media over which the communication occurs should be specified in the ARS.

The target object for this ARS is the HP-JetSend network, or link component.

4. References

[1] Allen Baker. HP-JetSend Session Protocol ERS. Hewlett-Packard Company, INSO R&D. Boise, ID. 1996.

[2] Allen Baker. Reliable Message Transport Protocol ERS. Hewlett-Packard Company, INSO R&D. Boise, ID. 1996.

[3] Allen Baker. HP-JetSend Transport Independent API ERS. Hewlett-Packard Company, INSO R&D. Boise, ID. 1996.

5. Architecture

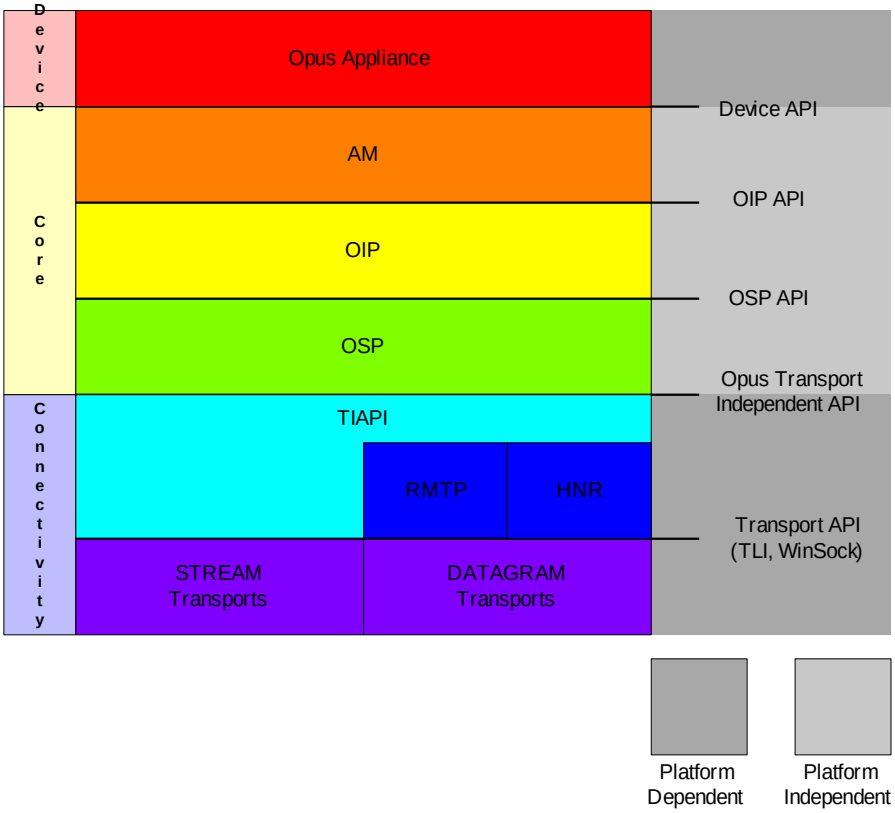


Figure 1

Figure 1 shows the components of the HP-JetSend network architecture within the context of the overall HP-JetSend architecture. The HP-JetSend network architecture includes the OSP, TIAPI, RMTP, and HNR components from Figure 1.

5.1 Generalized Architecture

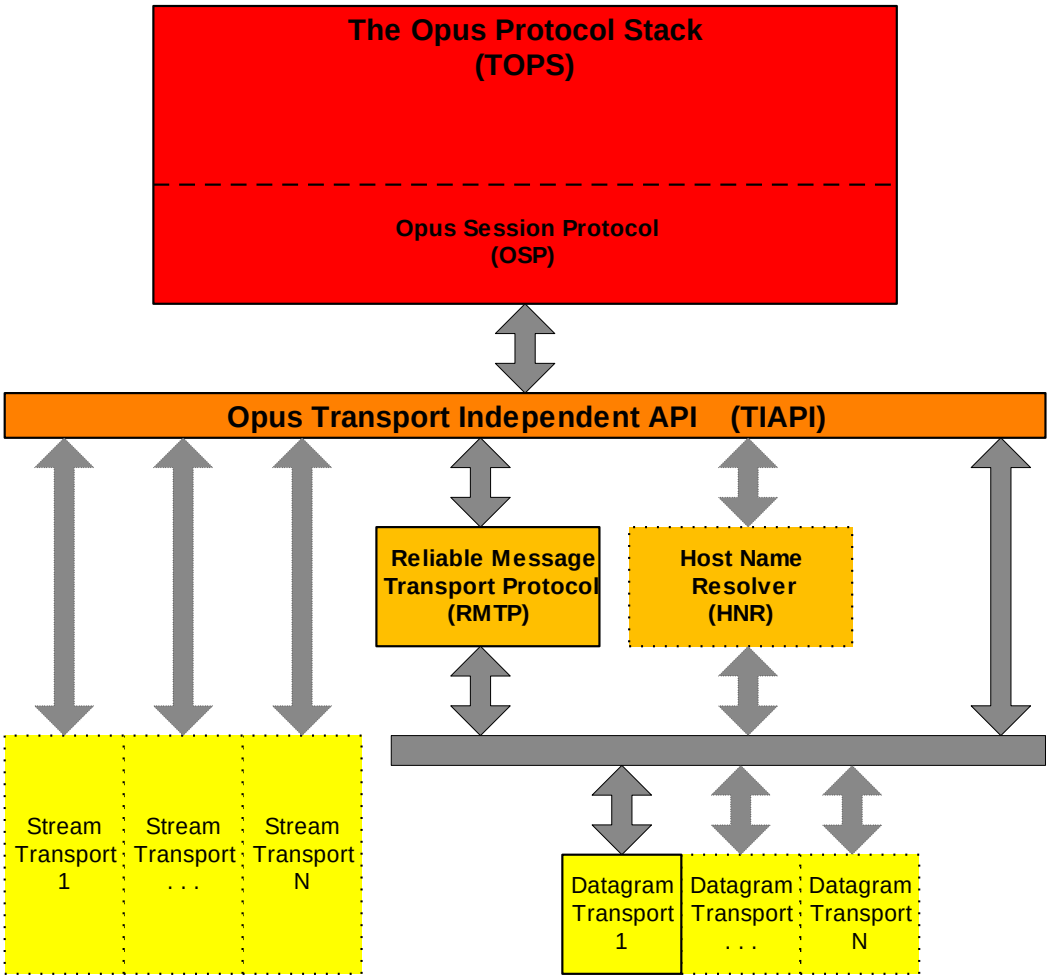


Figure 2

Figure 2 identifies the components of the HP-JetSend network architecture and their logical relationships to each other. Optional components are bordered with dotted lines. This Generalized Architecture section describes each of the components at an architectural level. Details about the interface (APIs, required system services etc.) to any of the components may be found in the External Reference Specification (ERS) that corresponds to that component. Details about the internal design of any of the components may be found in the Internal Reference Specification (IRS) that corresponds to that component.

