A12

Application Design Considerations When Web-Enabling your IMS System Parts 1 and 2

Suzie Wendler

Miami Beach, FL October 22-25, 2001

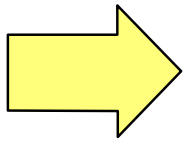
RETURN TO INDEX

Abstract

Initial decisions that are made on how to Web enable IMS oftentimes focus on the connectivity capability, e.g. network type and available toolkits. Ultimately, however, the application and business requirements have to be folded into the equation and the design must accommodate the actions to be taken in the case of a failure. This double session addresses the considerations that should be understood and reviewed when creating or choosing Web enabling solutions for IMS. It details the considerations based on application access (inquiry versus update), message data (3270 attributes versus raw data), access type (synchronous versus asynchronous), IMS transaction type (conversational versus non-conversational), and recovery actions (what to do based on the error type).

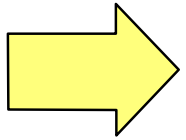
Where Do We Start?

- **Know what is available**
  - Technology, Products, Solutions
- **Define your requirements**
  - Network
  - Architecture
  - Applications
    - Data sensitivity
- **Understand the application design considerations that apply to each solution**
  - Web Browser - Server interaction
  - Server access to IMS



PART 1

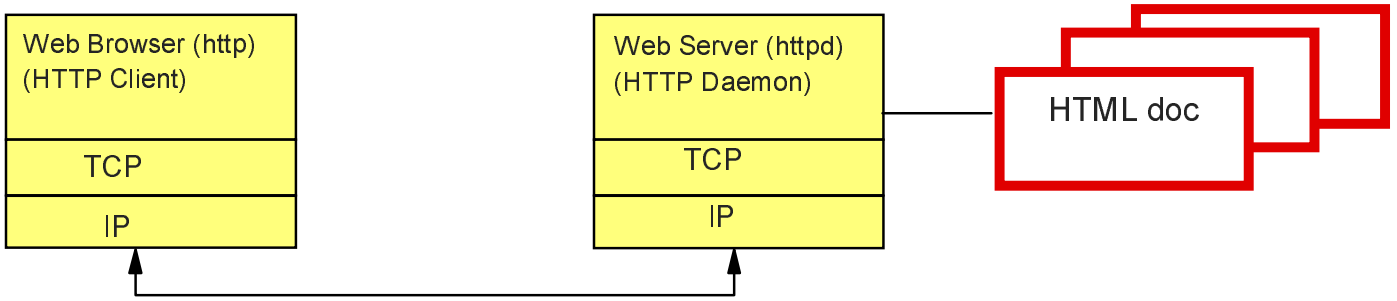
  - Server pgm interaction with the IMS pgm



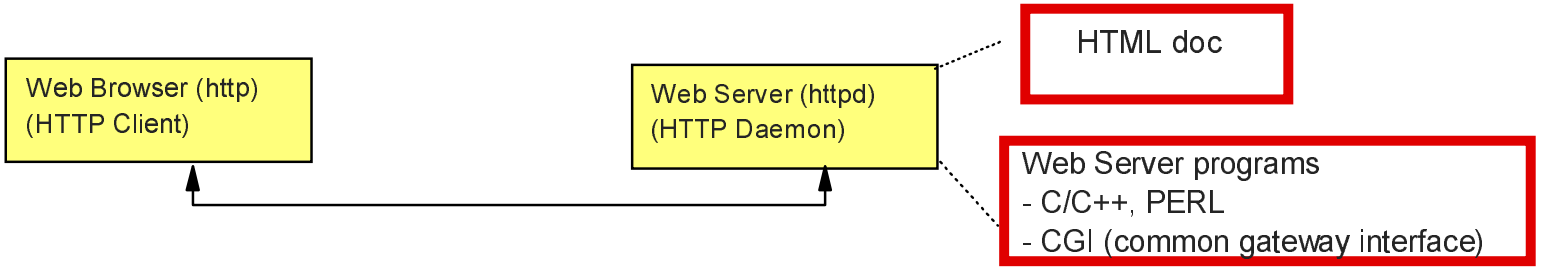
PART 2

A Little Bit of Background and History

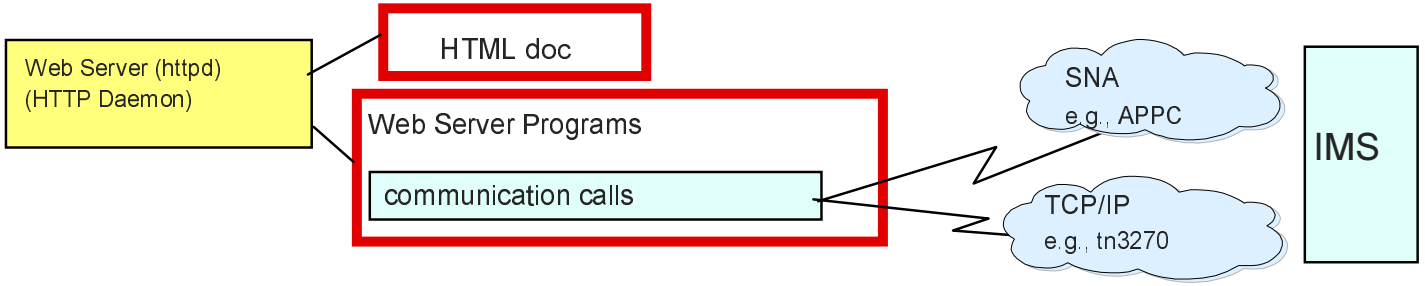
In the Beginning ...



Then there came a need to process data



Then came access to enterprise systems



Followed by The Evolution.... and The Confusion

- **Web Server programs**
  - CGI, vendor server APIs, Java applets and servlets, etc...
- **Web Server environments**
  - Multiple Vendors
  - Different platforms
  - A variety of ways to access the enterprise
- **Programming Languages**
  - C, C++, Visual Basic, Java, etc...
- **Solutions and Toolkits**
  - IMS Web, Net.data, VAJava, etc...

Web Server Program Evolution

- **The support and evolution of web server programs included:**
  - The programming interface
    - How to support input and output messages
    - How to schedule the programs
  - The programming languages that could be used

Web Server Program Evolution ...

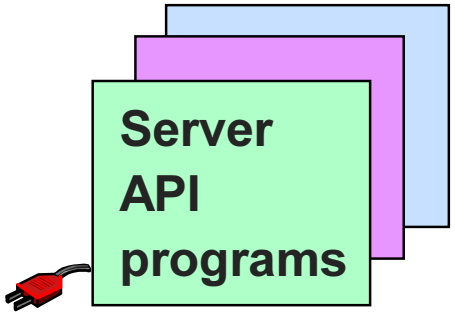
The programming interface

CGI programs

- **Standard interface**
  supported by Web server vendors
  * Simple coding interface
- Different programming languages
- Straightforward implementation on different web server vendors
- Supports lower volumes (thread management per invocation)

Fast CGI programs

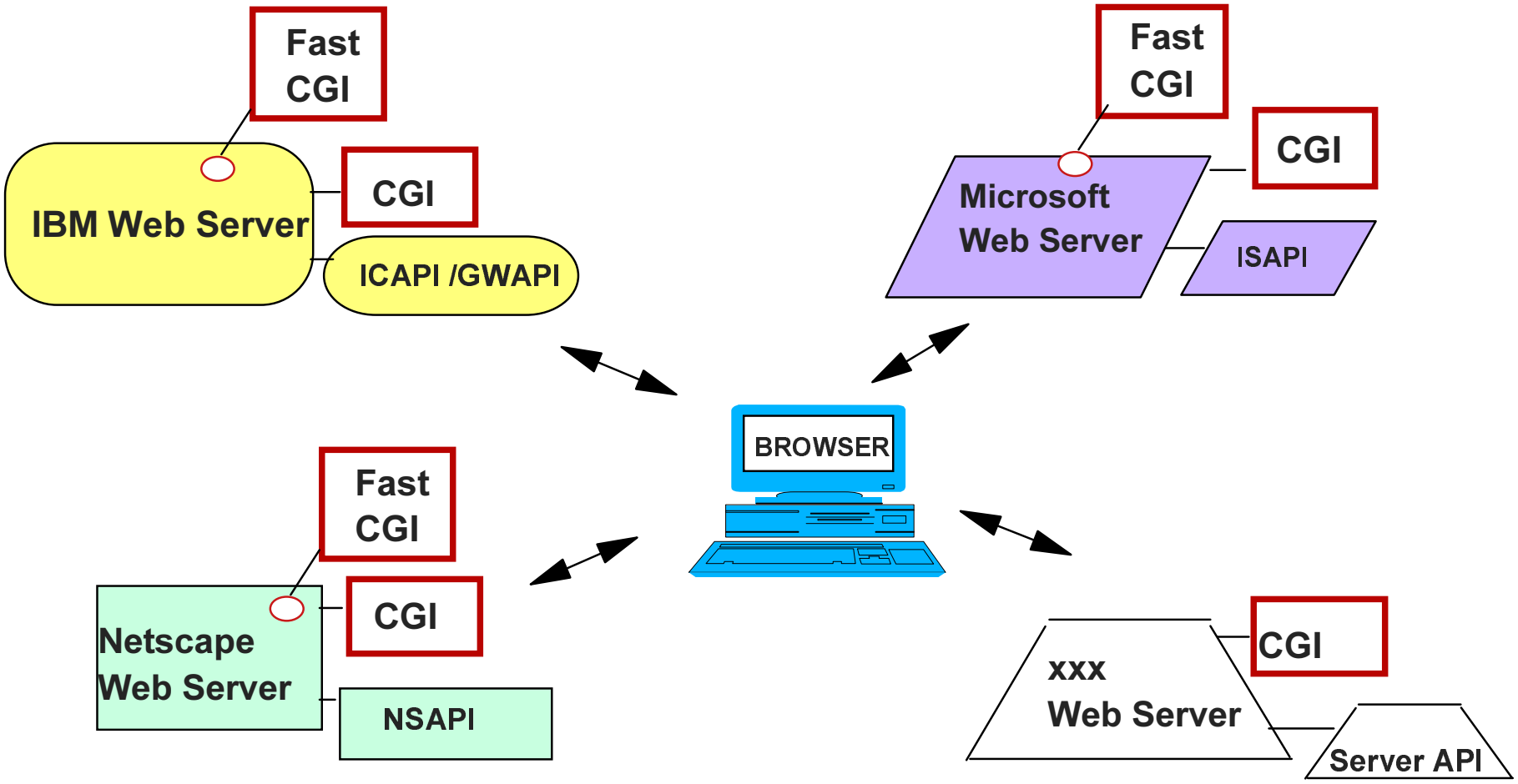
- **Standard interface**
  supported by Web servers that have been extended for this capability
- CGIs must be written in C and linked with the fastcgi C routines (www.fastcgi.com)
- Supports higher volumes (thread management across invocations)



Server plug-ins (code that extends an http server)

- **Unique interface by vendor**
  e.g., ISAPI, NSAPI, ICAPI, GWAPI
  …
- Programming languages supported are based on the vendor
- Migration from vendor to vendor involves rewrites
- Supports high volumes

Web Server Program Evolution...



