

E17

IMS Connect

Suzie Wendler



St. Louis, MO

Sept. 30 - Oct. 3, 2002

Abstract

As a general purpose TCP/IP socket server for IMS, IMS Connect lays the architecture framework for the IMS e-business world and for future support of new technologies. It is considered a strategic base for IMS connectivity solutions and is the foundation for the IBM IMS solutions for the internet environment. This session details the architecture of IMS Connect, it's usability, and discusses the recent updates to the product.

Agenda

▲ **The Basics**

▲ **Updates**

What is IMS Connect?

▲ **A product (5655-E51) that provides connectivity support between TCP/IP applications and IMS/TM**

▲ **Benefits and Value**

- Supports TCP/IP sockets access to IMS transactions and commands
 - ▶ No requirement to modify existing IMS transactions
- Provides a general purpose and structured interface
 - ▶ For the IMS Connectors
 - ▶ For user-written clients
- Provides a strategic base for new connection technologies

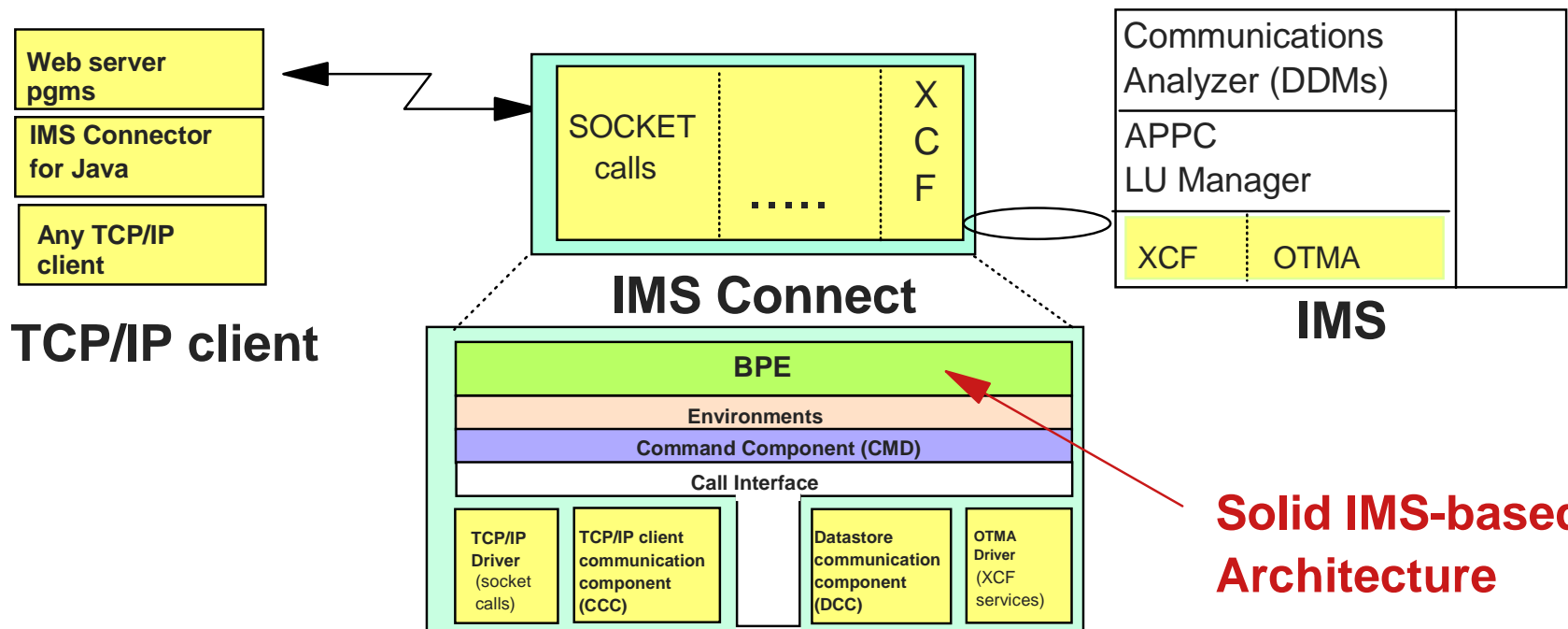
IMS Connect Architecture

▲ Executes in a separate MVS Address Space than IMS

- Functions as a TCP/IP server for communication with external clients
- Uses MVS XCF Services to access IMS OTMA