5.1.1 HP-JetSend Session Protocol layer (OSP)

A complete description of the OSP interface and protocol is contained in [1].

The OSP is part of The HP-JetSend Protocol Stack (TOPS). TOPS is the portable, platform independent portion of HP-JetSend. OSP is the last place, heading out of TOPS towards the network, that platform independent code can exist. Platform independence in OSP is achieved by always coding OSP to the HP-JetSend Transport Independent API (TIAPI), regardless of platform. OSP provides session services to the TOPS layer immediately above it - the HP-JetSend Interaction Protocol Layer (OIP).

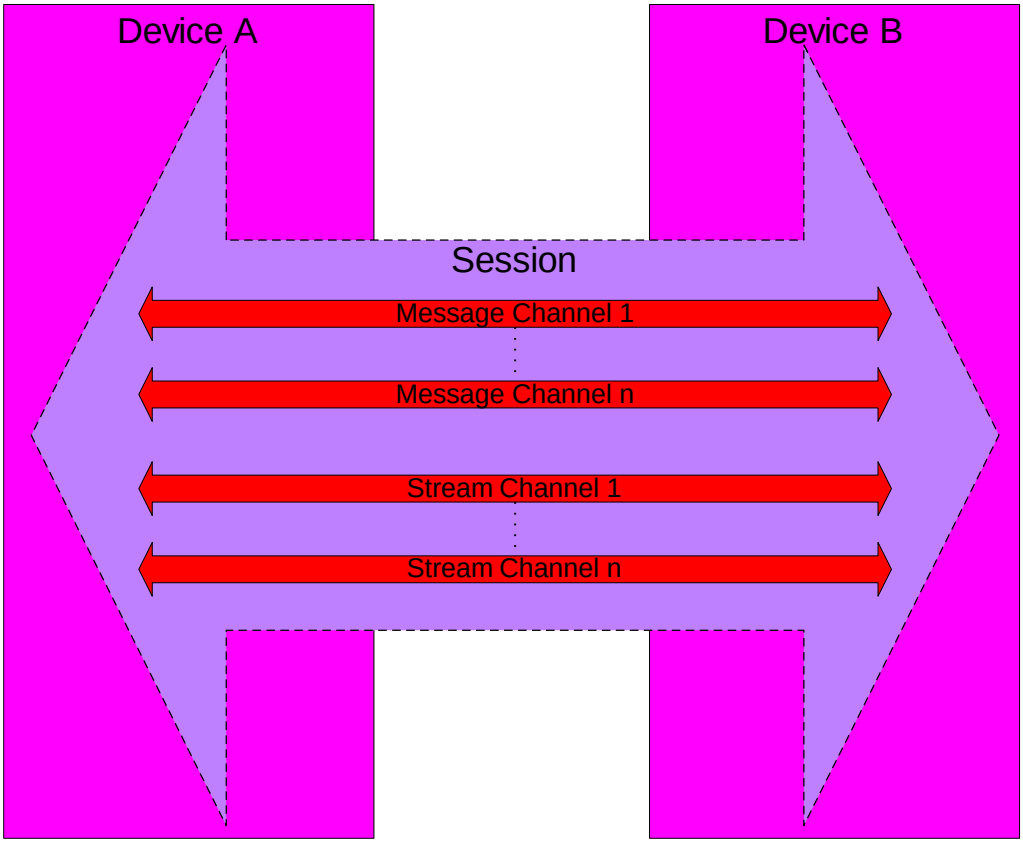


Figure 3

OSP is where OSP ports are actualized. (OSP ports have been known by various names including: HP-JetSend device numbers, HP-JetSend sub-addresses, and oipHandles). OSP is a session layer that allows the HP-JetSend Interaction Protocol (OIP) to establish multiple simultaneous sessions with other HP-JetSend devices. OIP can open multiple STREAM and MESSAGE channels to another HP-JetSend device through the session that it has with that device. Figure 3 illustrates a session between two HP-JetSend devices.

5.1.1.1 HP-JetSend Session Protocol

5.1.1.1.1 HP-JetSend Addresses

OIP can express an HP-JetSend address to OSP in either of two forms, HP-JetSend Multi-Network form or Universal Resource Locator form.

5.1.1.1.1.1 HP-JetSend Multi-Network form (OMN form)

OMN form is the standard form that OSP uses internally. If an address is given to OSP in URL form, OSP will store it internally in OMN form.

5.1.1.1.1.1.1 Syntax

Items enclosed in <> are non-terminal tokens.
Items enclosed in [] are optional fields.

```
<OMN>      ::= [<type>:]<host>[##<xptPort>][#<ospPort>]
<type>     ::= Network and/or protocol.
               Examples:
```

```
                    ip   -- IP address.  UDP transport.
                    ipx  -- IPX address.  IPX transport.
                    irda -- Infrared address.  TTP
                            transport.
                    fax  -- Telephone address and FAX
                            protocol
                    up   -- Unix path
                    lb   -- loopback
<host>    ::= Network specific station name or address
<xptPort> ::= Transport port number.  If the xptPort
              is left off it is assumed to be the
              default transport port number for the
              transport that is implied by the type
              field.
<ospPort> ::= OSP port number.  If the ospPort is left
              off it is assumed to be 1 since that is
              the default.
```

5.1.1.1.1.1.2 Examples

- ip:15.62.97.67
- ip:hpb15393.boi.hp.com#3
- ip:hpb15393.boi.hp.com##9101#3
- fax:9.1.208.396.4759

5.1.1.1.1.2 Universal Resource Locator form (URL form)

An address in URL form implies that the type is "ip". Since OSP works internally with the more general OMN form, it translates addresses in URL form to OMN form.

5.1.1.1.1.2.1 Syntax

HP-JetSend://<host>[:<xptPort>][#<ospPort>]

5.1.1.1.1.2.2 Examples

- HP-JetSend://15.62.97.67
- HP-JetSend://hpb15393.boi.hp.com#3
- HP-JetSend:// hpb15393.boi.hp.com:1782#3

5.1.1.1.1.3 Multi-Port Network-Parallel Gateway Addressing

JetDirect EX is an example of a multi-port network to parallel gateway. It has only one network address but [sometimes] more than one parallel port. Each device connected to one of the parallel ports shares the single network address with all the other devices attached to the other parallel ports. Each parallel port has a number, beginning with 0 for the first one. HP-JetSend associates a different “well-known” transport port with each parallel port number. For example, INS will register the following UDP ports with the Internet Assigned Numbers Authority (IANA):

UDP Port (xptPort)	Associated Parallel Port
1782 -- Default	0
9121	1
9122	2

An HP-JetSend appliance attached to parallel port 1 on a JetDirect EX with an IP address of 15.62.54.35 would be address as ip:15.62.54.35##9121. If that same HP-JetSend appliance contains more than one logical HP-JetSend device, the second logical HP-JetSend device would be addressed as ip:15.62.54.35##9121#2.

