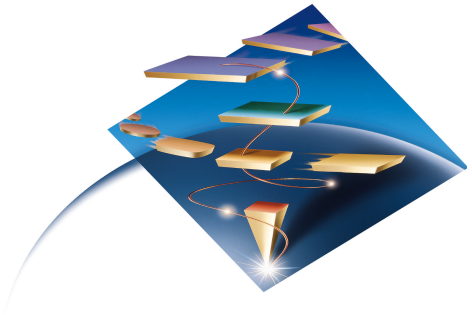
TCP/IP Applications in CS for OS/390

CS OS/390 Development, Raleigh
Alfred B Christensen - alfredch@us.ibm.com
SHARE Session 3612



TCP/IP Applications in CS for OS/390

e-business

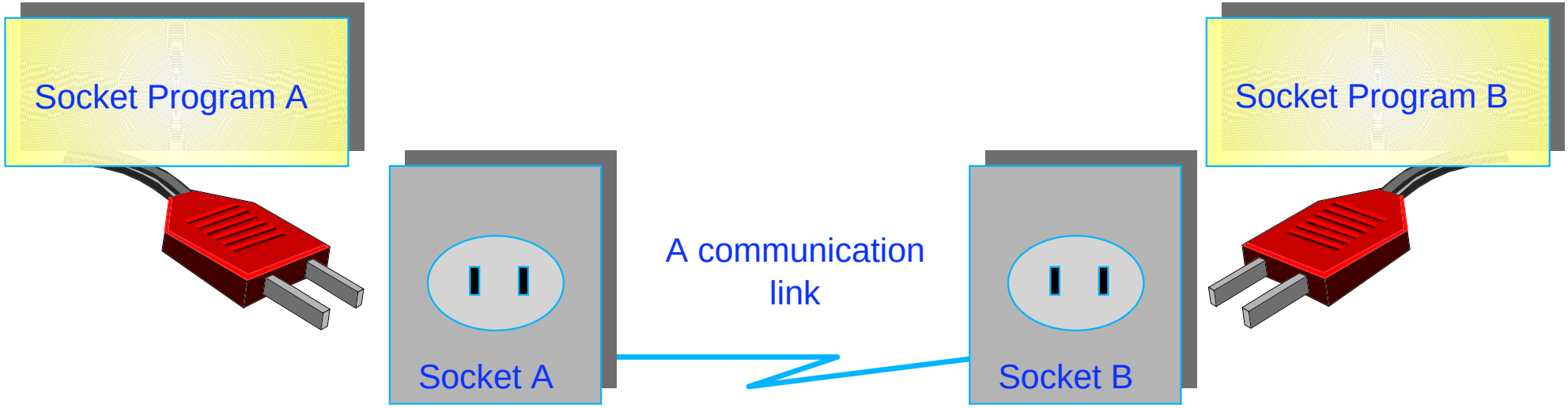
Session Number:	3912
Date:	Tuesday, February 23
Time:	11:00 AM
Location:	Parc 55, Third floor, Cervantes
Speaker:	Alfred B Christensen, IBM
Chair:	Eva Hocks, GSI
Abstract:	This session will focus on socket programming library structure on OS/390. It will present the concepts of native TCP/IP sockets vs. UNIX sockets on OS/390, and the relationship to resolvers and their configuration in an OS/390 environment where both native TCP/IP sockets and UNIX sockets are used. The session will further discuss the implications to socket programs when running multiple TCP/IP stacks in a Common INET environment.