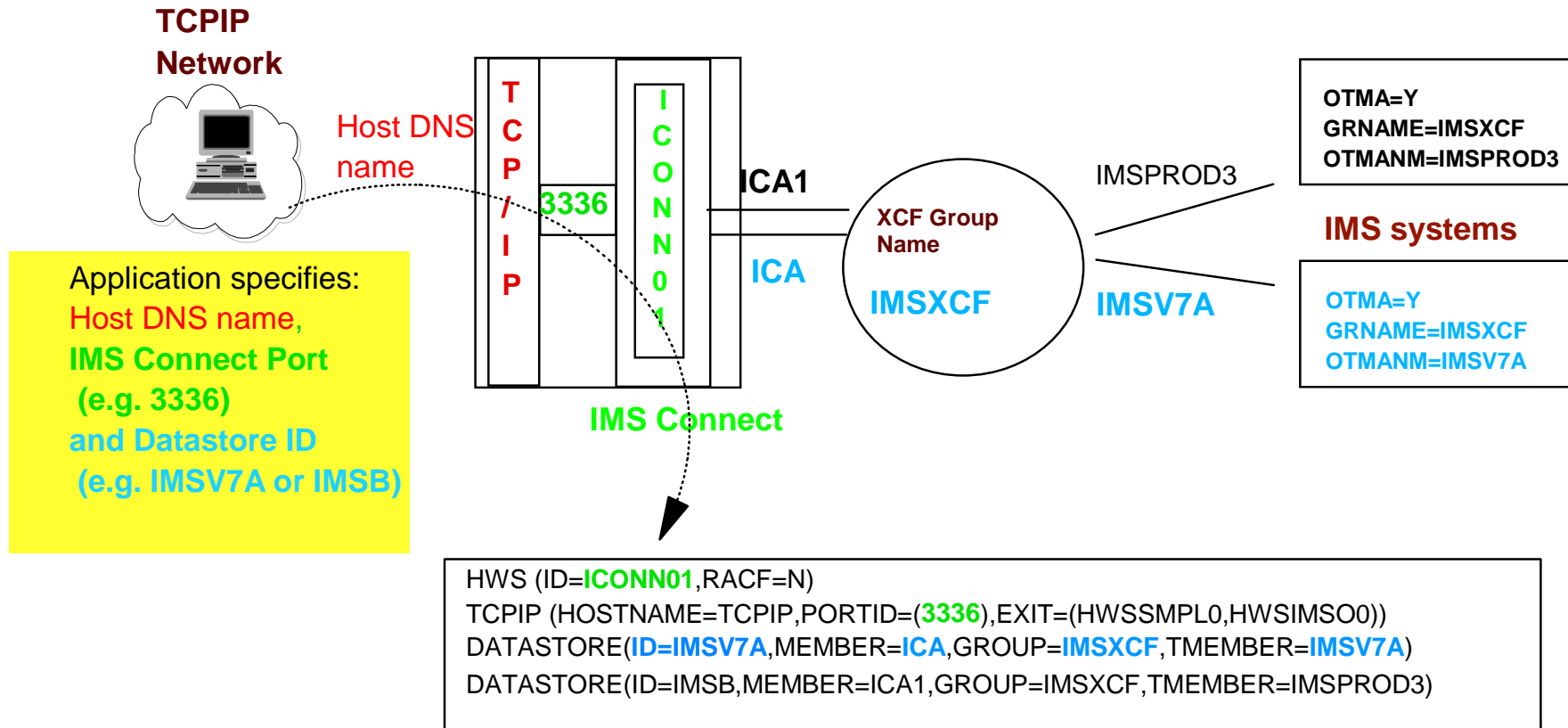
▲ Configuration supports

- Multiple IMS Connects accessing the same IMS system
- A Single IMS Connect accessing multiple IMS systems