If the HP-JetSend appliance is attached to parallel port 0, it would be addressed as either ip:15.62.54.35 or ip:15.62.54.35##1782. The xptPort in this case may be left off since it is the default xptPort.

5.1.1.1.2 HP-JetSend Session

An HP-JetSend session is an agreement between two HP-JetSend devices that allows either side to open channels to the other side. For two HP-JetSend devices to communicate, a session must be established between them. Establishing a session involves opening a MESSAGE connection between the devices and using the OSP session establishment protocol over the connection to negotiate a session into existence. The OIP simply makes a function call into the OSP to establish a session with another device's OIP. OSP handles all the mechanics of setting up the session.

The architecture of OSP allows any HP-JetSend device to both actively and passively establish sessions. For session establishment negotiation to occur, one side of the negotiation must start out passively listening for a session and the other side must actively try to establish the session with the passive side. OSP will deny a session establishment request if a session already exists between the two devices unless one of the devices is an HP-JetSend gateway. If the OIP that is the target of the request is not listening for session establishment requests on the network through which the request arrived, OSP will deny the request. (This situation can exist when a single OSP is servicing more than one HP-JetSend device but only some of the devices' OIPs have asked OSP to listen on the network. See discussion of shared OSP below.)

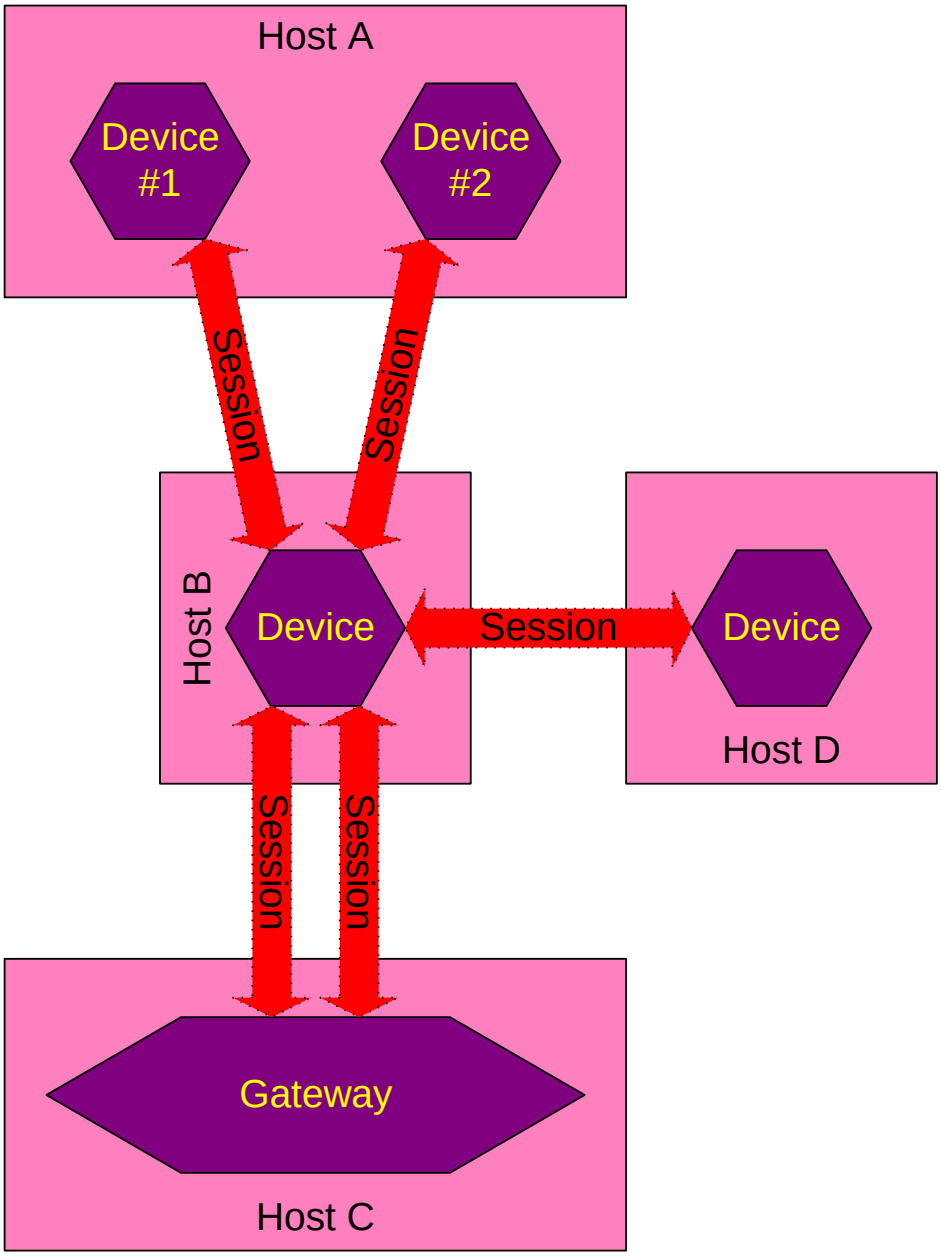


Figure 4

The architecture of OSP allows any HP-JetSend device to establish simultaneous sessions with multiple other HP-JetSend devices. Multiple sessions between a single pair of HP-JetSend devices are not allowed. Multiple sessions between an HP-JetSend device and an HP-JetSend gateway are allowed. Any HP-JetSend device may have - at the same time - both sessions that it passively created and sessions that it actively created. Figure 4 Illustrates the possibilities for multiple simultaneous sessions.

Either side of a session may initiate termination of the session. The requesting side is not required to wait (but should if it can) until it has no buffered data in the channel to the other side before requesting the termination. When a termination is requested, both sides stop sending data through all open channels and tear them down immediately.

When a session is terminated, OSP passes an event up to the OIP indicating that the session is terminated. At that point, the session identifier or handle along with any channel identifiers or handles that the OIP is tracking are no longer valid.