Agenda

e-business

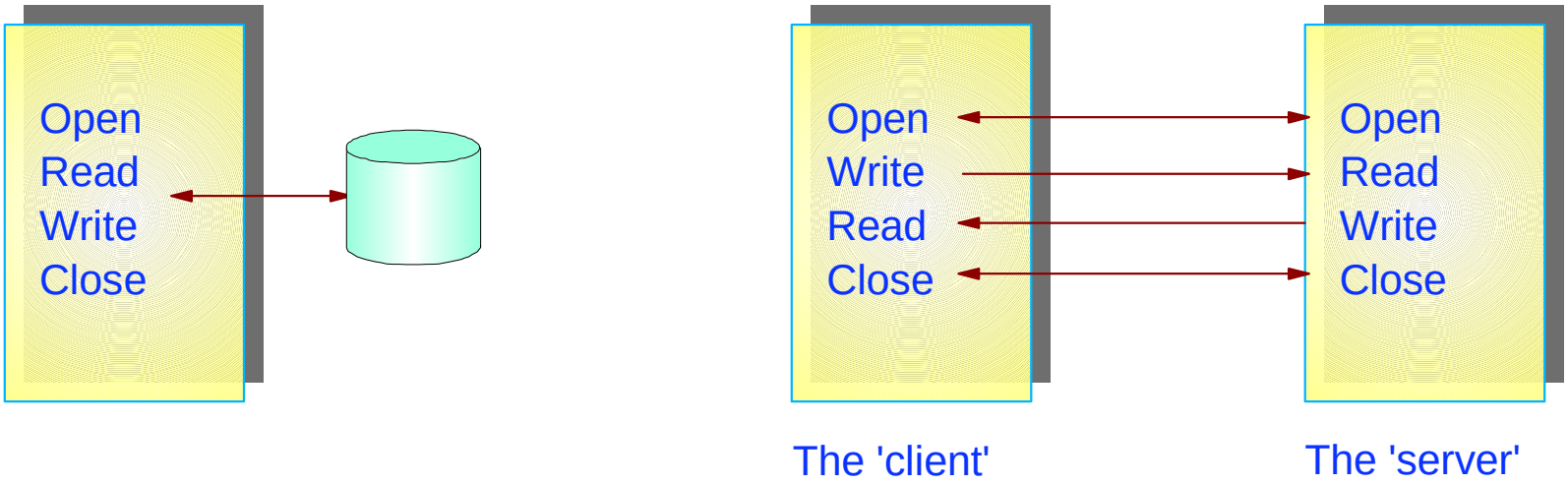
- Sockets Programs on OS/390
- Sockets Libraries on OS/390
- What is a resolver?
- Socket programs in a multi-stack configuration

The Socket Analogy



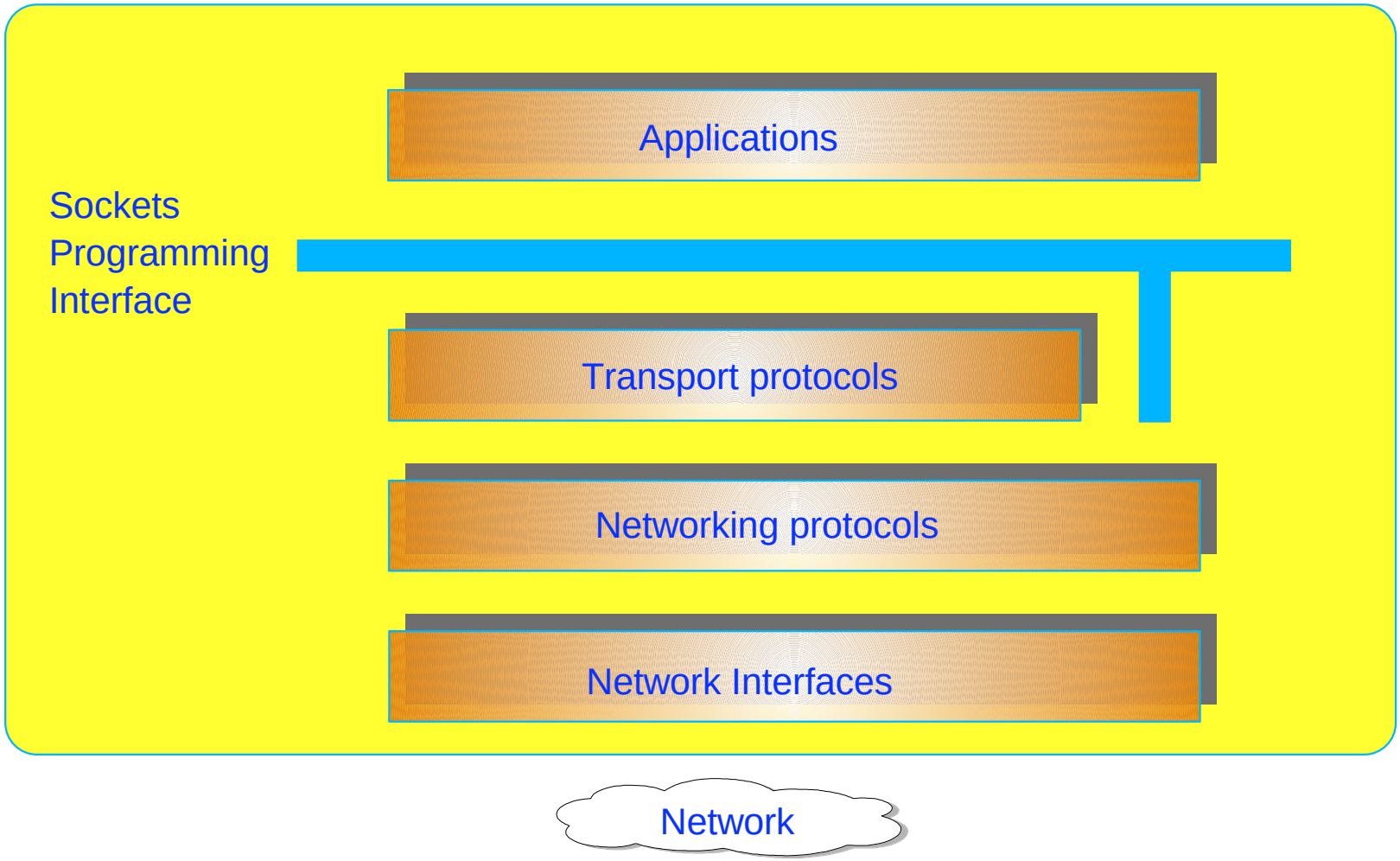
A *socket* uniquely identifies the *endpoint* of a *communication link* between two *application programs.*