Solid IMS-based Architecture

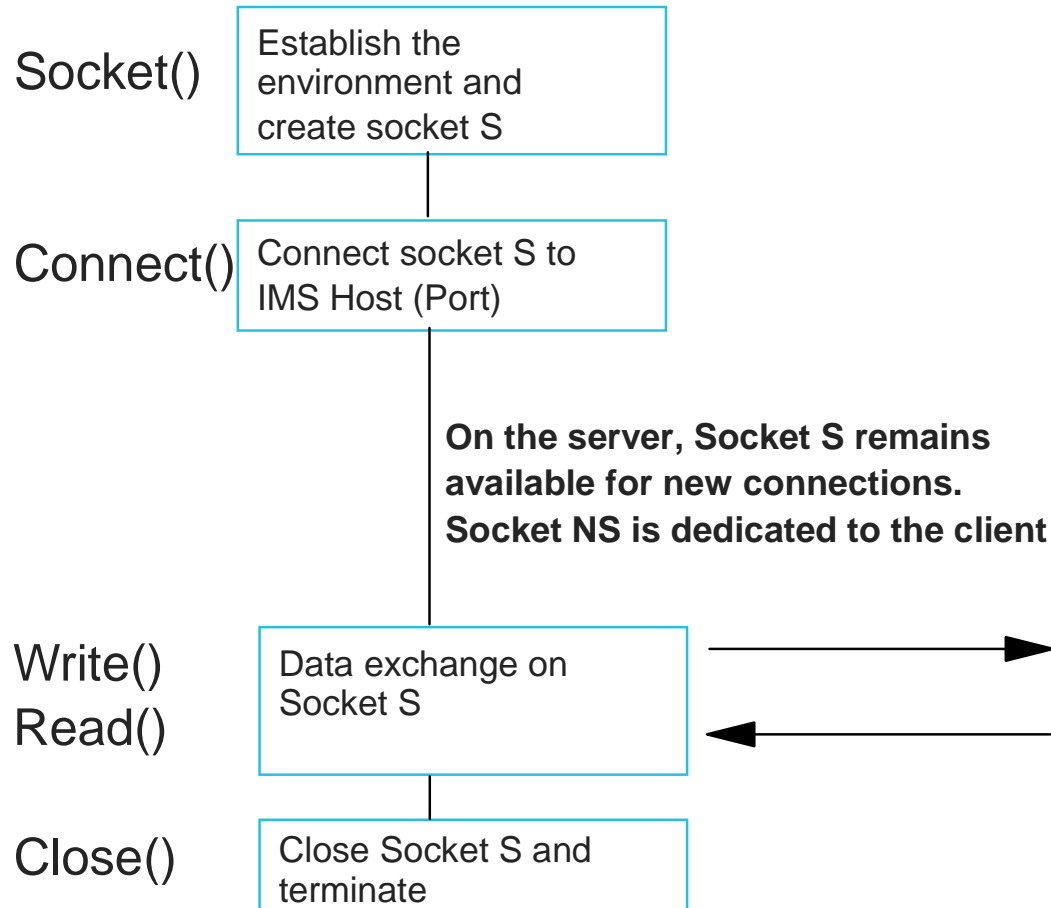
Configuration



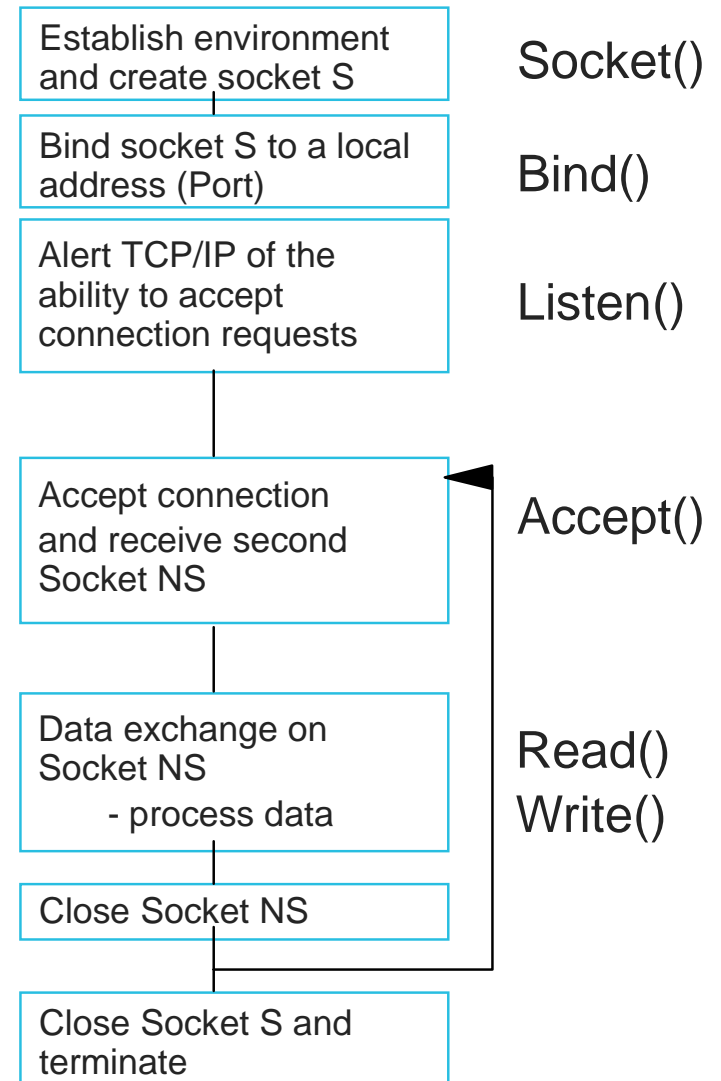
IMS Connect configuration (HWSCFGnn) member resides in IMS.PROCLIB