Web Server Program Evolution...

- **The programming languages**
  - C, C++, Visual Basic, Perl, ActiveX, Java ...
  - Web servers (vendors) defined which they supported
    - Oftentimes based on the platform (UNIX, Windows, OS/390, etc.)

Web Server Program Evolution...

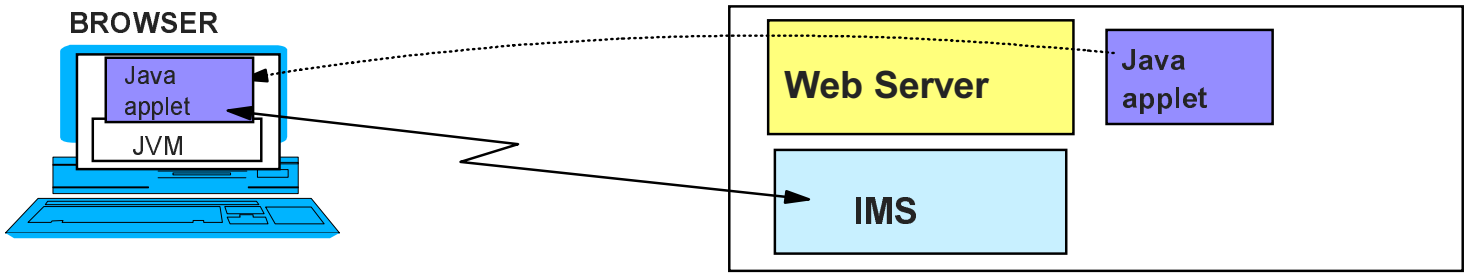
- **Java - the evolving standard**
  - OO Programming Language
    - Popular in the web environment
    - Specification created by Sun
    - Standard and portable
  - Write and compile once -- run anywhere
    - Uses Java Development Kit (JDK) for developing and compiling
      - Source is compiled into machine-neutral bytecode
    - Requires Java Virtual Machine(JVM) to run
      - During execution the bytecode is interpreted and converted into native machine code

Web Server Program Evolution...

Java execution environments

- Applications
  - Require a Java Virtual Machine (JVM) to execute
  - Can run as stand-along programs
  - Can run in a web server (CGI, servlet)

- Applets - little applications that are stored in a web server
  - To execute, they are downloaded to and run in Browsers
    - Most browsers have a JVM
  - Can issue communication calls
    - Can only access the same platform as the Web Server



Web Server Program Evolution...

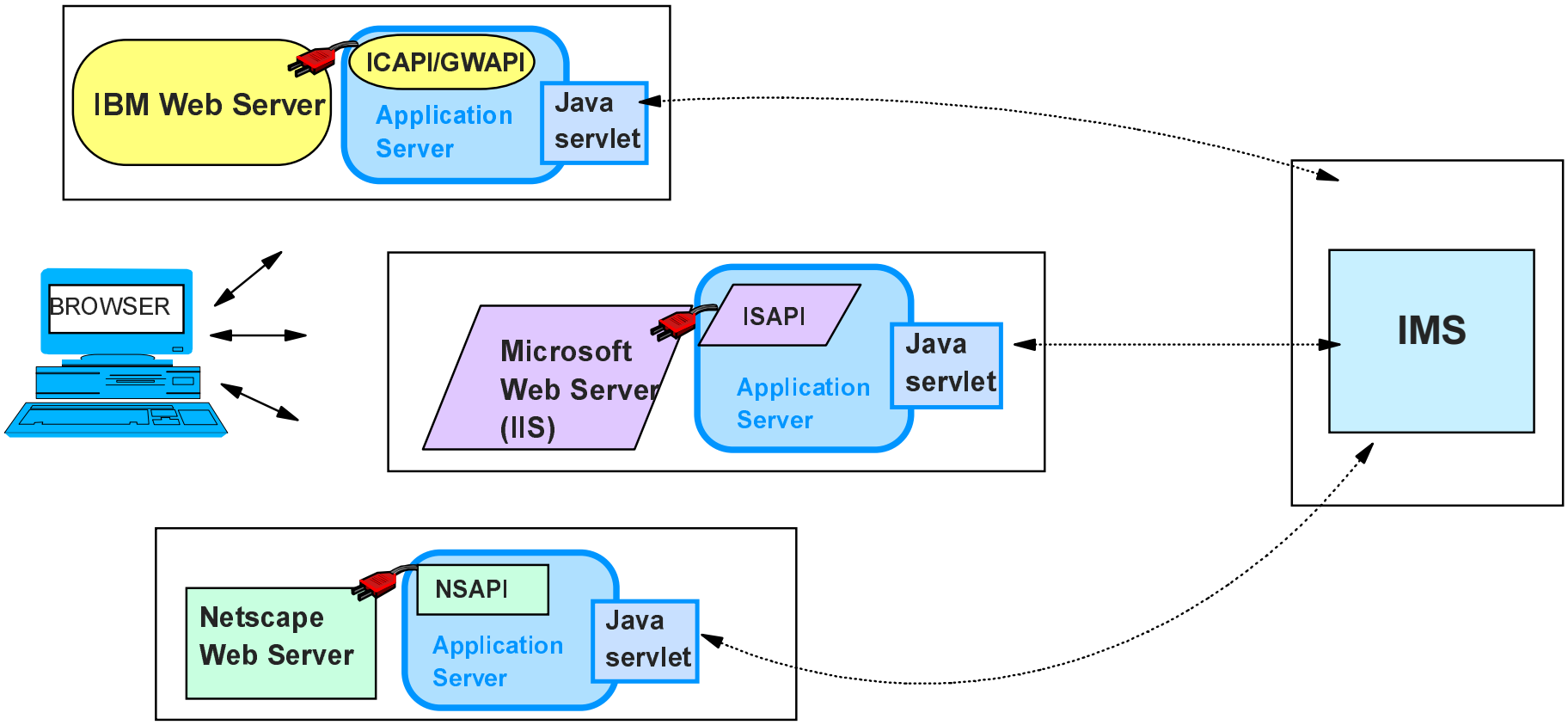
Java servlets - server-side programs

- Java replacement for CGI programs
  - Much faster
  - Can stay alive across HTTP requests
- Java replacement for server plugins (NSAPI, ISAPI, ICAPI, ...)
  - Will be the same on all platforms
  - Safe - cannot crash the web server
- Run on Application Servers ☆
  - Application servers can run on different web servers
    - Special purpose program that uses the vendor-specific API
    - Provide a high-performing standard API for Java programs

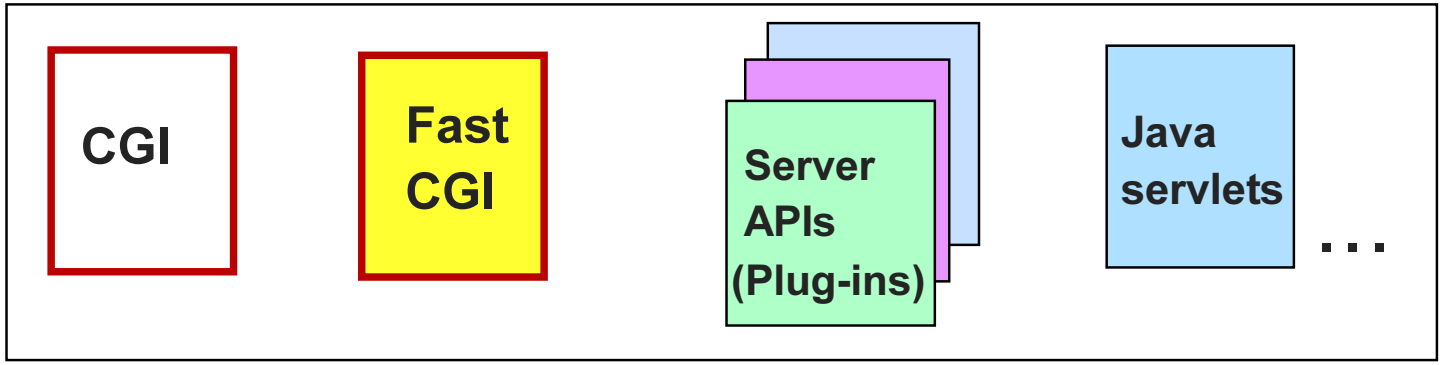
☆ IBM's application server is called the Websphere Application Server

Web Server Program Evolution...

- **Java servlets are written and compiled once**
  - They are executed on an Application Server that is "plugged into" a standard web server



Web Server Program Evolution ..



... And all can be used today

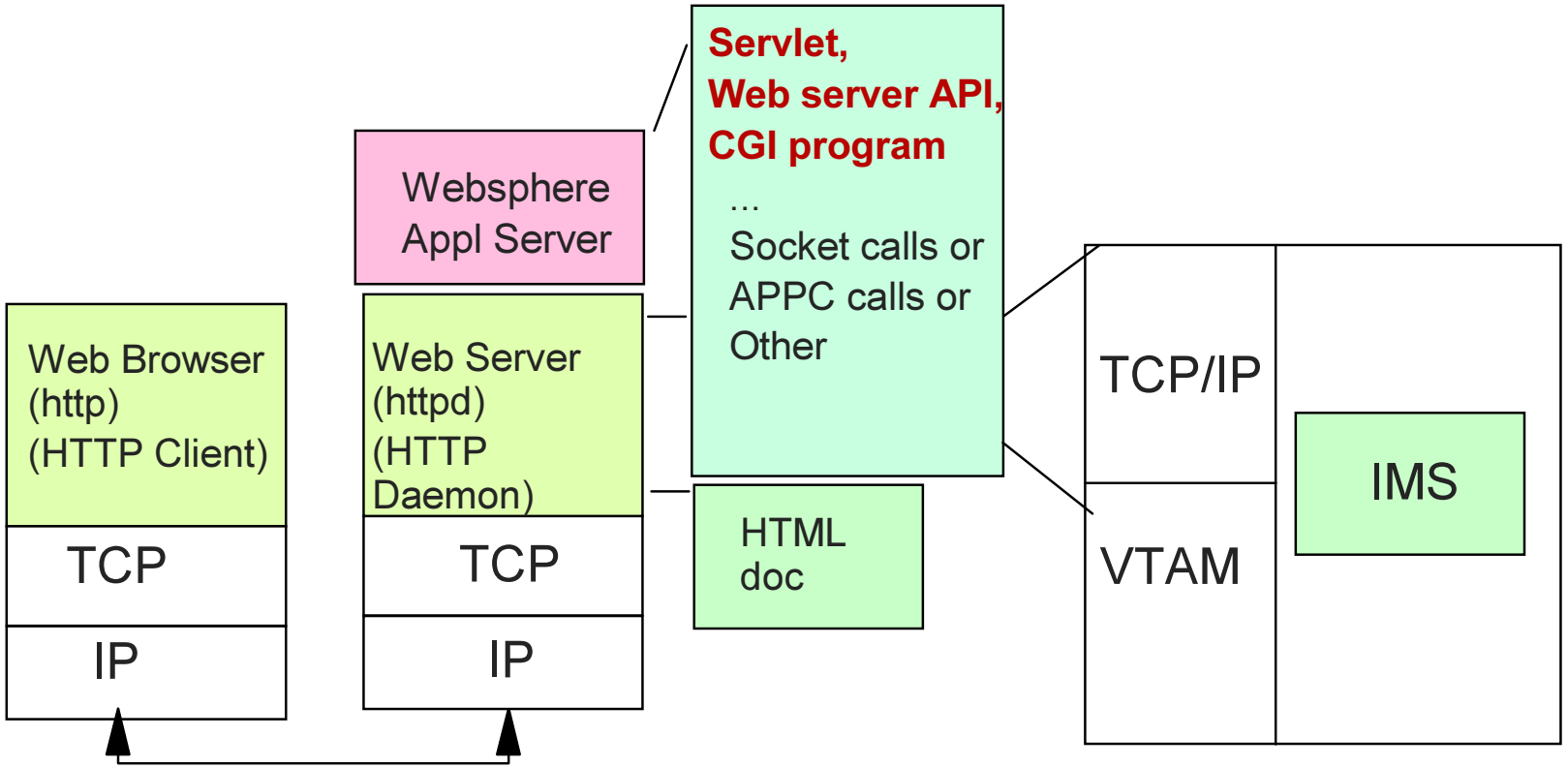
Programming - Strategy

Java, Java, Java - the hot, new language

- Support for a standard toolkit - VisualAge Java
  - Development environment
- Support for a standard high-performance processing environment
  - Websphere
- Support for emerging trends
  - Component technology
  - Enterprise beans
    - EABs, EJBs, etc...