5.1.1.1.3 HP-JetSend Session Channels

An HP-JetSend session by itself does not transmit or receive OIP data. The OIP must request that OSP open channels on its behalf for communicating with its OIP peer through the session. A pair of OIPs can have an open session between them without having any channels open through the session. The OIP at either end of the session may initiate the opening and closing of channels. Both OIPs can open and close channels through the session at will. There are no architectural limits to the number or mix of STREAM and MESSAGE channels within a session. If a particular HP-JetSend implementation is not able to accommodate a request to open a new channel, it will reply to the request indicating that it cannot open any more channels of that type.

OIP can open two types of channels through an OSP session, STREAM and MESSAGE channels. Both types of channel offer a reliable- ordered-delivery service. STREAM channels require more overhead to maintain than MESSAGE channels because they are implemented using connection-oriented transport protocols like TCP. MESSAGE channels are implemented using RMTP. STREAM channels offer higher performance than MESSAGE channels for the delivery of large amounts of data, however, the OIP can choose to use either channel type for anything it wants to do. Some implementations may only support a very limited number (or even no) STREAM channels.

5.1.1.1.4 Shared OSP

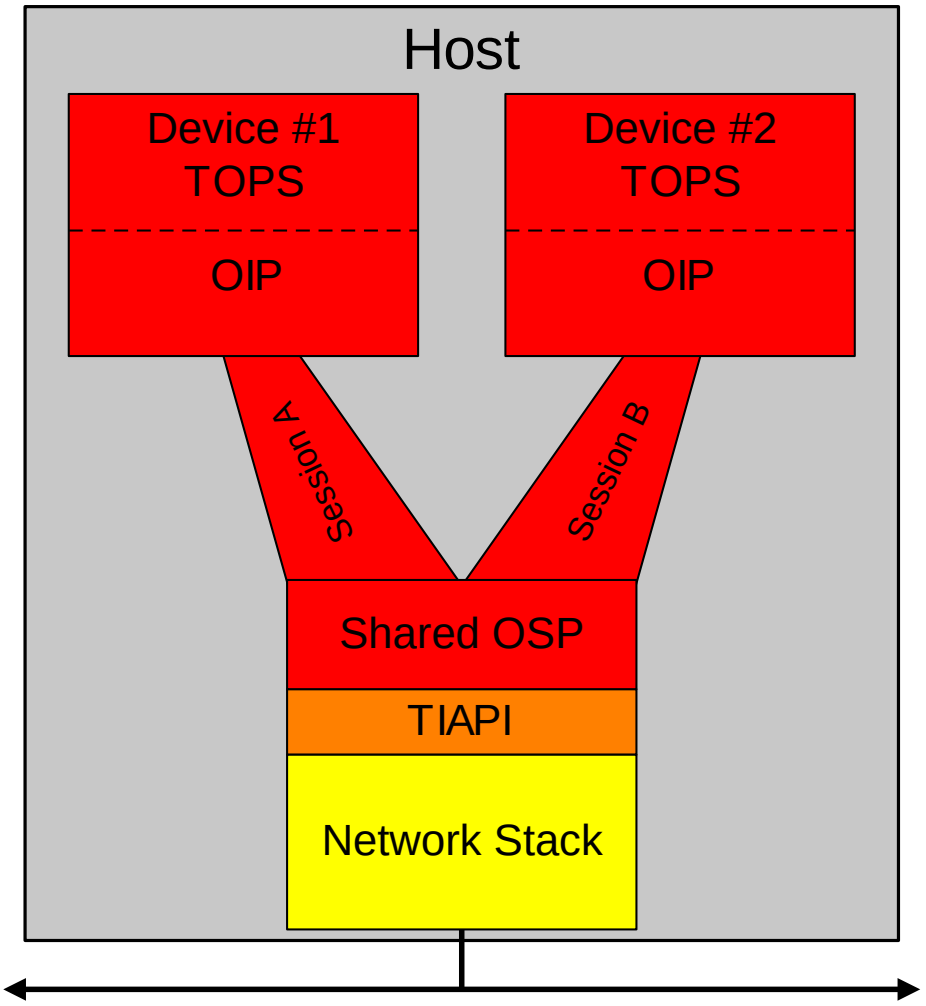


Figure 5

Within any one platform, only one instance of OSP runs. Figure 5 illustrates the fact that if more than one HP-JetSend device (soft appliance or desktop) is running simultaneously within the platform, they share the single instance of OSP.

An OSP port is a well-known session port that identifies an HP-JetSend device running within a platform. Each OIP running within a single host uses its own unique OSP port number and that port number is same as the device number with which the OIP is associated. The number is assigned to OIP administratively. OIP identifies itself to OSP using the number and OSP creates sessions for the OIP using the number as the local session endpoint identifier.

5.1.2 HP-JetSend Transport Independent API (TIAPI)

A complete description of the TIAPI is contained in [3].

The TIAPI provides an identical API to all transports on the host platform. It is the same syntactically and semantically on all HP-JetSend implementations. The internal implementation of TIAPI is device dependent. Depending on the native transport interfaces provided by the host platform, TIAPI may be implemented using TLI, Berkeley Sockets, WinSock, or any other interface or combination of interfaces.

OSP makes requests to and retrieves replies and events from TIAPI by using function calls. OSP "pushes" requests to TIAPI by making request function calls and "pulls" replies and events from TIAPI by making a retrieval function call. Each type of request has its own function call associated with it. For example, when OSP wants to request that TIAPI open a STREAM connection it calls the TIAPI function xptCall().

None of the TIAPI function calls are blocking calls. Request calls that TIAPI cannot process immediately or without blocking have a corresponding reply associated with them. For example, the request call xptCall() returns immediately to OSP with a return status that indicates whether or not TIAPI was able to accept the request. TIAPI then (asynchronous to OSP) processes the request by trying to set up the connection that OSP requested. When TIAPI completes the process it responds to OSP with a reply that corresponds to the original xptCall() request. It does not pass the reply back to OSP though. Instead it queues the reply internally and waits for OSP to retrieve it.

OSP retrieves replies and events from TIAPI by making a function call to TIAPI called xptGetNextEvent(). This function pulls the next reply or event off of TIAPI's internal queue in order and returns it to OSP. Replies and events are not ordered according to the order in which OSP made requests to TIAPI. TIAPI may be able to process several requests simultaneously. If so, requests may complete in a different order than their corresponding requests were made. Replies and events are queued for delivery to OSP in the order in which they occur.

To help OSP map replies that it retrieves to requests that it made earlier, all asynchronous TIAPI request calls include a request handle parameter. The value of the request handle is defined internally by OSP and is passed to TIAPI with the request call. When TIAPI generates the reply to a call it includes in the reply the request handle for the request that caused the reply. TIAPI does not know or care what the value of the request handle is. It also does not modify the request handle value. It simply keeps track of the request handle internally until it can return the handle to OSP in the appropriate reply.