Socket Application Basic Design

TCP/IP Socket CLIENT

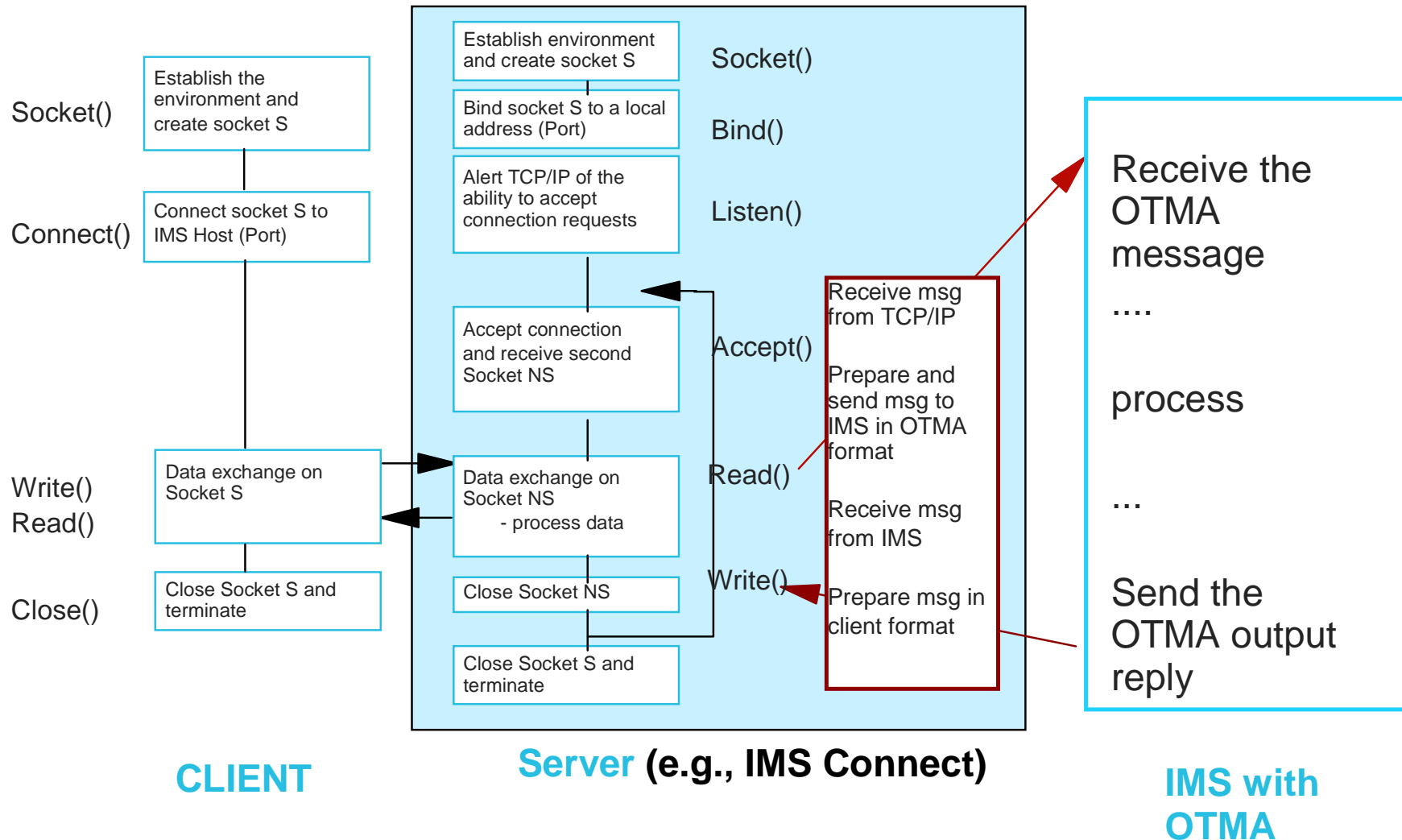


Server (e.g., IMS Connect)



Socket Application Basic Design ...