Accessing IMS

Accessing IMS



Web Server Program Considerations:

1. Access IMS using any supported protocol

2. Convert transaction input/output to HTML (parse the data)

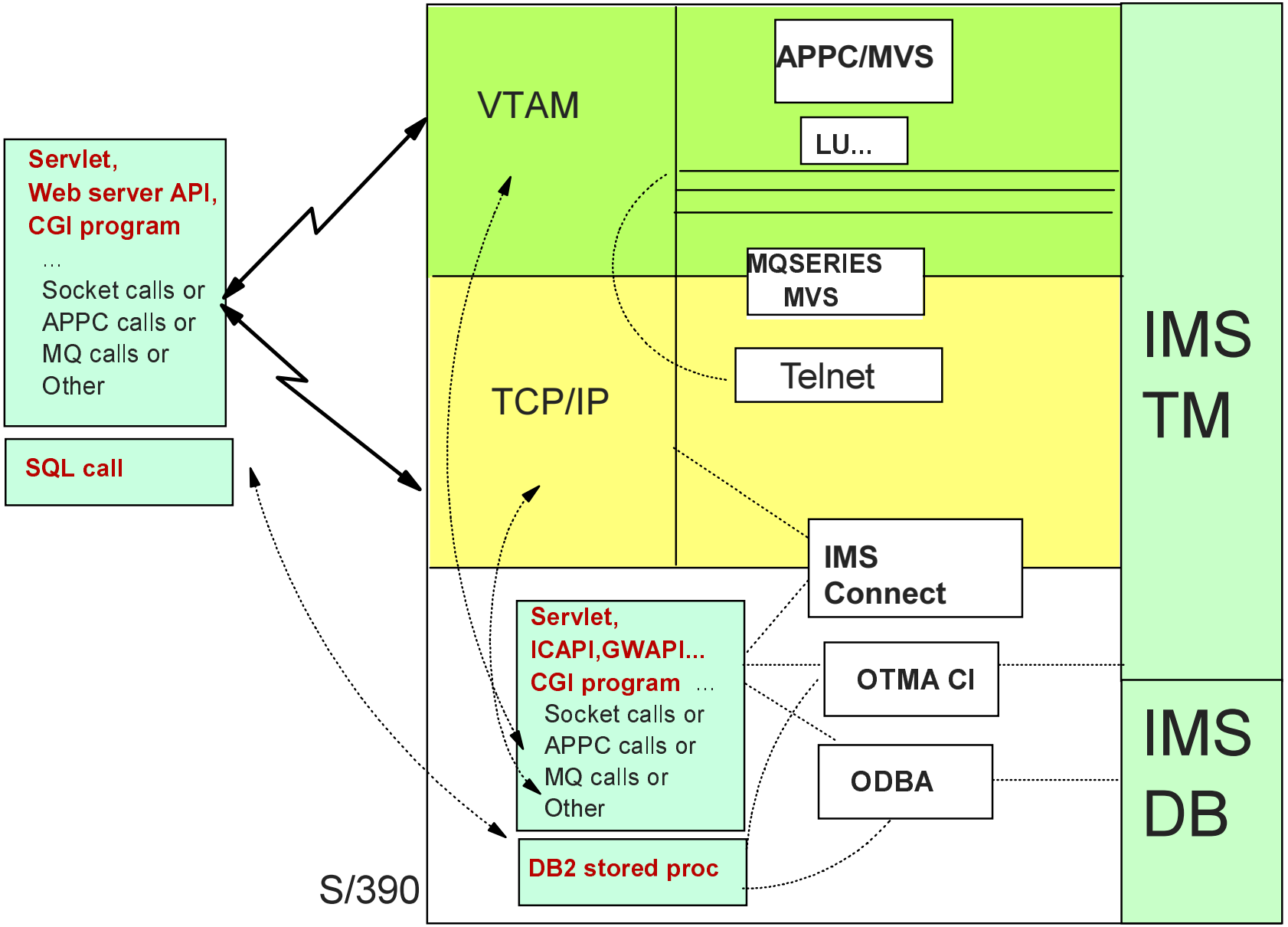
Accessing IMS ...

- **Define the Environment**
  - Network requirements - SNA or TCP/IP
- **Application requirements**
  - Direct connection model vs. Messaging and Queuing model
  - Access to transactions versus direct access to data
  - Inquiry (read-only) or Update
  - Simplicity or extensibility
- **Development requirements**
  - Programming language
  - Skill
  - Cost - Build versus Buy and Modify
    - Toolkits

Define the Environment

- **Where is the Web Server**
  - On the S/390 or on a remote platform?
- **For remote Web Servers**
  - What network is to be used
    - TCP/IP or SNA?

Define the Environment ...



Application Requirements

Direct Connection Model (transactions)

- Characteristics
  - Processing begins only if connections can be established
  - Immediate notification of problems
    - Error indicators sent in the case of failures

- Most popular types of support
  - **3270 emulation - Traditional interface**
    - SNA=EHLLAPI, TCP/IP=TN3270
  - **Program-to-Program support**
    - SNA=APPC, TCP/IP=Sockets
    - Interactive processing
    - Output messages can be sent before/after IMS syncpoint
      - Remote programs can affect whether or not commit occurs

Application Requirements...

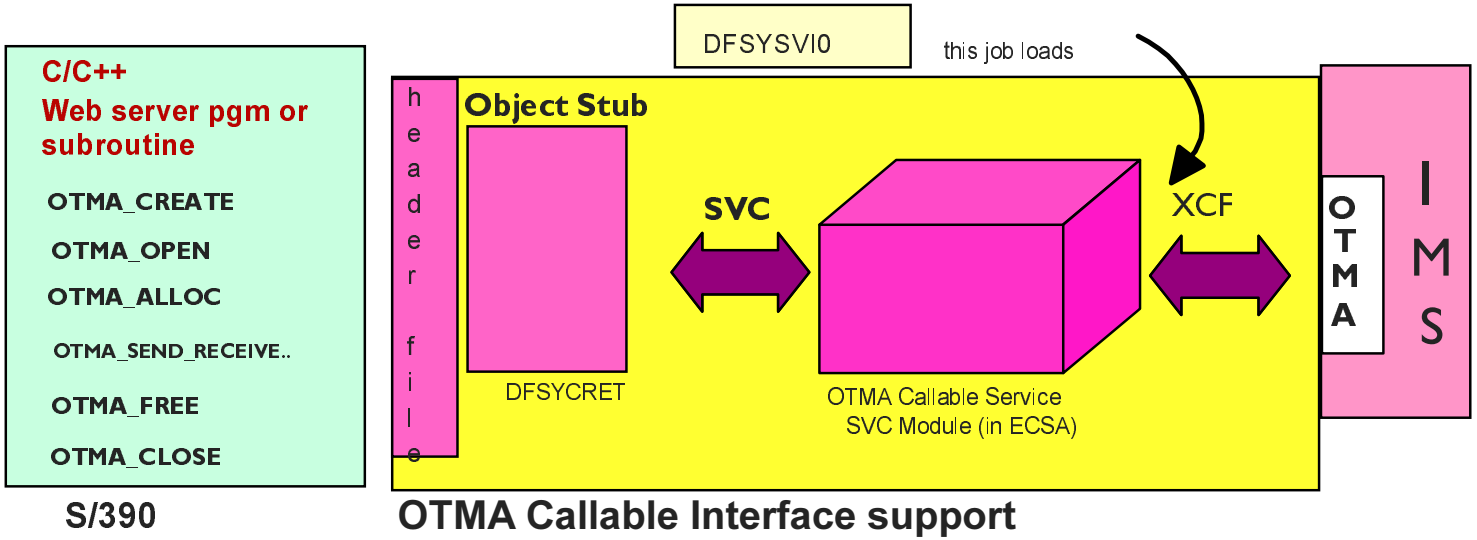
Direct Connection Model (transactions)...

- Considerations
  - Remote programs are sensitive to network type
    - Communication interface APIs are different
    - Switching between SNA and TCP/IP requires re-writes
  - Design should account for connection failures
    - Up front when connection cannot be established
    - During execution

Application Requirements...

Direct Connection Model (transactions) ...