Open/Close/Read/Write Paradigm



The basic sockets API functions are derived from the functions that a program uses to access files. The sockets API is based on the same paradigm.

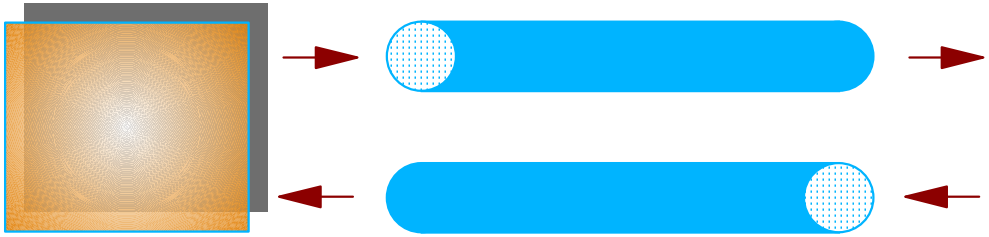
The Sockets API



Socket Types

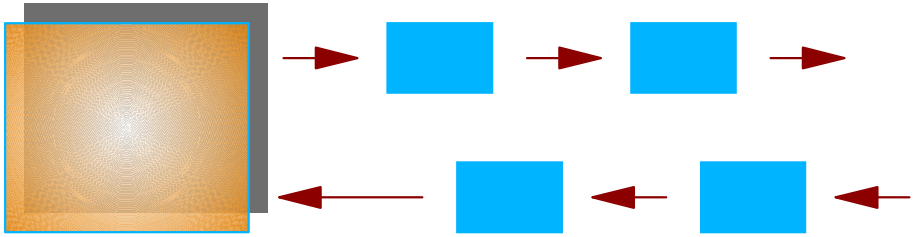
e-business

Stream socket



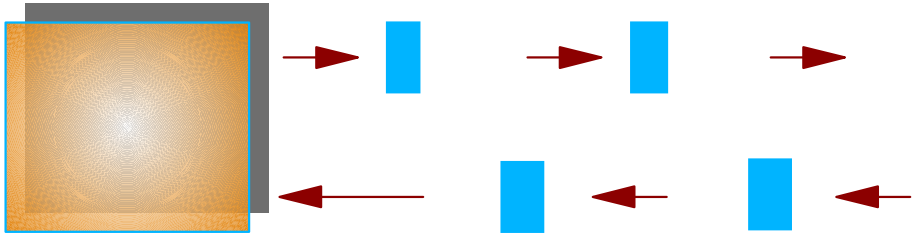
Two continuous streams in opposite directions

Datagram socket



Transport protocol datagrams independent of each other

Raw socket



Network layer datagrams independent of each other

Socket Type Characteristics

e-business

	STREAM	DATAGRAM	RAW
Reliable	Yes	Application responsibility	Application responsibility
Data size	Large amounts	Fixed datagram size	Fixed network packet size
TCP/IP protocol used	***TCP***	***UDP***	***IP***

Socket Address

A socket has an address. The address structure depends on the Addressing Family (AF) to which the socket belongs.