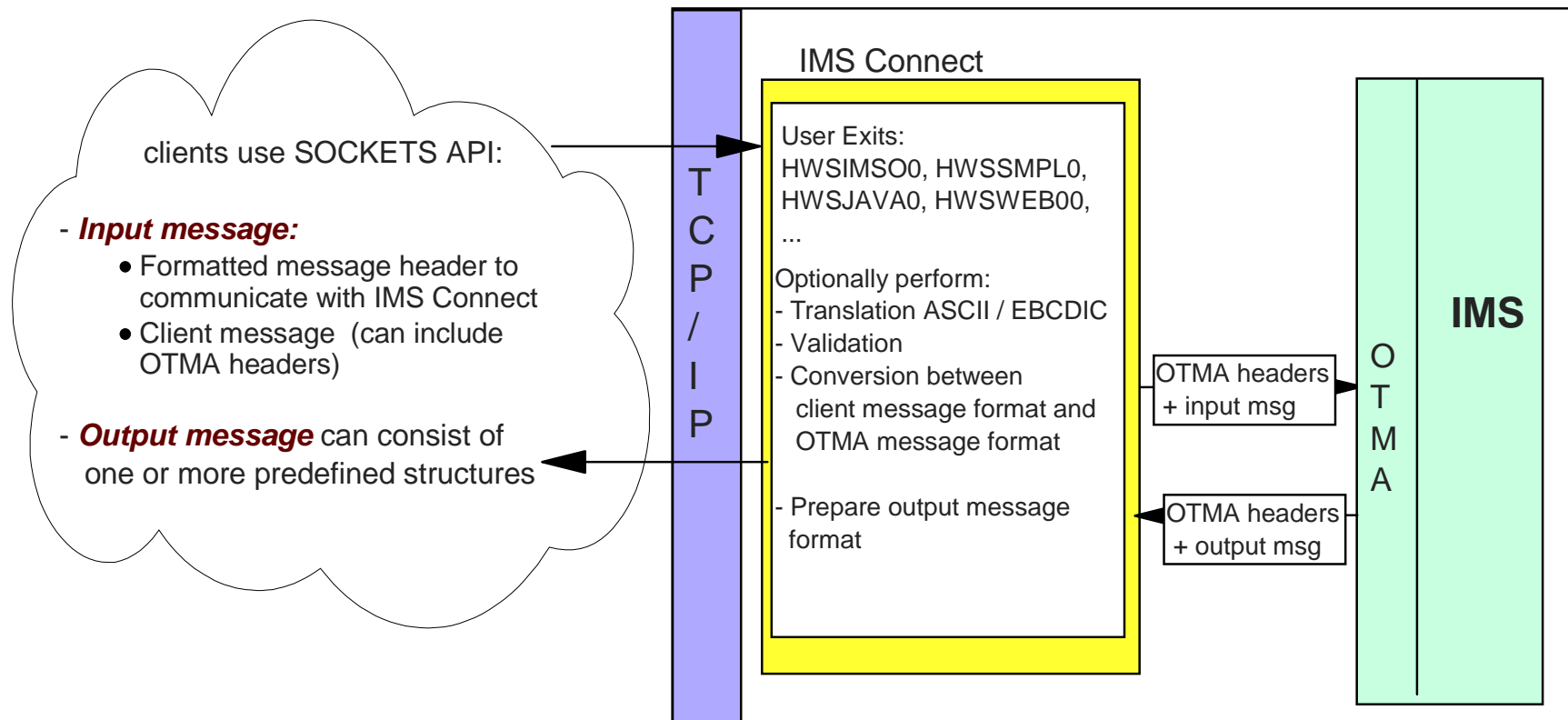
● Add IMS into the picture



Message Flow

▲ IMS Connect application protocol

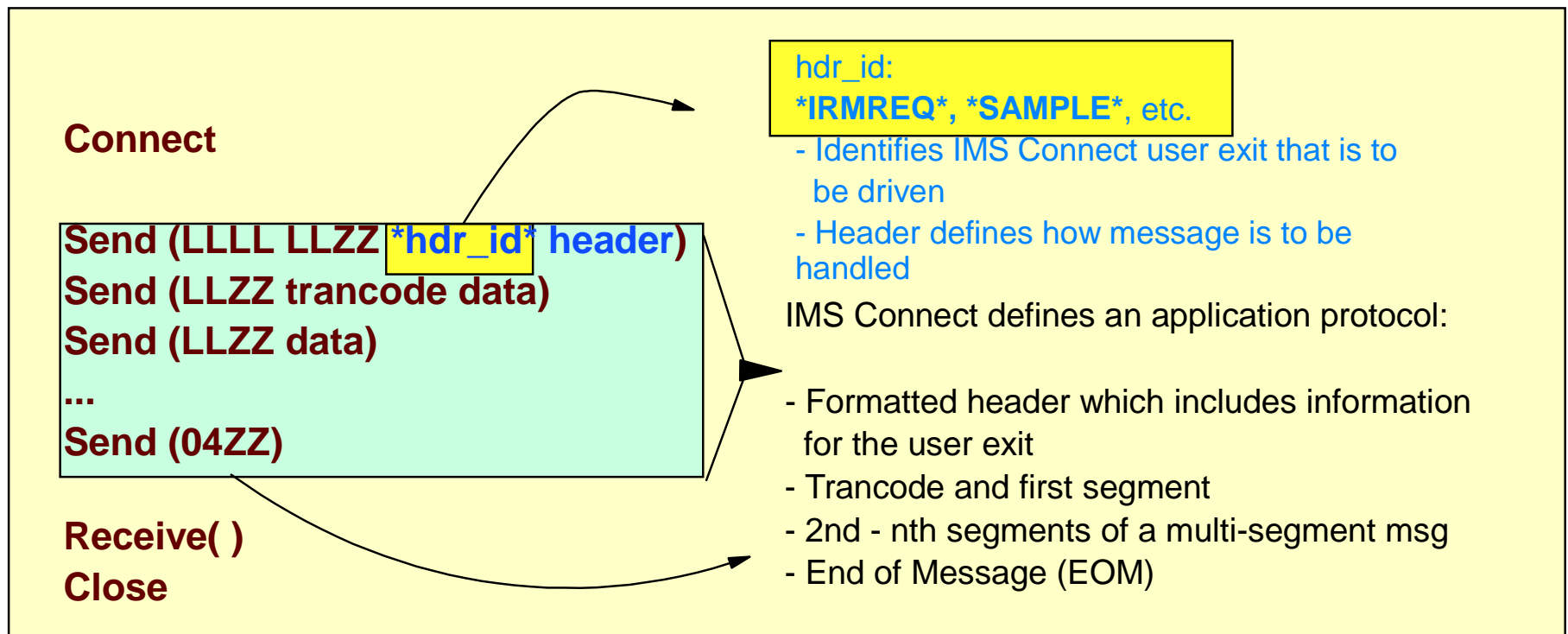
- defines layout of input/output messages



IMS Connect Application Protocol

▲ Input Messages

- LLLL = length of entire msg including all data segments and the EOM
- LL = length of the header data



Note: **hdr_id + header** are often generically referred to as the **IRM**

IMS Connect Application Protocol ...

▲ Output Messages

Connect
Send ()
...
Receive()
Close

Possible output replies:

```
LL ZZ data  
LL ZZ data ...  
LL ZZ *CSMOKY*
```

```
LL ZZ *REQMOD* mod name  
LL ZZ data ...  
LL ZZ *CSMOKY*
```

```
LL ZZ *REQSTS* return code reason code
```

**With PQ48182 and new message exits
HWSIMS01/HWSSMPL1:**

```
LLLL LL ZZ data ... LL ZZ *CSMOKY*
```

```
LLLL LL ZZ *REQMOD* ...
```

```
LLLL LL ZZ *REQSTS* ....
```

Output Messages ...

▲ CSMOKY (Complete Status Message)

- Sent by IMS Connect upon successful interaction with IMS

LLLL	LL	ZZ	*CSMOKY*
------	----	----	----------

▲ RMM (Request Mod Message)

- Returned as the first structure of an output message if the MFS mod name was requested

LLLL	LL	ZZ	*REQMOD*	MFS mod name
------	----	----	----------	--------------

▲ RSM (Request Status Message)

- Sent by IMS Connect upon rejection of an inbound request
 - ▶ Return and reason codes are documented in the IMS Connect Guide and Reference

LLLL	LL	ZZ	*REQSTS*	Return code	Reason code
------	----	----	----------	-------------	-------------

▲ Outbound application reply message

LLLL	LL	ZZ	data ...
------	----	----	----------

IMS Connect User Message Exits

▲ Basic capabilities of all IMS Connect sample exits

- Translates ASCII <--> EBCDIC
- Provides optional call to a user-written security exit
- Prepares messages for Client <--> OTMA
 - ▶ Creates OTMA headers inbound to IMS
 - ▶ Creates optional architected headers for output messages to the client
 - ▶ Value:
 - Relieves remote TCP/IP client programs from having to build/interpret OTMA headers
- Provides optional editing/processing
 - ▶ Before the input message is seen by IMS
 - ▶ Before output messages are sent back to the client
- Written in Assembler

IMS Connect User Message Exits ...