- Additionally, if the web server is on the S/390
  - The OTMA Callable Interface
    - Provides C/C++ programs/subroutines to access IMS transactions

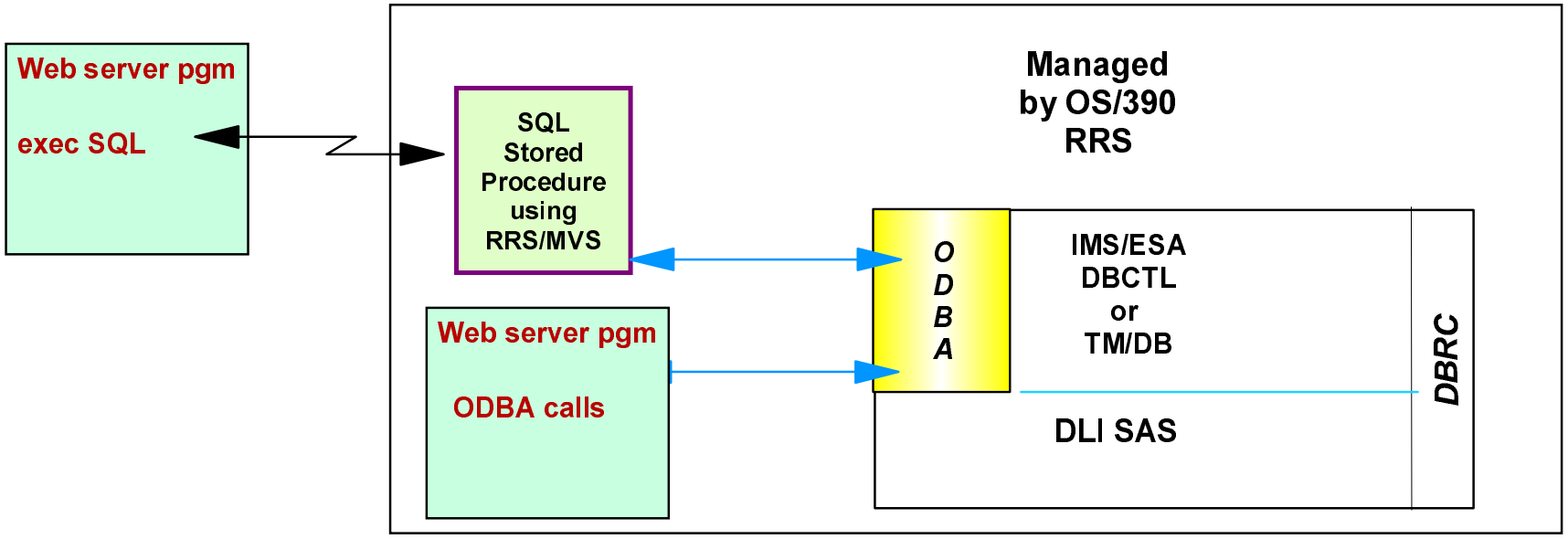


JAVA Wrapper for OTMA C/I can be found in OTMAWWWX which is part of package DB2IMSRX at http://www.s390.ibm.com/nc/download.html.

Application Requirements...

Direct Connection Model (database) ...

- ODBA interface (Open DataBase Access)
  - Programs that issue database calls must reside on the same MVS as IMS



Application Requirements ...

Messaging and Queuing Model (transactions)

- Characteristics
  - Processing occurs whether or not a connection is made
    - Guaranteed delivery of messages (inbound/outbound) when components and/or network are available

- Support
  - MQSeries
    - Remote program is not sensitive to the network type
      - MQ provides its own high-level standard API
      - Same applications can be deployed on TCP/IP or SNA

Application Requirements ...

Messaging and Queuing Model...

- Considerations
  - Deployment of appropriate MQ software on associated platforms
  - MQ is primarily an asynchronous model
    - Design should account for a component/network failure
      - Delayed messages
      - Dead-letter queue messages

Choices

Build

- Create and build your own program
  - Access to IMS can use any available communication protocol

Buy and modify

- End-to-end solutions
  - e.g., IBM Websphere Commerce Suite (Net.Commerce)
    - Catalog and storefront creation, merchandising, relationship marketing, payment processing, etc.
      - www.ibm.com/software/webservers/commerce/
  - Industry-based
- Products and toolkits to help generate the web server program
  - General purpose
  - Web server vendor specific
  - Language specific
  - ...

And Just As Important

- **Priorities**
  - Find a solution now, build quickly -
    Pick a solution that can be used with evolving technology
  - Availability of toolkits -
    Complete end-to-end solutions
  - ...
- **What does your organization consider as "strategic" for <u>your</u> environment?**
  - Industry trends
  - Programming language
  - Network
  - Platform
  - ...

Products, Tools, Toolkits, and Assists

Direct Connection Model (transactions)

- 3270 - Host on Demand / Host Integration Solution / Host Publisher, ResQNet, Jacada
- APPC - IMS Web Templates, Websphere EE Component Broker
- TCP/IP - IMS Connect, VAJava EE IMS Connector for Java
- OS/390 program access via OTMA CI

Direct Connection Model (database)

- SQL access via Classic Connect and Data Joiner
- Stored procedure and OS/390 program access via ODBA

Messaging and Queuing Model

- MQSeries

Additionally - Other Products, Other Vendors

- Neon - www.neonsys.com
- Microsoft - www.microsoft.com
- BEA WebLogic - www.bea.com
- IONA - www.iona.com
- ...

The message - IMS is alive and well !!!

Many solutions continue to be introduced are being provided by IBM and other vendors

The OS/390 Planning Assistant

http://www.s390.ibm.com/os390/wizards/ebiz/

to guide you through IBM solutions

Bookmarks Location: http://www.s390.ibm.com/os390/wizards/ebiz/

IBM United States ShopIBM Support Downloads

Home | Products | Consulting | Industries | News | About IBM

OS/390 Planning Assistant for e-business

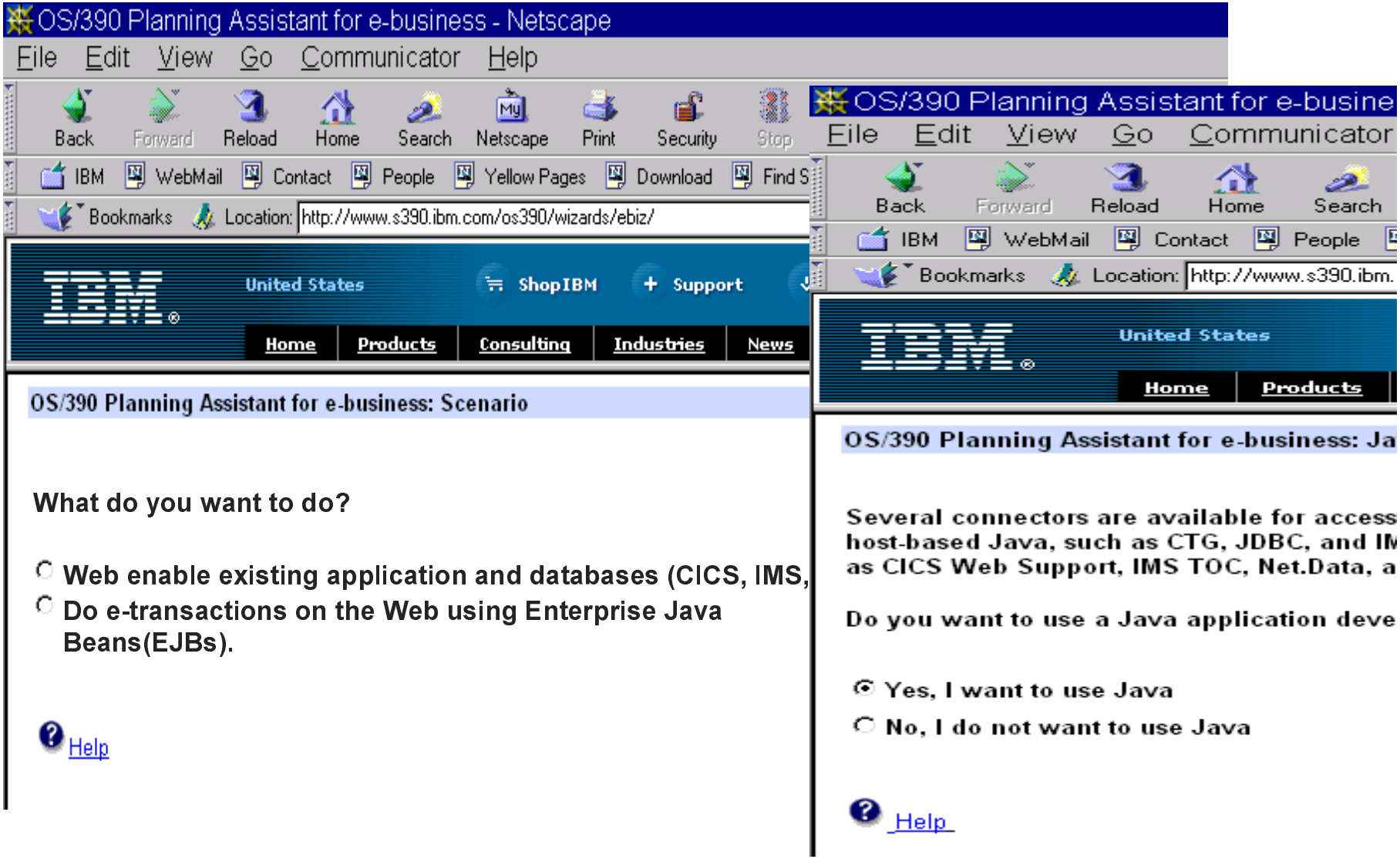
Do you want to know how to use WebSphere products to access CICS, IMS, DB2 or other databases on OS/390? This interactive tool will ask you some questions, and based on your answers, provides an recommended scenario, as well as an introduction and overview of the products and functions you can use for your e-business applications on OS/390.

To get started, click on "Start the Interview", and answer a series of questions about your current system environment and the applications running on OS/390 that you would like to access from the Web. We will use this information to choose a scenario for e-business in your OS/390 environment.

For a list of all the recommended scenarios, click on the Scenario Summary Page.

Please read legal information before starting.

Start the interview


OS/390 Planning Assistant for e-business - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
IBM WebMail Contact People Yellow Pages Download Find S
Bookmarks Location: http://www.s390.ibm.com/os390/wizards/ebiz/
IBM United States ShopIBM Support
Home Products Consulting Industries News
OS/390 Planning Assistant for e-business: Scenario
What do you want to do?
Web enable existing application and databases (CICS, IMS,
Do e-transactions on the Web using Enterprise Java Beans(EJBs).
Help
OS/390 Planning Assistant for e-busine
File Edit View Go Communicator
Back Forward Reload Home Search
IBM WebMail Contact People
Bookmarks Location: http://www.s390.ibm.
IBM United States
Home Products
OS/390 Planning Assistant for e-business: Ja
Several connectors are available for access
host-based Java, such as CTG, JDBC, and IM
as CICS Web Support, IMS TOC, Net.Data, a
Do you want to use a Java application deve
Yes, I want to use Java
No, I do not want to use Java
Help

IBM

Home | Products | Consulting | Industries | News | About IBM

OS/390 Planning Assistant for e-business: Transactions per Day

The transaction load that your Web site experiences in a day
performance and scalability options.

Specify the total number of transactions per day that you expe
press Finish.

- ○ Hundreds
- ○ Thousands
- ◉ Hundreds of Thousands
- ○ Millions
- ○ Don't Know

Help

< Back

Privacy | Legal | Contact

Accessing IMS using the IMS Connector for Java with a Web Server on a non-O...