All socket address structures start with a binary halfword (2 bytes) that holds the addressing family identifier. The remaining part of the address structure is addressing family dependent.

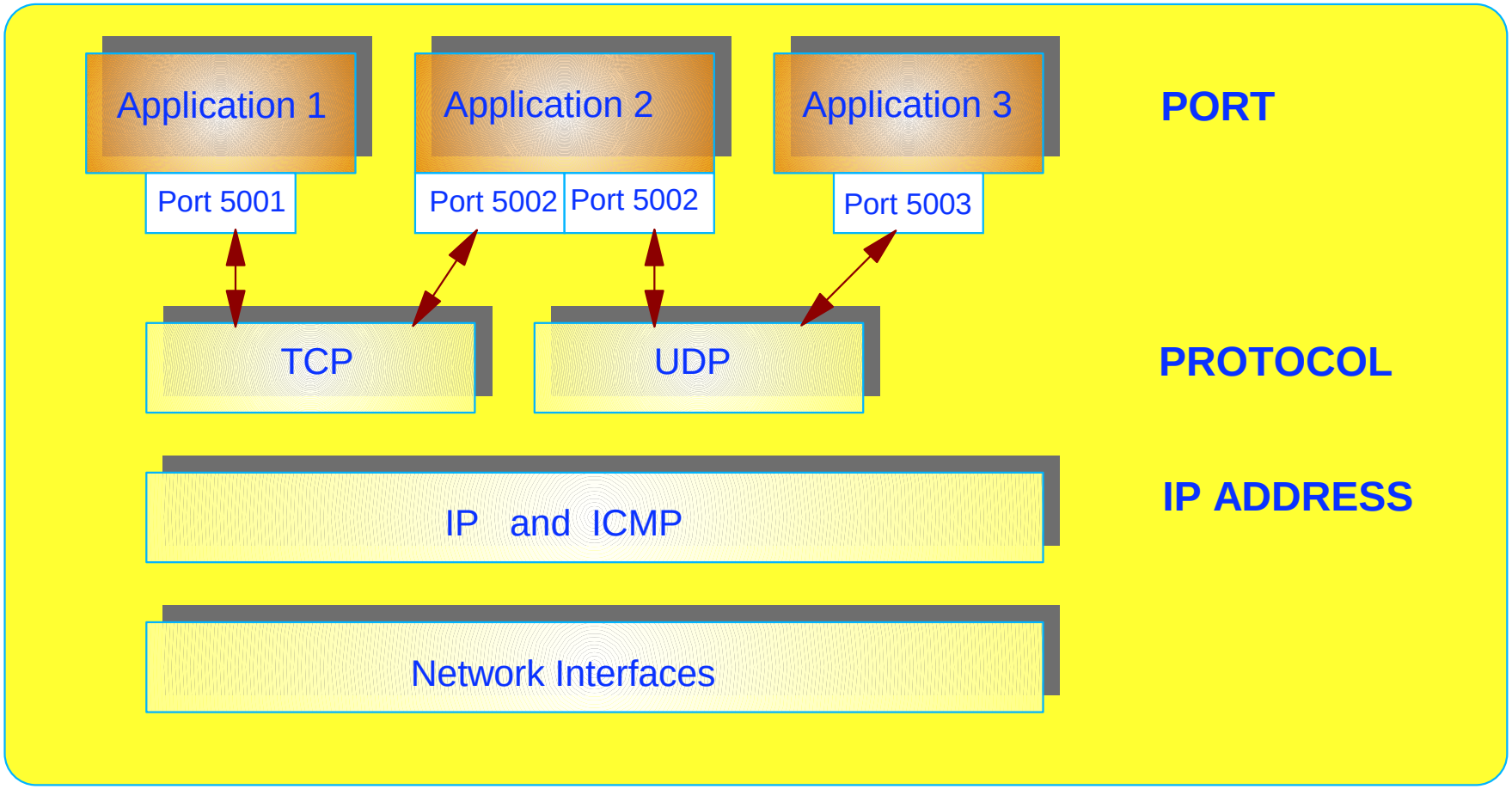
AF-INET

```
+0 Addressing Family=2
+2 Port number
+4 IP Address
+8 8 reserved bytes
   (zero)
```