▲ **HWSIMSO0 - *IRMREQ* header**

- Install this exit if any TCP/IP clients use the default interface

▲ **HWSIMSO1 - *IRMRE1* header (PQ48182)**

- Similar to HWSIMSO0 but defines full length (IIII) preceding output

▲ **HWSJAVA0 - *HWSJAV* header**

- used by the IMS Connector for Java

▲ **HWSSMPL0 - *SAMPLE* header**

- Used by the sample IMS Client for Java

▲ **HWSSMPL1 - *SAMPL1* header (PQ48182)**

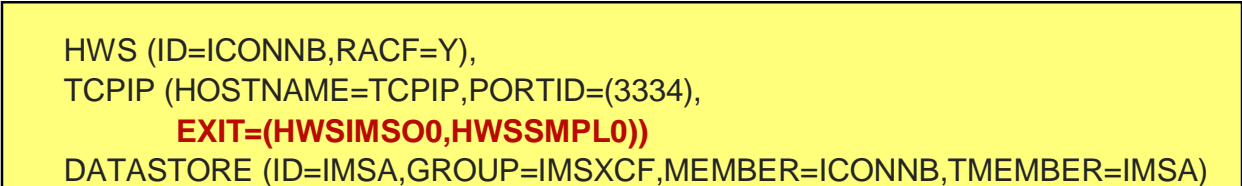
- Similar to HWSSMPL0 but defines full length (IIII) preceding output

Invoking Exits

▲ How are the exits invoked?

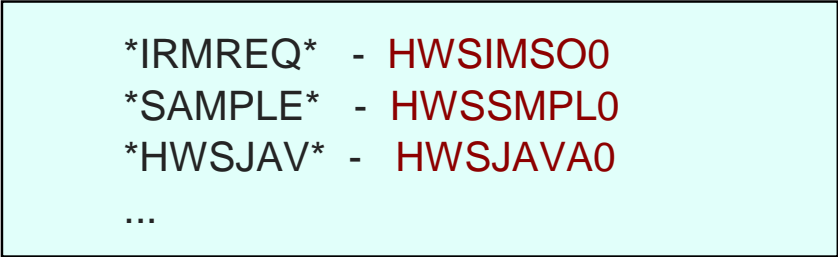
- IMS Connect loads each requested exit

IMS Connect configuration file



```
HWS (ID=ICONNB,RACF=Y),  
TCPIP (HOSTNAME=TCPIP,PORTID=(3334),  
      EXIT=(HWSIMSO0,HWSSMPL0))  
DATASTORE (ID=IMSA,GROUP=IMSXCF,MEMBER=ICONNB,TMEMBER=IMSA)
```

- IMS Connect calls each exit's INIT subroutine and determines the exit's header



```
*IRMREQ* - HWSIMSO0  
*SAMPLE* - HWSSMPL0  
*HWSJAV* - HWSJAVA0  
...
```

HWSUINIT

▲ User Initialization Exit Routine (HWSUINIT)

■ USAGE

- ▶ Driven during initialization and termination
- ▶ Load user table(s) and obtain any needed storage
- ▶ Add user data to INIT and DATASTORE tables

■ INIT TABLE

- ▶ Points to the datastore table
- ▶ Allows user data to be stored

■ DATASTORE TABLE (datastore = an IMS system)

- ▶ Contains **datastore id's, status** (active or inactive) and optional user data

Note: INIT and DATASTORE tables are passed to the user message exits

Datastore Table

▲ Potential uses

- Route the message to another IMS if the original is not active
 - ▶ Assumes another IMS can process the message
- Accept a generic datastore id from the client and choose a specific IMS
- Provide translation of a datastore id