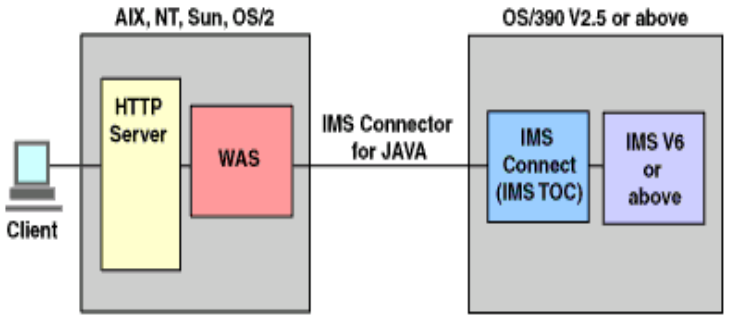
File Edit View Go Communicator Help

Platform

Input Summary

What: Web enable apps & DBs. Browser Control: No
Java: Yes OS: OS/390 V2.7
Subsystem: IMS V6 Upgrade Subsystem: Yes
Web Server: OS/390 Web Server
Transactions Per Day: Hundreds of Thousands LPAR: No

Benefits and Considerations | Variations | Software Requirements | Hardware Requirements | Packaging and Ordering | Security Considerations | Supporting Documentation | Services and Offerings



Description

The IMS Connector for Java (formerly called the IMS TOC Connecor for

Summary (Part 1)

IMS - continues to be the best

Part 2

Accessing IMS

Part 1

- Define the Environment
  - Network requirements - SNA or TCP/IP
- Application requirements and differences
  - Direct connection model vs. Messaging and Queuing model
    - Access to transactions versus direct access to data
- Available toolkits and solutions

Part 2

- Application considerations associated with different solutions
- Design considerations for interfacing with IMS

Starting Simple - 3270 emulation

3270 emulation

- Straightforward and simple
  - IMS is unaware that the access is from the Web
- Traditional IMS communication model

Initial Web approaches to 3270 emulation

- Some browsers packaged 3270 emulators in the browser
- Some software mapped 3270 data streams to HTML

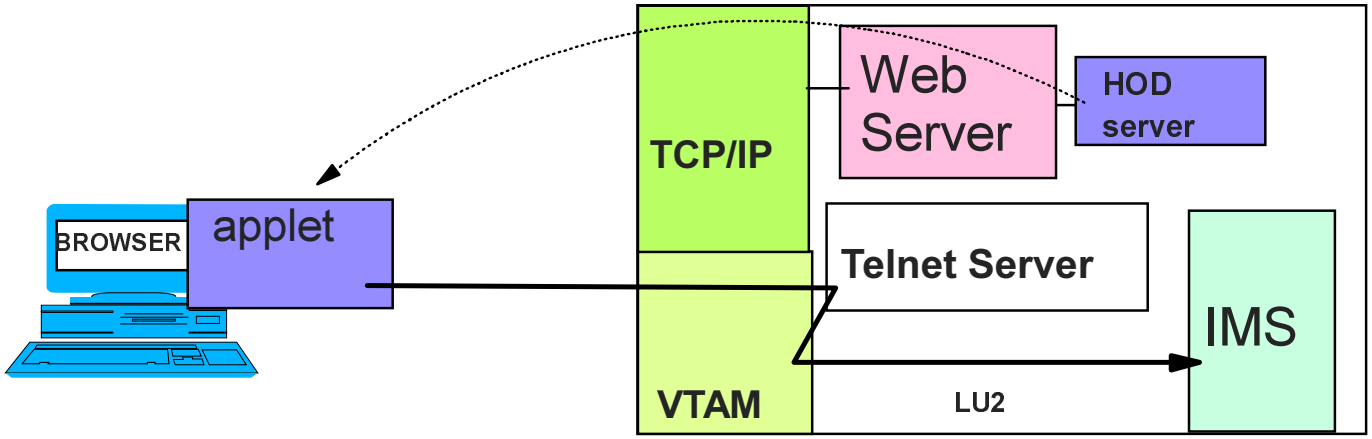
Java-based approach

- Applet provides 3270 emulation at the browser when needed
- Products / tools
  - Host On-Demand - www.ibm.com/software/webservers/hostondemand
  - ResQNet - www.resqnet.com
  - Jacada - www.jacada.com

Starting Simple - 3270 emulation

Host On-Demand

- IBM product - part of Host Integration Solution
  - Downloads a Java applet (includes a TN3270 emulator)
  - Provides GUI functions, screen customization
  - Host Access Class Library API
    - Allows access to the emulator data stream to extend
      - Create customized e-business applications



Host Publisher

- Provides a servlet that provides the TN3270 client support

Starting Simple - 3270 emulation

Benefits

- Straightforward implementation
  - Quick and easy from an IMS perspective
- IMS continues to communicate using SNA LU2 protocols
  - Web browser does sees more than just a "green screen"
    - Customized web screens
- Easy way to web-enable existing 3270-based transactions
  - 3270 attributes and interaction

Starting Simple - 3270 emulation ...

Application Considerations

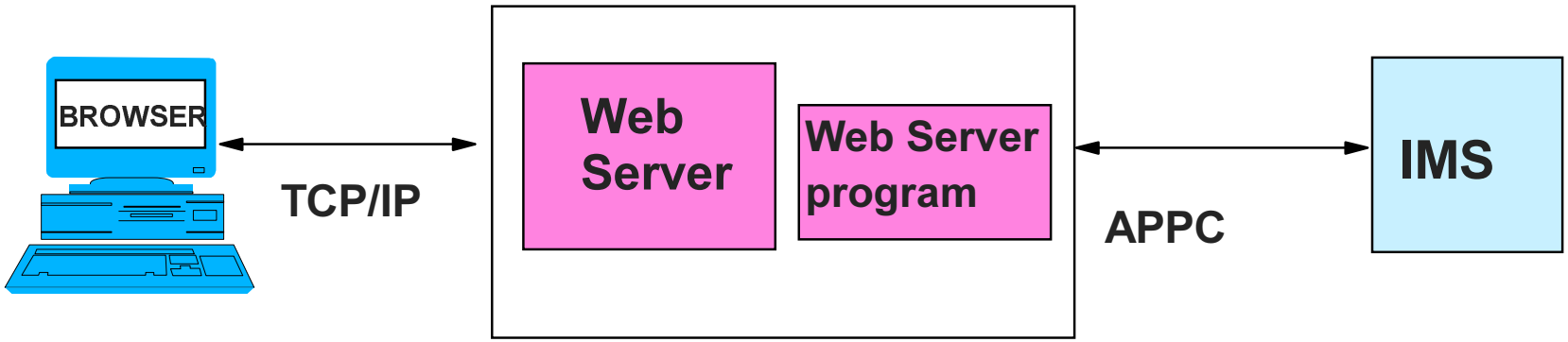
- IMS
  - Applications execute "business as usual"
    - Connection to IMS is SLU2 through a Telnet server
    - Traditional commit model (commit-then-send)
- Remote program
  - Understand emulation package and any restrictions
  - Determine interaction requirements for communication
    - Emulation-only with browser front-end screens versus
    - More complex application interfaces
- Setup and Configuration
  - Telnet uses the concept of pooled LUs
    - Are the IMS applications sensitive to LTERM name?

Starting Simple - 3270 emulation...

- **Why consider any other solution?**
  - Need for greater extensibility and a solution based on something other than 3270 emulation

Direct Connection - SNA (APPC)

- **If your network is SNA or you have a S/390 Web Server then consider an APPC solution**
  - Direct connection model



Direct Connection - SNA (APPC) ...

Benefits

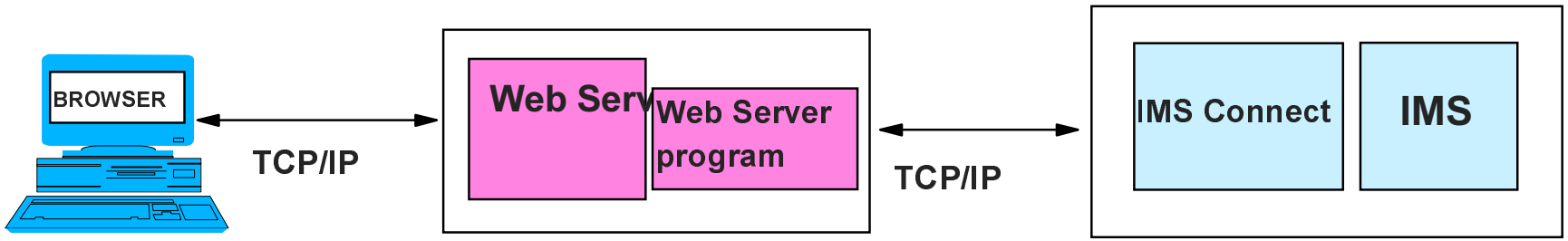
- IMS TM supports APPC natively with a choice of capabilities
  - Implicit - IMS functions as the partner program
    - IMS applications continue to use DL/I calls
  - Explicit - the IMS application program directly controls the the communication sequence using APPC calls
  - Synchronization levels (none, confirm, syncpoint)
  - Commit modes (synchronous, asynchronous)

- The remote program uses a standard well-defined interface
  - Connectivity using APPC has been available and used in production environments for many years

- There are several existing solutions available from IBM and other vendors

Direct Connection - TCPIP

- **If your network is TCP/IP or you have a S/390 Web Server then consider a TCP/IP solution**
  - Direct connection model
    - IMS Connect
      - Provides the IMS support for TCP/IP sockets
        - Generalized interface that can be used by any sockets program
      - Supports a variety of connectors (sample programs and tools)



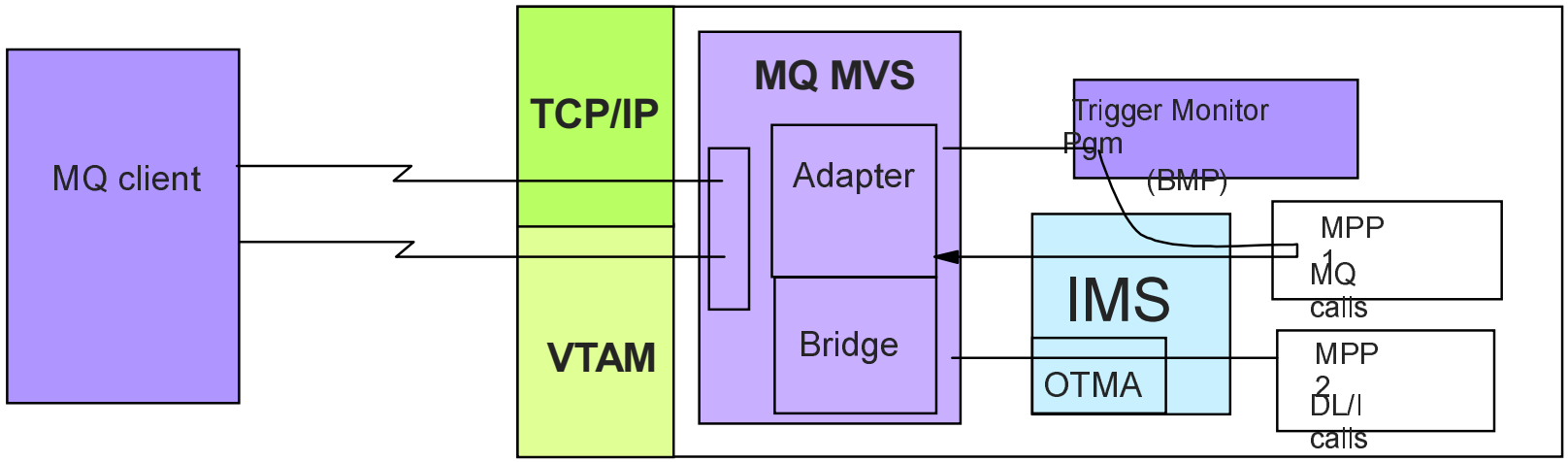
Direct Connection - TCP/IP ...