5.1.3 Host Name Resolver (HNR)

HNR provides the functionality that allows users to enter addresses using "easy" host names instead of "difficult" network addresses. For example, the user can enter a host name like "MyComputer" instead of its equivalent IP address "15.63.115.201". TIAPI must use network addresses in all of its interactions with the transport service providers so TIAPI requires that OSP use network addresses in all of its requests for transport services. TIAPI provides OSP with a function that translates an OSP address into a TIAPI address, which contains the needed network address. Since OMN form addresses can contain host names and TIAPI addresses can not, TIAPI needs access to a functional unit that translates host names into network addresses. HNR is that functional unit.

HNR may contain functionality for resolving various types of host names into network addresses. Depending on the networks that a particular implementation of HP-JetSend supports, the implementation HNR may include one ore more of: Domain Name Service (DNS) resolver, Windows Internet Naming Sevice (WINS) resolver, NetWare Bindery Name resolver, etc.. The two most important to HP-JetSend are DNS and WINS.

DNS, one of the fundamental building blocks of the Internet, is a distributed mechanism for organizing the Internet's name space, and is responsible for providing translation between IP addresses and host names, as well as storing other information. DNS is a

client/server system. DNS servers fulfill requests from DNS clients to translate host names into IP addresses. DNS Resolvers provide the client side of the transaction.

WINS is similar to DNS in that it translates names into IP addresses. However, it translates NetBios names instead of the hierarchical DNS names. It is also different because it automatically updates its internal host name to IP address mappings when DHCP is used to dynamically assign IP addresses. DNS by contrast, must have its mappings (which are stored in a server based host table file) manually updated by an administrator. When DHCP is used in a DNS environment, the host tables are not updated.

There is a proposal before the IETF to extend the DNS protocol to make it dynamically update host tables when DHCP is used for address assignment. This new protocol is called DynDNS and as it is deployed on the internet, HP-JetSend implementations of HNR will need to include the name resolver client side of the protocol.

5.1.4 Reliable Message Transport Protocol (RMTP)

A complete description of the RMTP interface and protocol is contained in [2].

Older HP-JetSend documentation refers to RMTP as “reliable datagrams”. We changed the name because “reliable datagrams” is an oxymoron. RMTP is a reliable, sequenced delivery protocol. It is implemented by adding a minimal reliability mechanism and sequence numbering to a datagram transport protocol such as UDP or IPX. RMTP does not promise the same level of performance or reliability that STREAM protocols like TCP and SPX do. Depending on the networks that a particular implementation of HP-JetSend supports, RMTP implementations may run over one ore more of the following unreliable transports: UDP, IPX, IrDA-TTP, etc..

5.1.4.1 Service Objectives

- RMTP provides a full-duplex communications channel between the two ports of each transport connection.
- RMTP attempts to reliably deliver all user messages and will report a failure to the user if it cannot deliver a message. RMTP extends the datagram service of UDP (and/or other datagram protocols) to include reliable delivery.
- RMTP attempts to detect and discard all damaged and duplicate messages
- RMTP provides sequenced delivery of user messages.

5.1.4.2 Relation to other protocols

RMTP is a transport protocol. It extends the end-to-end services that a datagram transport protocol like UDP or IPX provides to include reliability and ordered delivery.

RMTP accepts data from OSP (via TIAPI) in units of messages. RMTP packages for transport each submitted OSP message as a single RMTP segment and passes the segment through the datagram transport for delivery. To package an OSP message into a RMTP segment, RMTP prepends its own header to the OSP message. RMTP will not attempt to fragment a large OSP message into smaller segments and reassemble the message on the receiving end. (OSP must ensure that any message it passes to RMTP is small enough for RMTP to prepend its header to and send as a single RMTP segment). This differs from a byte-stream protocol such as TCP which supports the transfer of an indeterminate length stream of data between ports, buffering data until it is requested by the receiver.

5.1.5 Supported Transport and Network Protocols

HP-JetSend will support these transports if they are available on the host platform: TCP and UDP. Other protocols such as SPX, IPX, and IrDA may be supported in future implementations.

5.2 Fall 97 Implementation Architecture

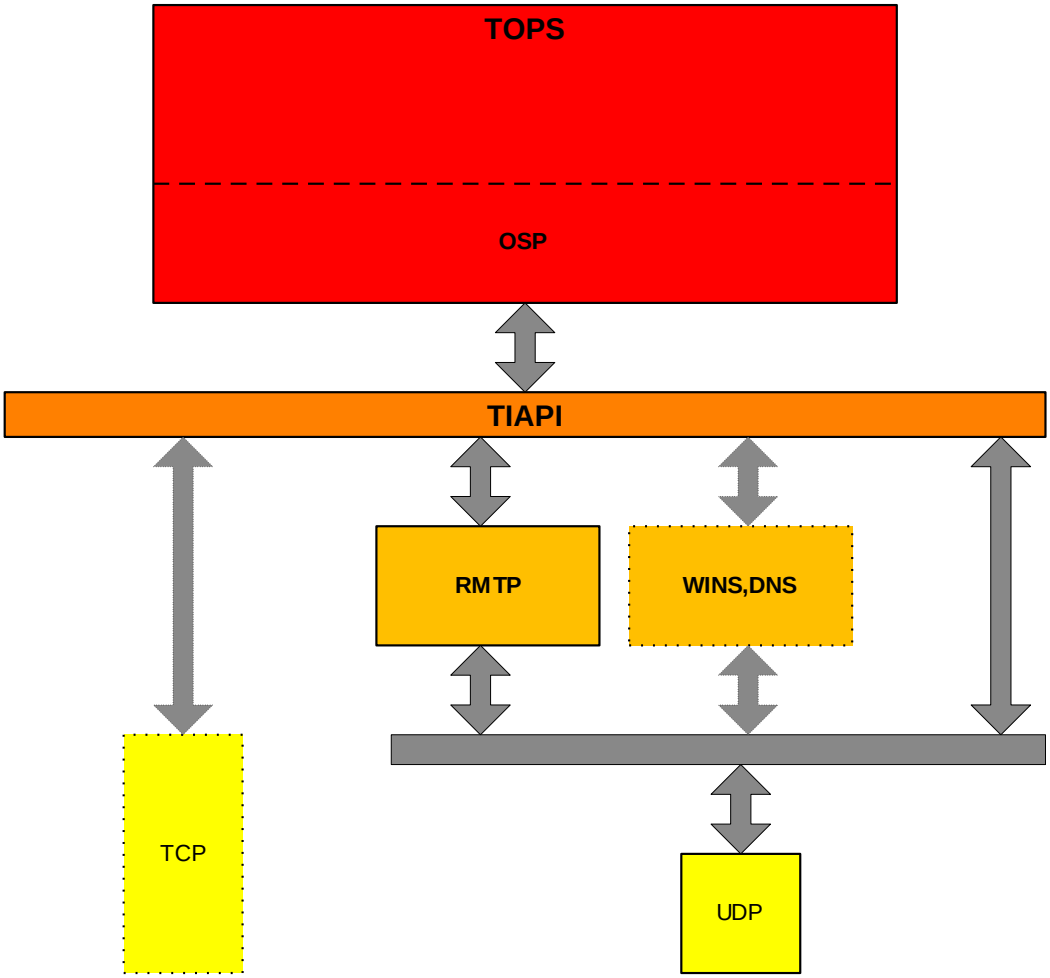


Figure 6

Figure 6 shows the generalized implementation architecture for the Fall 97 implementation. The following sections describe the specific implementation architectures for the Fall 97 platforms.