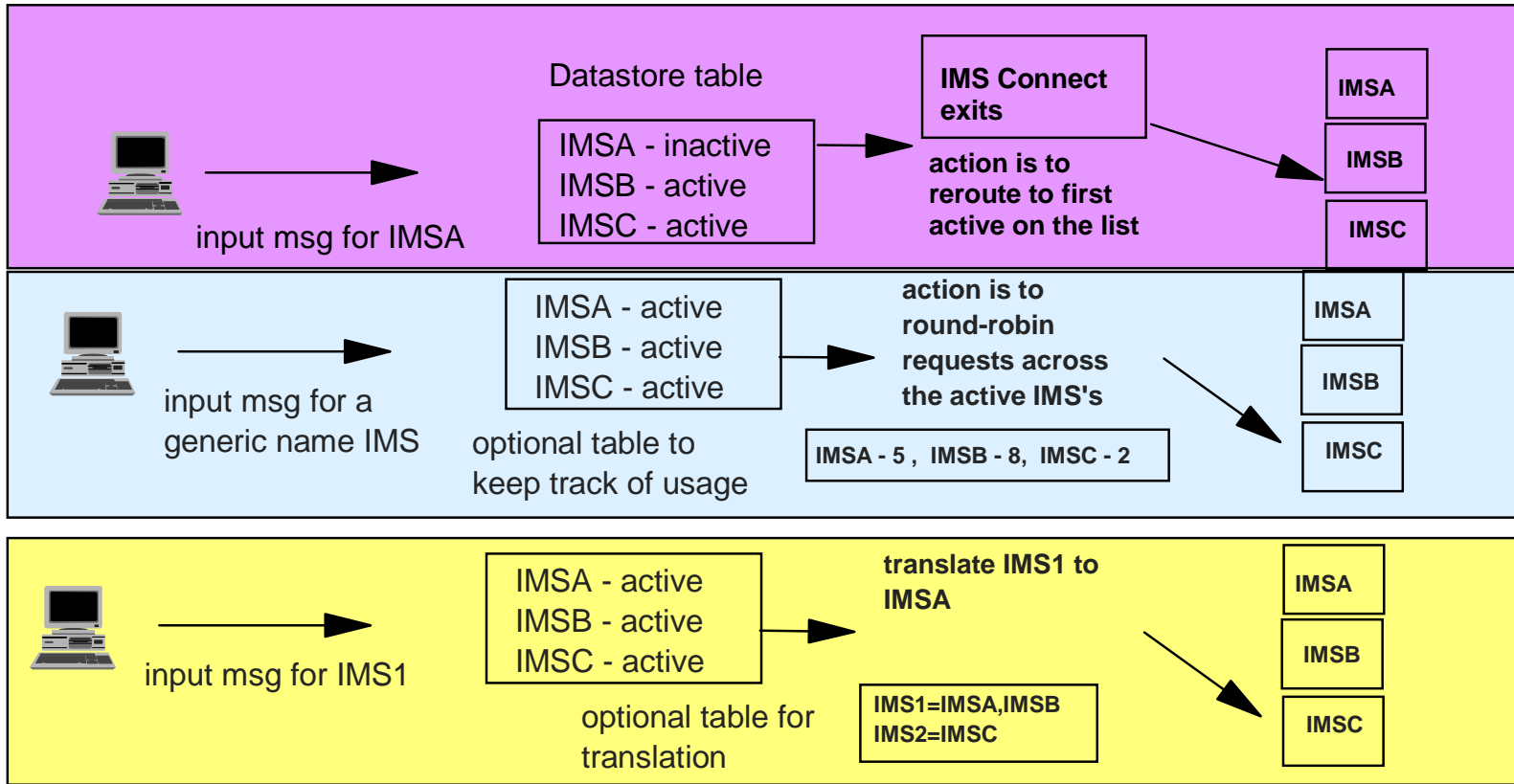
▲ NOTE: IMS Connect provides the interface

- IMS Connect **message** exits that are provided do not take advantage of the capability (they can be enhanced to do so)

Datastore Table

```

HWS (ID=ICONNA,RACF=Y)
TCPIP (HOSTNAME=TCPIP,RACFID=IDX,PORTID=(3333),EXIT=(HWSIMSO0,HWSSMPL0)
DATASTORE (ID=IMSA,GROUP=IMSXCF,MEMBER=ICONN1A,TMEMBER=IMSA)
DATASTORE (ID=IMSB,GROUP=IMSXCF,MEMBER=ICONN1B,TMEMBER=IMSB)
DATASTORE (ID=IMSC,GROUP=IMSXCF,MEMBER=ICONN1C,TMEMBER=IMSC)
    
```



Application Protocols

▲ Synchronization level (Sync_level)

- ▶ NONE
- ▶ CONFIRM
- ▶ SYNCPOINT (PQ57191)
 - Supports two-phase commit via local option (S/390, z/OS) for Websphere/390

▲ Commit modes

- *Commit_then_send (Commit mode 0)*
 - ▶ Output is sent as a result of syncpoint
 - ▶ Always uses sync_level of CONFIRM
 - ▶ Output is queued until client sends an ACK
- *Send_then_commit (Commit mode 1)*
 - ▶ IOPCB output is sent before syncpoint
 - ▶ Sync_level can be either NONE or CONFIRM

Sockets

▲ Sockets

- TCP/IP application programming interface (API)
- Connection between two TCP/IP programs

▲ Socket type is controlled by the client application

- Set socket flag in the IRM header:

- Non-Persistent socket - connection is terminated after each send to a client, even in a conversation
- Transaction socket - connection is terminated after each transaction general default
- Persistent socket - connection is maintained across transactions requested by client application

Note: For "Transaction sockets", a transaction is defined as:

- A single input/output message sequence for non-conversational transactions.
- Multiple input/output message sequences for a conversational transactions. The connection is

kept

until the conversation is terminated.

Persistent Sockets

▲ Capability that allows the socket connection to remain active for multiple transaction iterations

- Only supported for Send-then-commit (Commit mode 1)
- **Client actions**
 - ▶ To establish persistence:
 - Set the Persistent Socket flag in the message header
 - Must be done on every inbound msg (transaction or not)
 - ▶ To terminate persistence:
 - Issue a Disconnect, or
 - Send a message without the persistent flag
- **IMS Connect**
 - ▶ Maintains persistence and assumes the client will request the disconnect
 - ▶ Allows the User Exits to override the persistence request

Asynchronous Output

▲ Asynchronous output support

- Alternate TP PCBs (ALTPCB) messages
- Queued commit-then-send reply messages (IOPCB) that could not be sent back on the original connection

▲ IMS Environment - IMS V7/V8

- IMS application ALTPCB destinations
 - ▶ Specify a destination = tpipe name = client id
- IMS OTMA Exits needed for ALTPCB output
 - ▶ Prerouting Exit Routine (DFSYPRX0)
 - ▶ Destination Resolution Exit Routine (HWSYDRU0)
 - Sample exit provided with IMS Connect

 Requires IMS Connect and IMS V7

Asynchronous Output ...