Benefits

- IMS Connect provides the capability for existing IMS applications to be invoked using standard TCP/IP socket calls
  - The capability is flexible and extensible for capacity and performance requirements
  - Exit interfaces are provided for modification and tailoring to a specific environment's needs
  - IMS application programs do not have to be modified
    - Continue to use the DL/I call interface
- Remote Programs
  - Have a direct connection capability to the IMS environment
  - Are provided with a documented, standard interface

Message and Queuing - MQSeries

- **If you need a messaging and queuing model solution consider MQSeries**
  - Message delivery even when the connection is not available
    - MQ Adapter for IMS
      - Uses the External Subsystem (ESS) interface
    - MQ Bridge for IMS
      - Uses the OTMA interface



Message and Queuing - MQSeries...

Benefits

- MQSeries provides a programming interface that can be deployed across multiple platforms on different types of networks
- Adapter
  - Uses the ESS interface
    - The IMS application uses explicit MQ calls to get/put messages in MQ with syncpoint coordination in IMS
      - Calls to MQ, DB2 and IMS in the application are considered one UOW
- Bridge
  - Uses the OTMA interface
    - Coordinates message transfer between the MQ queue and the IMS message queue
      - Allows the application to use DL/I to access the messages

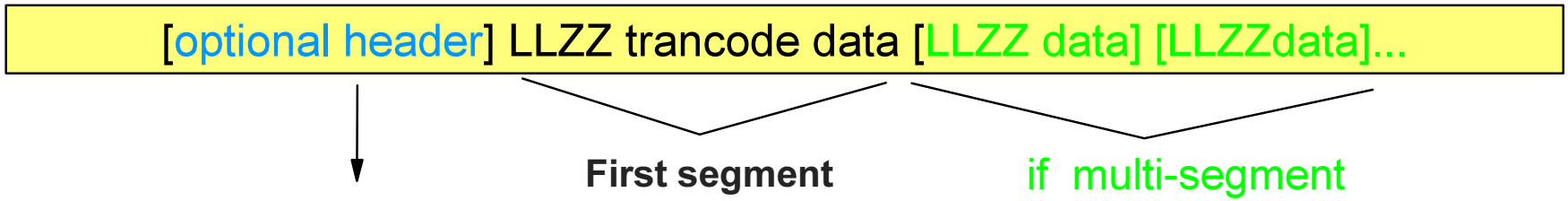
Application Design

- **Design considerations that pertain to:**
  - APPC applications
  - OTMA clients
    - IMS Connect for TCP/IP socket applications
    - MQSeries,
    - OTMA CI
    - ...

Note: the subsequent pages provide overall design considerations when comparing the different protocols/clients. They do not go into the details of application design.

Messages

Input Message Format



Headers

- Unique to Application environment
  - APPC - parameters in the Allocate verb
  - OTMA CI - parameters in the calls
  - MQSeries - MQIIH
  - IMS Connect - *IRMREQ*, *SAMPLE*, ...
- Shield the remote application from having to understand and create IMS message headers
- Provide a way to influence the interaction with IMS
  - Commit mode and synclevels, Overrides (Iterm, etc.), Timeouts...

Messages ...

IOPCB output reply messages

- Single segment: ISRT IOPCB (LLZZ data)
- Multi-segment: ISRT IOPCB (LLZZ data), ISRT IOPCB (LLZZ data)...
- Multiple messages: ISRT - PURGE, ISRT - PURGE, ...
  - APPC
    - Synchronous: First message is sent synchronously, subsequent messages are sent asynchronously
    - Asynchronous: All messages are sent asynchronously
  - OTMA
    - Send-then-commit: Messages are sent as one multi-segment message
      - Purge is ignored, only one output message per commit scope
    - Commit-then-send: Messages are sent as separate messages

Data Sensitivity

MFS is not invoked

- Remote programs send/receive data in the "raw" form used by the IMS application program in the IOAREA
  - Determine what this data looks like

3270 attributes

- Understand how they are used, if at all, by the IMS application
- If necessary, they can be sent as data in the data stream
  - Remote program(s) will need to deal with this
    - E.g., highlight, color, re-display of data, ...
  - Can add complexity to the remote program
- May be a reason for a specific application to be web-enabled using a solution like Host On Demand

Data Sensitivity ...

Sensitivity to MOD/LTERM name in the IOPCB

- Default LTERM name
  - APPC - partner_lu name
  - OTMA - Tpipe name
- Default MOD name
  - blanks
- To override defaults
  - APPC - implement DFSLUEE0 (LU 6.2 Edit Exit Routine)
  - OTMA - specify value in message prefix (differs per OTMA client)
    - OTMA CI: in the verb parameters
    - MQ: in the MQIIH header
    - IMS Connect: in the message exits or in the header

Timing Considerations

Timing out

- How long should a process wait?
  - Should a timeout be set?
  - What action should be taken when a timeout occurs?

- Setting timeout values

  - APPC

    - In IMS: APPCIOT value in DFSDCxxx
      - Times out waits on the IMS side

    - In remote program, this is based on
      - Verb used: receive, prepare_to_receive, etc.
      - Specific implementation: post_on_receipt, blocking wait, etc.

Timing Considerations ...

- **Timing out ...**
  - Setting timeout values ...
    - OTMA
      - OTMA CI
        - Verbs that wait have an ECB (event control block) parameter
        - Caller must check ECB and wait for it to be posted before inspecting the return code and output fields
        - Sample wait routine, DFSYCWAT, to wait on ECB
      - MQ program - MQGET with timeout or wait interval
      - IMS Connect - timeout value in the configuration
        - IMS Connect client program - timer value in header

Synchronization levels

- **These levels control the interaction between the IMS environment and the remote/calling program**
  - **None** - assumes the partner received the message
    - No acknowledgement required
  - **Confirm** - requests acknowledgement of message receipt
    - Allows greater integrity and interaction
  - **Syncpoint** - implements the support for distributed commit
    - Ensures all partners go through commit/backout
    - Supported by APPC and OTMA in IMS
      - Ability to use support depends on environment of calling program

Commit Scopes and Implications

Asynchronous (APPC) - Commit-then-Send (OTMA)

- Reflects traditional IMS processing model
  - Message sent as a result of a successful commit
  - Input and output messages are enqueued

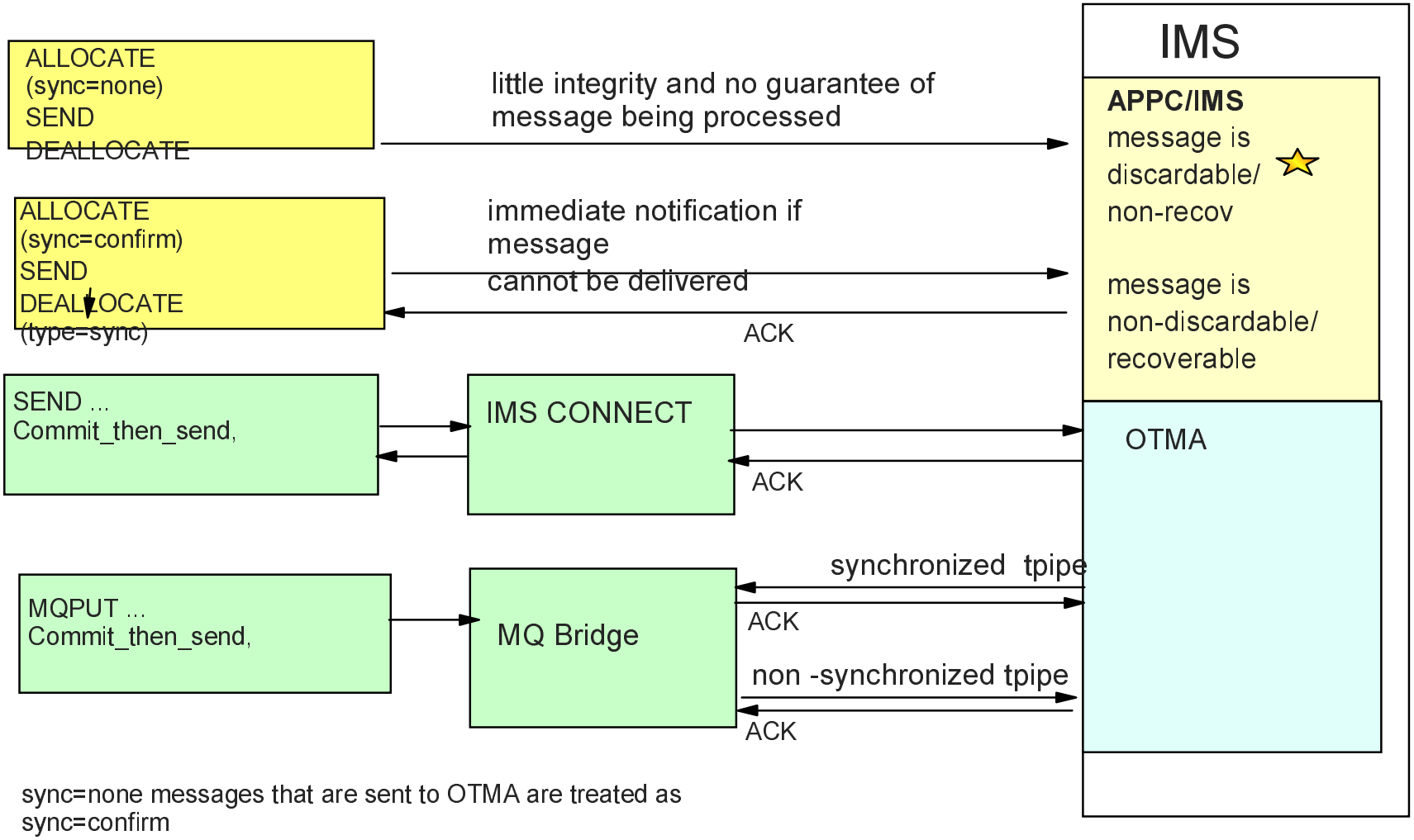
- Has restrictions on transaction types
  - APPC requests cannot access:
    - Response mode, Conversational, or IFP transactions
  - OTMA requests cannot access:
    - Conversational or IFP transactions

- Does not require an output reply

- Is used in conjunction with synchronization level
  - APPC supports sync level of NONE and CONFIRM
  - OTMA enforces sync level CONFIRM

Commit Scopes and Implications ...