5.2.1 HP Scanner Platforms (Verona)

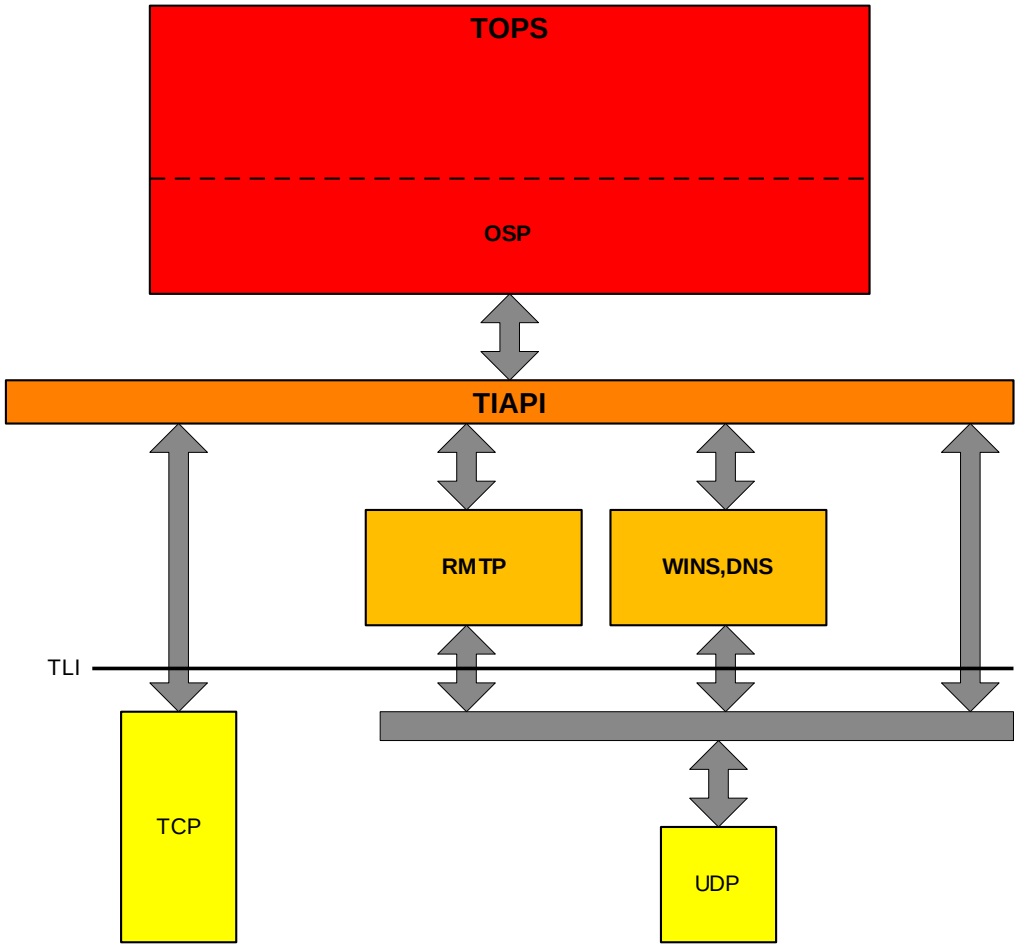


Figure 7

Figure 7 is logically equivalent to the Generalized Architecture diagram of section 5.1. It represents the logical relationships of the components as they will exist in an HP scanner implementation. The transports' services are projected through the peripheral IO subsystem from JetDirect and exposed to HP-JetSend through a TLI interface.