▲ Remote client environment

- Retrieval of messages is a client application responsibility

- Client program must be written to:

RESUME TPIPE - specify client id

RECEIVE - receive first output msg

ACK - acknowledge receipt of first msg

RECEIVE - receive second output message

ACK ...

- ▶ IRM header specifies:

Asynch request type:

- **Single** - receive one msg and disconnect the socket

- **NoAuto** - receive all available msgs, wait a specified time
and disconnect if no more messages

- **Auto** - receive all available msgs, wait for next message
with no timeout

Time delay value for ACK or Resume Tpipe:

- **x'00'** - default value - 1/4 second

- **x'01' - x'19'** - 1/100 to 1/4 second

- **x'E9'** - no timer is set, no wait occurs

- **x'FF'** - the receive waits indefinitely

UNICODE - PQ47906

▲ **A standardized character coding system that provides a unique number for every character regardless of platform, program or language. (Used by XML and Java)**

▲ **IMS Connect supports**

- Language groups 1,2,3
- UTF-8, UTF-16, UTF-32 and UCS-2 encoding schema

▲ **Note:**

- Data portion of a UNICODE message is NOT translated
 - ▶ IMS application must be able to deal with UNICODE

New fields/flags in the IRM for UNICODE support:

IRM_ES - Encoding schema (UTF-8, UTF-16,...)

IRM_F1 (new flags)

IRM_F1_UC - Unicode message text

IRM_F1_UCTC - Unicode transaction code

UNICODE ...

▲ Input Messages

- Trancode can be sent in as ASCII, EBCDIC, UNICODE
 - ▶ Must be left-justified, 8 bytes, and padded with blanks
 - ▶ Message exit translates tranocode to EBCDIC if needed
- Any IRM or OTMA headers must be sent as ASCII or EBCDIC
- Data portion of message in UNICODE is untranslated

▲ Output messages

- IMS error messages (DFS....) are sent as ASCII or EBCDIC based on the code type in the IRM
- Data portion of message in UNICODE is untranslated

Local Option - PQ45057

▲ Non-TCP/IP connectivity

- MVS Program Call (PC) interface to IMS Connect
 - ▶ Avoids TCP/IP Firewall issues
 - ▶ Provides compatible performance to TCP/IP connectivity
- Defined in the CONFIG file as PORT=(9999,LOCAL,...)
 - ▶ Only 1 local PORT per IMS Connect
- Supports commit mode 1 (send-then-commit)
 - ▶ 10 TPIPEs per IMS

▲ Only supports IMS Connector for Java on S/390, z/OS

- Running on 1- to- N Webspheres
- IMS Connect and Websphere must be in the same LPAR

ACK/NAK Flag - PQ46195

▲ Architected mechanism to allow a remote program to determine if an ACK or NAK needs to be sent back to IMS Connect

- Removes the need for remote programs to query different messages to determine if an ACK or NAK is required
 - ▶ Of value when using Sync_level=Confirm
 - Some messages sent by IMS Connect do not require any acknowledgments
- New flags in the CSM (Complete Status Message) and RSM (Request Status Message)
 - ▶ CSM_ACK_NAK added to CSM_FLG1 possible values
 - ▶ RSM_ACK_NAK added to RSM_FLG1 possible values

Sample Client programs

▲ Sample programs that can be downloaded to validate the IMS Connect install

- JAVA
- COBOL
- ASSEMBLER
- C

<http://www-3.ibm.com/software/data/ims/about/imsconnect/index.html>

Troubleshooting

▲ If you receive errors back in the remote client

- Review IMS Connect address space or OS syslog for associated error messages

▲ Possible return codes received in the remote client

- OTMA - documented in IMS V7 OTMA Guide (SC26-9434)
- TCP/IP - documented in TCP/IP manuals

▲ HWSnnnnnn Error Messages

- Documented in IMS Connect Guide and Reference (SC27-0903)

▲ Some OTMA and IMS Connect messages may refer to TCP/IP return codes

Troubleshooting ..

▲ Recorder Trace

- Shows the message layout
 - ▶ Input message received from the client
 - ▶ Input message after message exit processing
 - Before it is sent to IMS
 - ▶ Output reply from IMS
 - ▶ Output reply after message exit processing
 - Before it is sent to the client

▲ Internal Trace

- Provides information about the activity through the IMS Connect components

Troubleshooting ..

▲ Dump Formatter support

- Formats controls blocks under ISPF IPCS
 - ▶ IMS Connect control blocks
 - ▶ BPE control blocks
- Requires APAR PQ34229
 - ▶ IMS V7 Dump Formatter is used to access the IMS Connect Dump Formatter

Requirements, Futures, Etc.

- ▲ **Timer enhancements**
- ▲ **Security enhancements**
- ▲ **Continued roll-out of Two-Phase Commit support**
- ▲ **Support for new architectures**
- ▲ ...

IMS Connect Summary

▲ **IMS Connect continues opening up IMS to TCP/IP Clients**

- Standard interface
- Defined application protocol
- Comprehensive set of capabilities

▲ **Accessed by the IMS Connectors**

- IMS Client for Java, IMS Connector for Java, ...

▲ **Accessed by user-written programs**

- Documented and well-defined interfaces