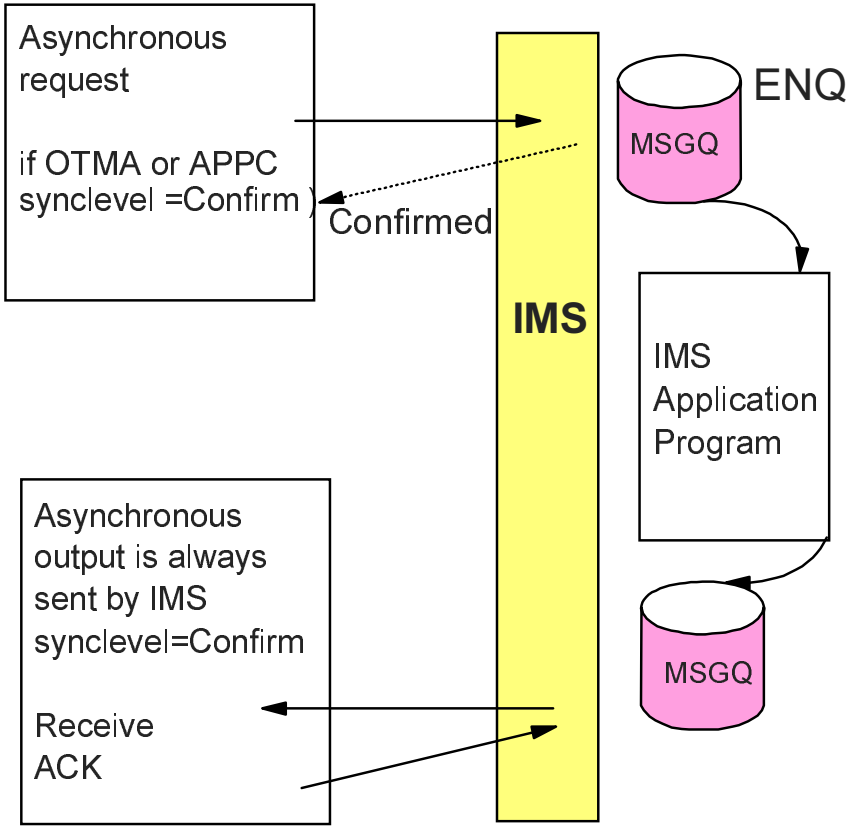
Asynchronous (APPC) - Commit-then-Send (OTMA) ...



sync=none messages that are sent to OTMA are treated as sync=confirm

Commit Scopes and Implications ...

Asynchronous (APPC) - Commit-then-Send (OTMA) ...



Normal Processing	System Failure	Session Failure
Input message is queued	Input message is reprocessed if non-discardable	Once message is received, session failures do not affect processing
Transaction is scheduled	Transaction is backed out - non-discardable messages are reprocessed	
Output messages are always sent synclevel=Confirm and require an ACK for IMS to dequeue the message		

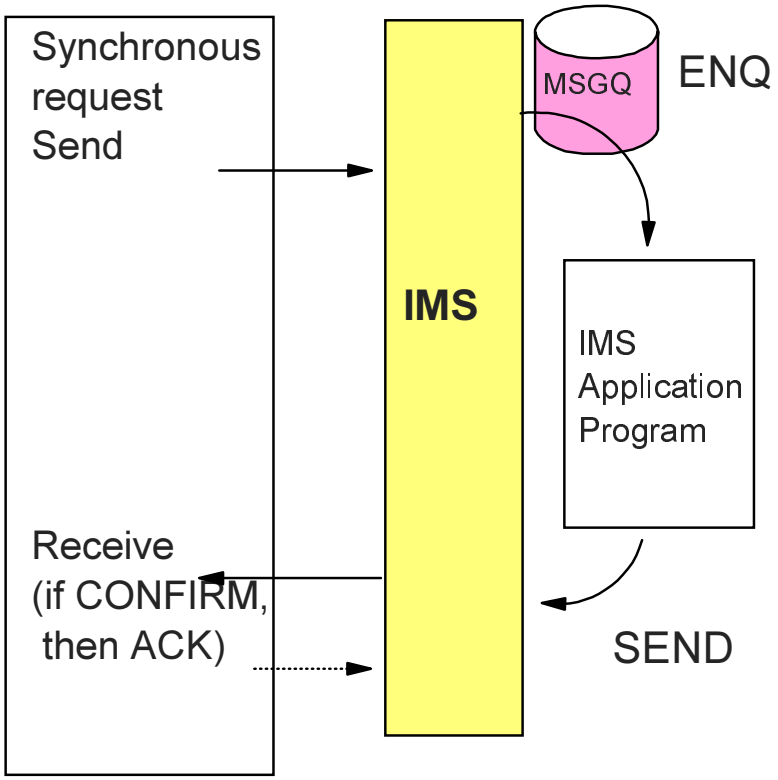
★ discardable/non-discardable conditions apply to APPC support

Commit Scopes and Implications ...

- ***Synchronous** (APPC) - **Send-then-Commit** (OTMA)*
  - APPC/OTMA can access all transaction types
  - Remote program waits for a reply
    - Terminating a transaction without a reply can result in the remote program receiving a DFS2082 message
  - IMS sends the output reply before commit
  - When used with synchronization level = confirm
    - Remote program controls when/if the commit occurs in IMS
      - Impacts dependent region occupancy and database locks
  - Must be used if synchronization level = syncpoint

Commit Scopes and Implications ...

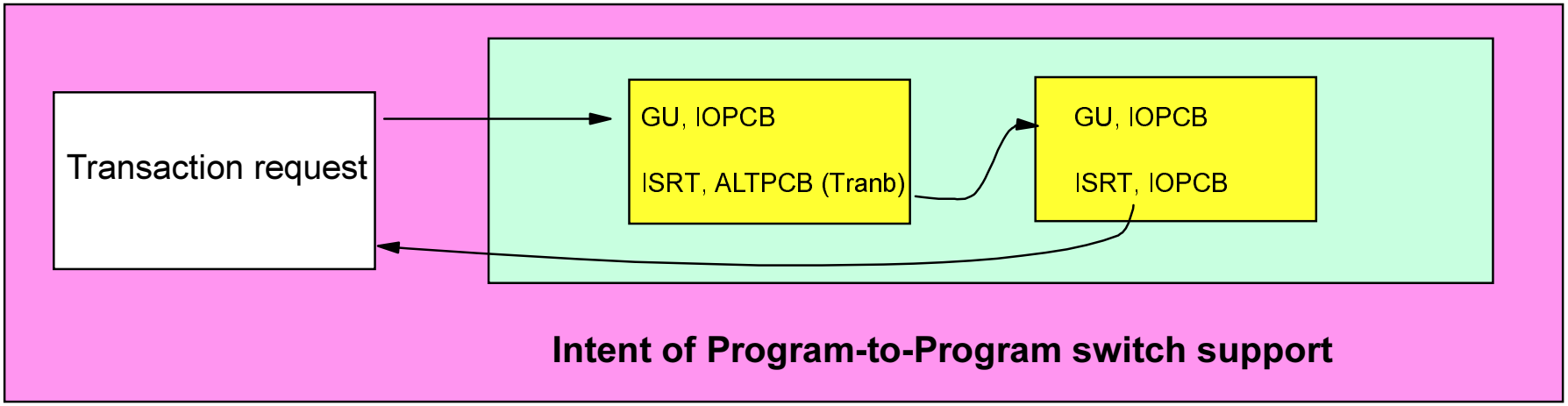
◭ *Synchronous (APPC) - Send-then-Commit (OTMA) ...*



Normal Processing	System Failure	Session Failure
Input message is queued	Input message is lost	
Transaction is scheduled	Input Message is discarded	Abend U0119 or Message Control Error Exit (DFSCMUX0) can decide on COMMIT/ABORT ** (If COMMIT is chosen, msg sent asynchronously)
Prior to commit: Send output - If Confirm, wait for ACK - if None, or when ACK is received, proceed to commit - if NAK, U0119 abend	Transaction is backed out, and discarded	

Program-to-Program Switches

Transfer responsibility of replying to the IOPCB



- IMS transaction chain
  - Can involve multiple transactions e.g., A -> B -> C ....
  - Can be sent across an MSC link

- The Remote transaction request can be
  - Asynchronous / Commit-then-send
  - Synchronous / Send-then-commit

Program-to-Program Switches ...

Asynchronous conversations/commit-then-send requests

- Remote program design accounts for a reply to be sent asynchronously or expects no reply

Synchronous conversations / send-then-commit requests

- Remote program waits for a reply that is sent back via the IOPCB

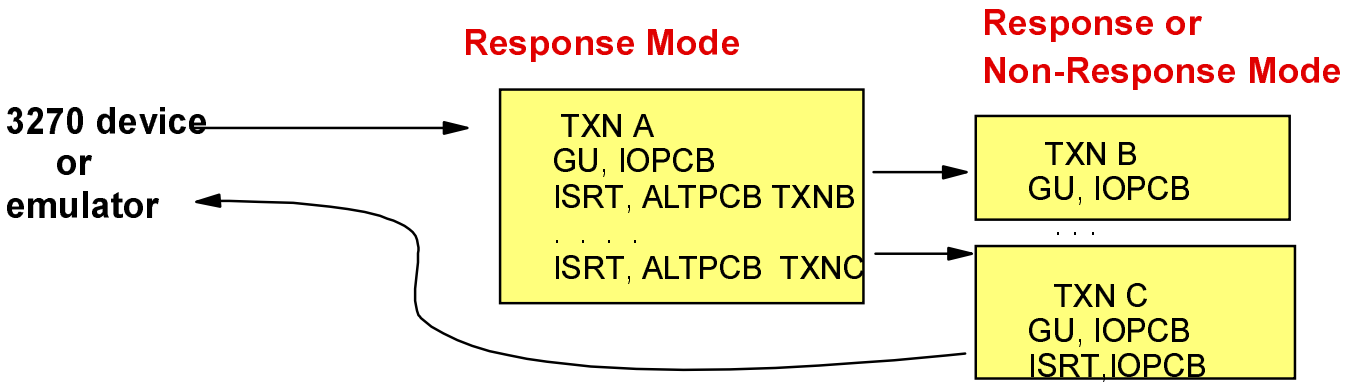
- Considerations:
  - Are the ALTPCBs defined as express?
    - IOPCB replies from express PCB trans do not satisfy the synchronous APPC/OTMA request
  - Does the receiving tran spawn more than one transaction?
    - Are the switched-to transactions response/non-response mode?

- Depending on the protocol used, behavior may differ

Program-to-Program Switches ...