5.2.2 HP Printer Platforms (MoBY, Monarch)

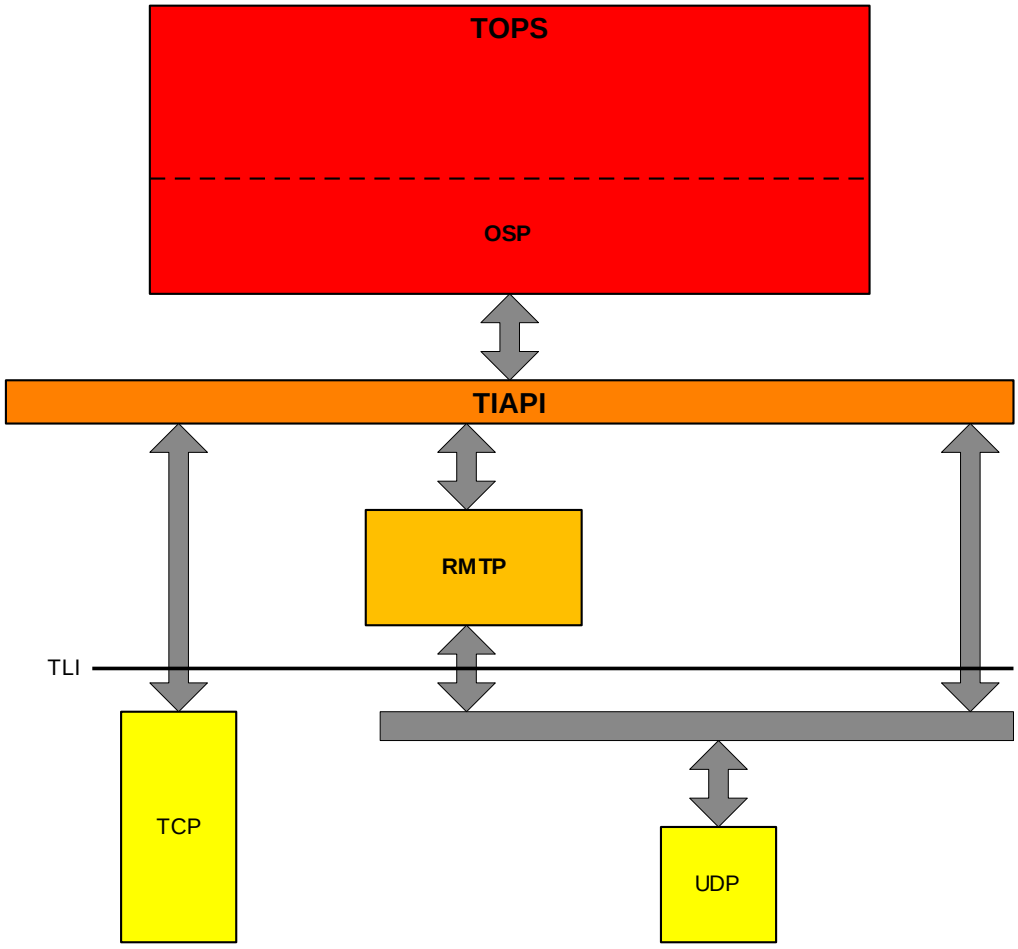


Figure 8

Figure 9 is logically equivalent to the Generalized Architecture diagram of section 5.1. It represents the logical relationships of the components as they will exist in an HP printer implementation. The transports' services are projected through the peripheral IO subsystem from JetDirect and exposed to HP-JetSend through a TLI interface.

5.2.3 Desktop Platforms (Windows 95 and Windows NT)

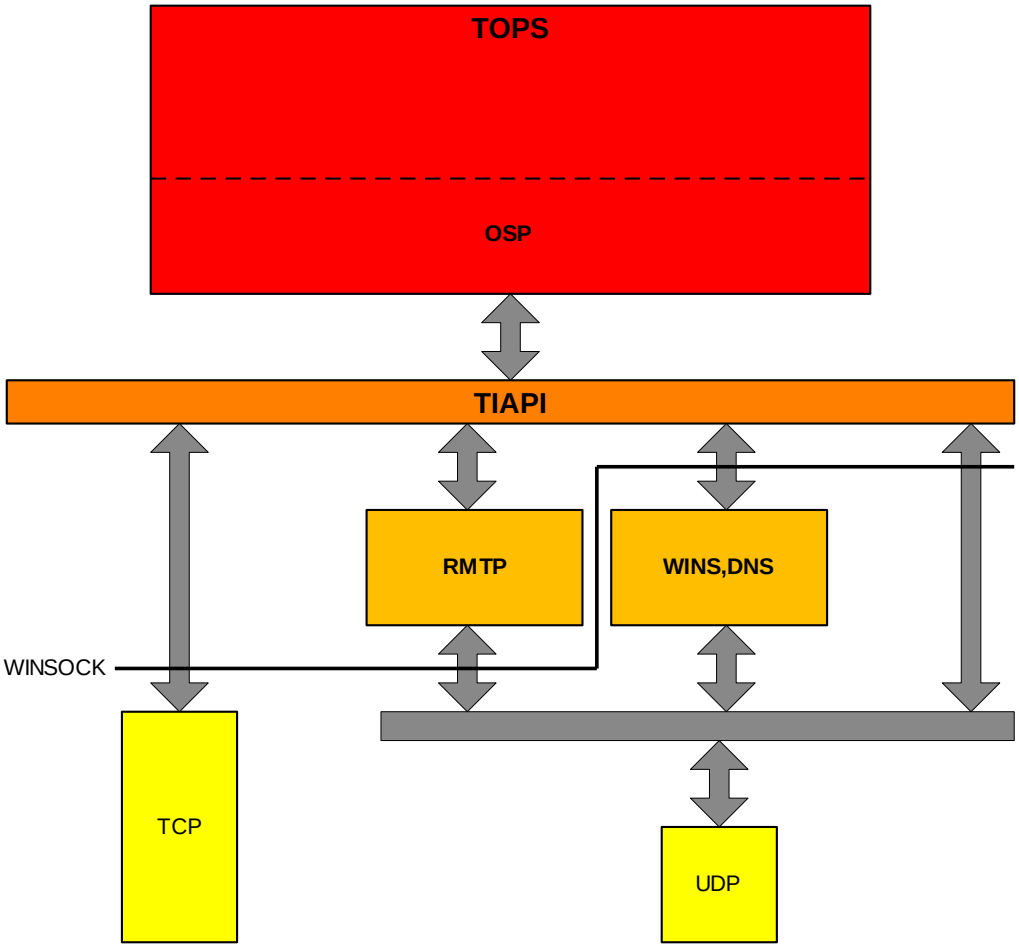


Figure 9

Figure 9 is logically equivalent to the Generalized Architecture diagram of section 5.1. It represents the logical relationships of the components as they will exist in a Windows desktop implementation. The transports' services are provided by the Windows environment and are exposed to HP-JetSend through a WinSock interface.

6. Appendix A (Addressing Considerations)

6.1 TCP/IP

6.1.1 IP Address

An IP address is of the form xxx.xxx.xxx.xxx where xxx is a decimal number between 0 & 255 inclusive.

6.1.2 Host Name

A host name must be resolved to an IP address through a DNS (Domain Name Service) or WINS (Windows Name Resolution).

6.1.3 Recommendation

Allow for both IP addresses and host names as viable addressing mechaniOIPs in the architecture. HP-JetSend devices may choose to limit themselves by only allowing users to enter IP addresses due to control panel constraints, for example, HP-JetSend devices that only have a ten-key pad. In a DHCP environment, a host name will be the only means available to identify target HP-JetSend devices as DHCP will allow for dynamic assignment of IP addresses.

6.2 SPX/IPX

6.2.1 IPX Address

An IPX address is of the form nnnnnnnn:xxxxxxxxxxxx or xxxxxxxxxxxx (no network specified) where n & x is a hex number between 0x00 & 0x0f inclusive. The address is broken down into two fields: 1) where n is the network address and 2) where x is the LAN card hardware address. If the user doesn't enter the network address it's assumed to be a local network address. This network address is dependent on the frame type (Ethernet 802.2, Ethernet 802.3, Ethernet II, or Ethernet SNAP). This method is far from being usable for end users.

6.2.2 User Names

NetWare user names are placed in the bindery for NetWare 3.X or in NDS (NetWare Directory Services) for NetWare 4.X. NetWare only names servers in it's networked environment. NetWare client machines are not named.

6.2.3 Real Names

NetWare real names are placed in the bindery for NetWare 3.X or in NDS (NetWare Directory Services) for NetWare 4.X. NetWare only names servers in it's networked environment. NetWare client machines are not named.