Pgm-to-pgm switch and synchronous requests

- **3270 devices/emulators**
  - Synchronous equates to response-mode



Remote Device

A. Receives response message from any spawned transaction that responds to the IOPCB

OR

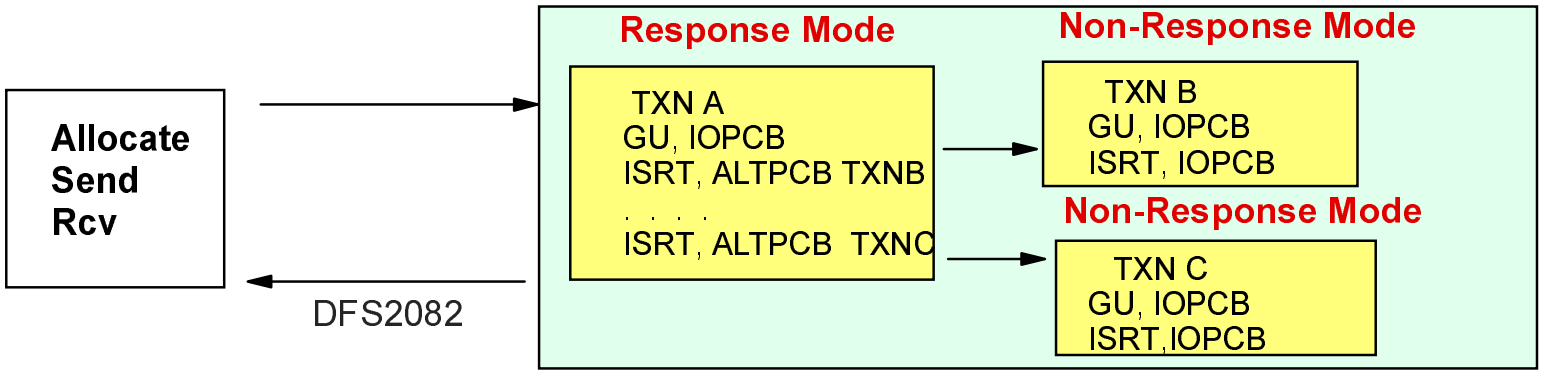
B. HANGS, if no transaction responds to the IOPCB
   - DFS2802 sent only if TXN A does not respond AND does not spawn another transaction

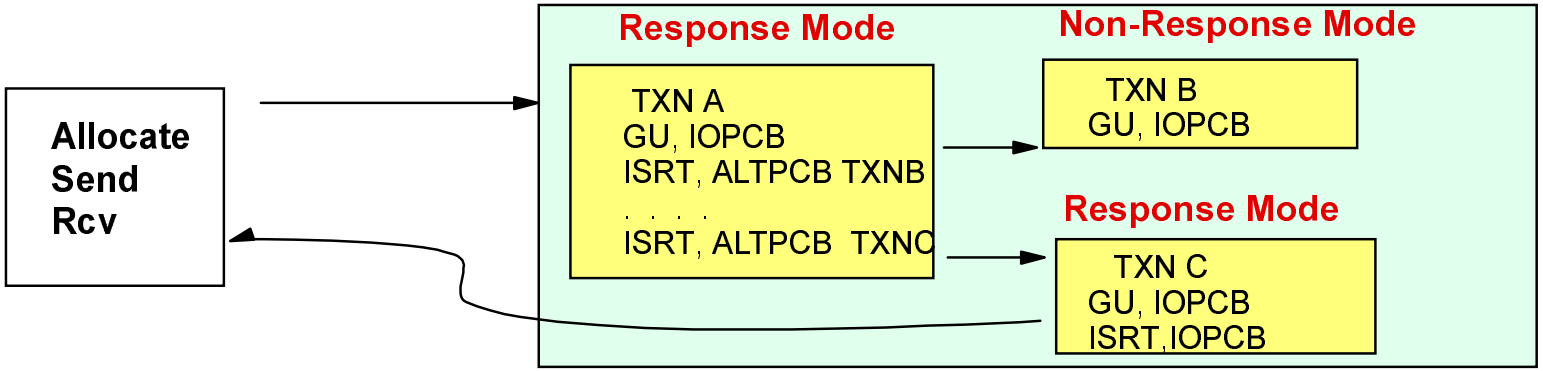
Program-to-Program Switches ...

- **Pgm-to-pgm switch and synchronous requests**
  - **APPC - based on the APPCASY=Y | N (PQ17309/PQ19930)**
    - Relies on accurate transaction specification
      - Response mode versus Non-Response mode
      - Way to control which spawned transactions are eligible to reply to the synchronous request
    - Specified in DFSDCxxx
      - Global specification

Program-to-Program Switches ...

- **Pgm-to-pgm switch and synchronous requests...**
  - APPC: APPCASY=Y
    - Responsibility for synchronous reply is given to response-mode transaction, else DFS2082

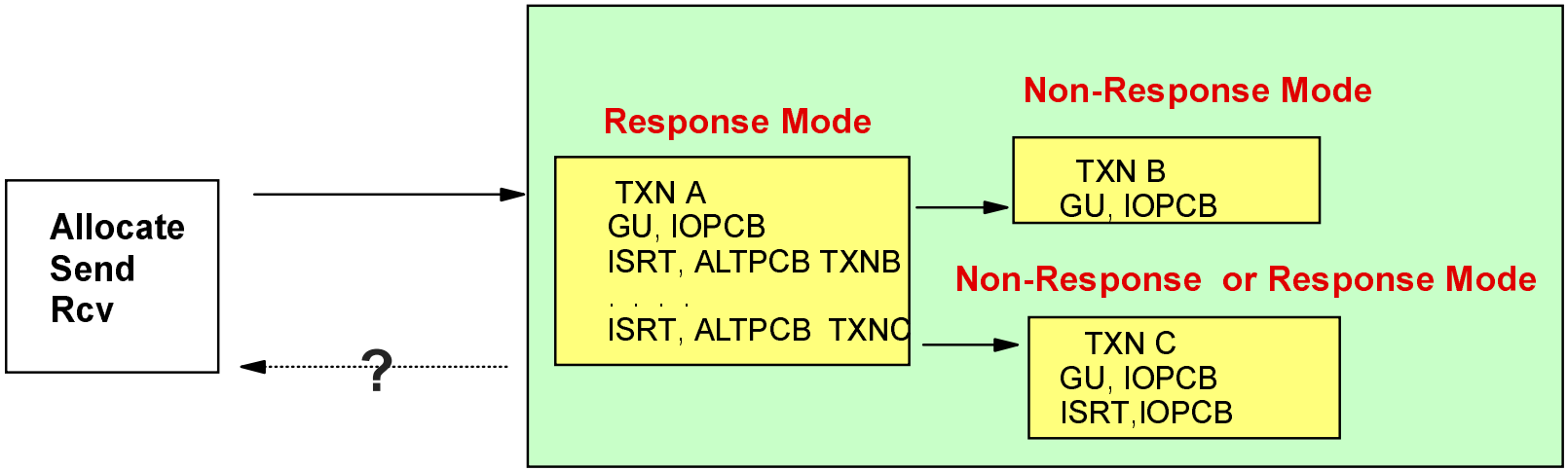




Program-to-Program Switches ...

Pgm-to-pgm switch and synchronous requests

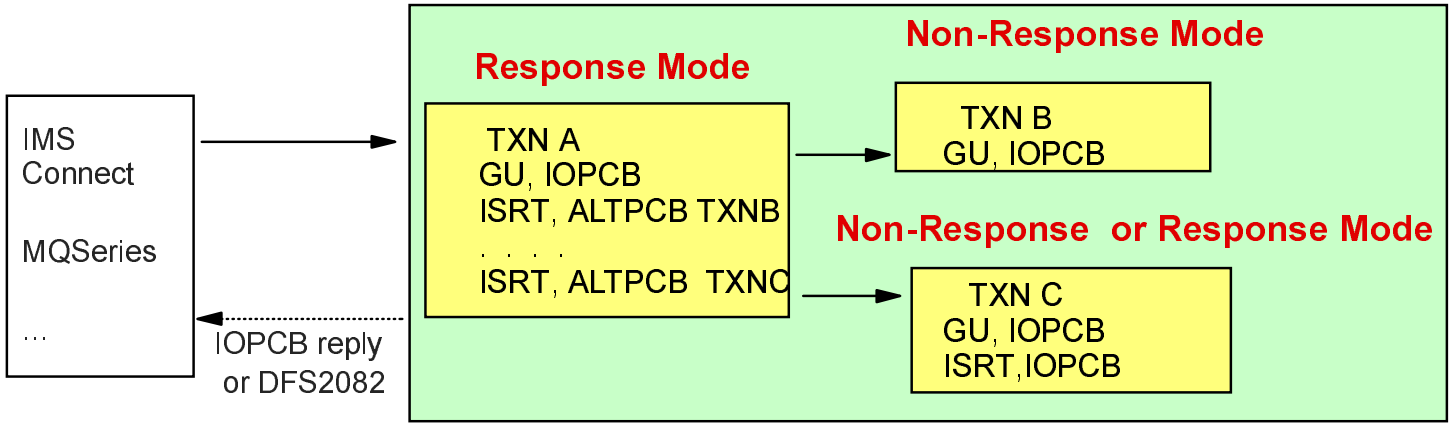
- APPC: APPCASY=N
  - Responsibility for the synchronous reply is given to the first spawned transaction that goes through commit
  - Need PQ37780



The result will be one of the following:
- Without PQ37780, if TXNB completes first, then the request will hang since there is no iopcb reply
- With PQ37780, if TXNB completes first, then a DFS2082 is sent
- If TXNC completes first, the IOPCB reply is sent

Program-to-Program Switches ...

- **Pgm-to-pgm switch and send-then-commit requests**
  - **OTMA**
    - Need PQ23360(V6) / PQ23213(V5)
    - Responsibility for the synchronous reply is given to the first spawned transaction that goes through commit



IMS Conversational Transactions

IMS Conversational transaction processing

- Access to IMS Conversational transactions is supported
- Remote program keeps track of the interaction
  - Maintains synchronous / send-then-commit requests
  - Follows the rules provided by the environment
    - APPC and OTMA clients differ
    - Cannot use certain commands to control the conversation, e.g., /HOLD, /RELEASE, etc.

IMS Conversational Transactions ...

APPC

- Straightforward support
  - IMS keeps track of both the IMS and the APPC conversation
    - Conversation state and conversation id
- Remote program must maintain a synchronous conversation
  - Each send/receive invokes an iteration of the IMS conversation
- IMS Conversation is terminated either by:
  - Remote program terminating the APPC conversation with a deallocate
  - The IMS transaction putting a blank trancode in the spa

IMS Conversational Transactions ...

OTMA

- IMS
  - Provides information to the OTMA client (MQ, IMS Connect)
    - Conversation state and conversation id
  - Relies on the OTMA client / remote program to manage the conversation
- OTMA client can choose to externalize this to remote programs
- Remote programs
  - Must use Send-then-Commit requests
  - Check for conversation state
  - Save and pass back the conversation id

Design for Failure

Direct Connection Model (transactions)

- Designing for failure (when transactions in IMS do updates)
  - Did the commit occur in IMS?
    - Depends on
      - Commit mode (commit-then-send vs send-then-commit)
      - Synchronization level (none vs confirm vs syncpoint)
      - Whether any output messages have been received
    - Possible actions for the web/client program
      - Act on error indicators
      - Provide the ability to send an inquiry
  - On the IMS side - review exit routines
    - DFSCMUX0 - can choose to retain the output reply on a connection failure and send it later/elsewhere
    - DFSNDMX0 - can choose to preserve the input message when the IMS application abends

Designing for Failure ...

- **Messaging and Queuing Model...**
  - Designing for failure
    - Determine what messages could go to the dead-letter queue
    - Decide on specifying timeout values
    - If a web server program times out waiting for a message reply
      - Did the commit occur in IMS?
      - Is there a forthcoming output reply that needs to be handled,
        - Application output or DFS error message
      - Possible actions
        - Create a process to respond to the reply - save/ print/ route, ...
        - Provide the ability to send an inquiry
    - IMS Exit Routines (DFSCMUX0, DFSNDMX0)
      - ...

Application Design - Summary

Remote program - IMS program interaction sequences

- Understand the existing process / requirement of navigating from one transaction to the next
  - Save output from one transaction as input to next
  - Buttons instead of PFKeys
  - Determine if the browser back button should be disabled
  - ...
- Determine the extent to which the IMS transaction is coded to 3270 screen behavior
  - Define what needs to be added to the remote program
- Understand the failure scenarios

RETURN TO INDEX