6.2.4 Recommendation

Allow for IPX addresses as a viable addressing Mechanism in the architecture. HP-JetSend devices may not allow users to enter IPX addresses due to control panel constraints, for example, HP-JetSend devices that only have a ten-key pad and since the IPX address is a 20 digit hex address. One way around this problem is to depend

on pre-configuration of HP-JetSend devices that wish to use IPX as a transport. This wouldn't be as bad as it seems since users in this type of environment are already accustom to having to deal with NetWare administrators.

6.3 IrDA (Infrared)

When an IrDA device comes into range of another IrDA device, the two devices will appear on each others' connected list. Users will not need to enter an address.

6.4 FAX

6.4.1 FAX Gateway

A FAX gateway should be pre-configured for an HP-JetSend device. All the user would need to do is enter a telephone number and the HP-JetSend device will pass on the telephone number to the FAX gateway. FAX gateways are nothing more than network nodes with either an IP address or IPX address.

6.4.2 FAX Appliance

A FAX appliance is nothing more than a regular HP-JetSend device. An HP-JetSend device connects to the FAX appliance and the FAX appliance would send a remote control panel to allow the user to enter a telephone number or telephone numbers for a distribution list. FAX appliances are nothing more than network nodes with either an IP address or IPX address.

6.4.3 E-mail Gateway

E-mail could be a possible way for two HP-JetSend devices to communicate. E-mail could be thought of as nothing more than a very slow transport which would be unusable for E-material negotiation; however, since all HP-JetSend devices are required to support a lowest common denominator data type or encoding, that data type could be sent in all cases. Most companies already have an e-mail infrastructure and it is one way (in addition to FAX) of getting around fire-walls.

6.5 Sub-Addressing

Once the network node is identified, further addressing will be needed to identify the service requested. For examples, PC's will have multiple services such as a Desktop target and maybe a printer soft appliance. Sub-addressing was handled in the HP-JetSend field trial by using trailing #<number> identifiers. There is really no reason for needing to change this.

7. Appendix B (Dynamic Address Assignment)

Automatic addressing provided through network services (infrastructure) create the problem of not being able to use the network address as a 'handle' for a target network node. Without automatic addressing there is a one-to-one relationship between the network address and the target network node; while with automatic addressing, that relationship is lost. To solve the addressing problem we need to find a 'handle' that can be mapped directly to the target network node.

7.1 TCP/IP

7.1.1 Bootp

This method is typically used for assigning IP addresses on a Unix TCP/IP network. It requires a server on the network to be running a Bootp service.

7.1.2 DHCP (Dynamic Host Configuration Protocol)

This method is typically used for assigning IP addresses on a Microsoft TCP/IP network. It requires a server on the network to be running a DHCP service. It has three modes of operation: 1) infinite lease for an IP address, 2) timed lease for an IP address, or 3) assigned IP address based on the network card's MAC (hardware) address.

7.1.3 IPCP (Internet Protocol Control Protocol)

This method is typically used for assigning IP addresses over a PPP (Point-to-Point Protocol) link running on a Microsoft TCP/IP network. It requires that the connecting server handles the IP address assignment. This method is also typically used for assigning IP addresses over a SLIP (Serial Line Interface Protocol) link running on a TCP/IP network. PPP is typically used for running several protocols simultaneously over a phone line. SLIP is typically used for running TCP/IP over a phone line.

7.2 SPX/IPX

7.2.1 NetWare

The network part of the SPX address is determined by a NetWare server. For JetDirect, the network number is sensed by looking at packets on the network.

7.3 IrDA (Infrared)

All devices on the IrDA network will identify themselves and be assigned an address by the IrDA protocol.

8. Appendix C (Name Services)

Name services are network services (infrastructure) that allow the mapping of a 'handle' to a network address. Name services are network and network operating system dependent (specific).

8.1 TCP/IP

8.1.1 DNS (Domain Name Service)

This name service is implemented as a static table structure in a file (/etc/hosts for Unix, or either hosts or lmhosts in the Windows directory for Windows). This name service lookup can either occur locally or over the network. If it occurs over the network, the machine requesting this service must have been previously configured as to where to look for this service. Configuration is nothing more than either an IP address or a hostname.

8.1.2 WINS (Windows DNS)

This name service is implemented as being more dynamic in nature than the DNS structures. It is Windows specific.

8.2 SPX/IPX

8.2.1 Bindery

This is NetWare 2.x/3.x specific. NetWare 4.x servers run a bindery service called "bindery emulation". A bindery, in the Novell sense, is a name resolution service (database) that is specific to a NetWare server. NetWare servers can exchange elements within their bindery with other NetWare servers.

8.2.2 NDS (NetWare Directory Services)

This is NetWare 4.x specific. NDS is a network-wide name resolution service (database) that is specific to a group (referred to as a NDS tree) of NetWare 4.x servers. NetWare 4.x servers can exchange elements within their part of the NDS tree with other NetWare 4.x servers. This is called partitioning. Also NDS replication and duplication (for backup purposes) is done.

8.3 IrDA (Infrared)

All devices on the IrDA network will identify themselves and be assigned an address by the IrDA protocol.

9. Appendix D (Device Browsing / Discovery)

9.1 Browsing and Discovery Methods

9.1.1 Using Existing Name Services

9.1.1.1 Pros

- Fits into existing network services (infrastructure).
- Not limited to the local subnet.

9.1.1.2 Cons

- Would require network services (infrastructure) to be available (it doesn't follow the HP-JetSend paradigm).

9.1.2 Using Broadcast Messages

A datagram packet would be broadcast to all HP-JetSend devices on the network. The HP-JetSend devices would listen on a specific well-known port number which would be the target of the broadcast. All HP-JetSend devices would reply based on the data in the broadcast packet. Wildcard matching would be possible. This wildcard matching could be made into a very powerful feature. The data in the datagram packet could be used to limit the number of responses to the query. Many HP-JetSend devices would not have the resources required to handle all the possible network responses. For example, these are the types of queries that can be made:

- query for only receive HP-JetSend devices.
- query for only send HP-JetSend devices.
- query for only HP-JetSend FAX gateways.
- query for only HP-JetSend devices that can exchange address books.
- query for only HP-JetSend devices that contain a certain string in their self surface description.
- etc.

9.1.2.1 Scope

How far would the broadcast packet be carried? Probably only to the local subnet so as to minimize the overall effect to the network with respect to bandwidth and performance.

9.1.2.2 Pros

- Doesn't require network services (infrastructure). Follows a true peer-to-peer model.

9.1.2.3 Cons

- Possible increase in network traffic.
- Limited to the local subnet.

9.2 Browsing and Discovery Uses

9.2.1 FAX Gateway Discovery

FAX gateways may be discovered by the various methods described above.

9.2.2 Address Book Sharing

Address book sharing is defined as the ability for two HP-JetSend devices to share their address books. The HP-JetSend devices may be discovered by the various methods described above.