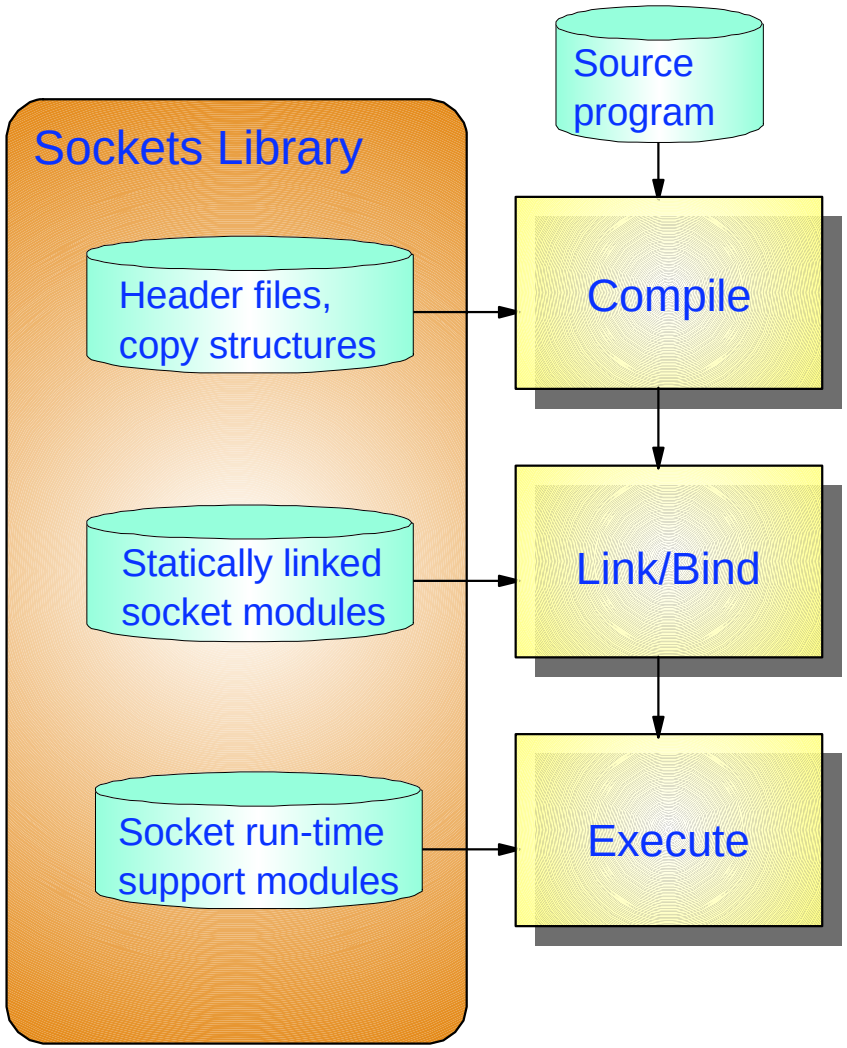
AF-UNIX

```
+0 Addressing Family=1
+2 108 character
   UNIX path name
```

AF-INET Address Elements



What is a Sockets Library?



A sockets library consists of:

- Compile-time structures (header files, copy structures, include files, macros, etc.)
- Statically linked modules (resolver modules, interface stub routines, etc.)
- Run-time modules (inter-address space communication routines)

The resolver code is part of the sockets library and is used for:

- gethostbyname()
- gethostbyaddr()
- getservbyname()
- gethostent()
- ....

Sockets Execution Environments and Stacks

e-business

There are two main sockets execution environments in OS/390:

1. **Native TCP/IP** - implemented by TCP/IP for MVS
   - A native TCP/IP socket program can use one and only one TCP/IP protocol stack at a time

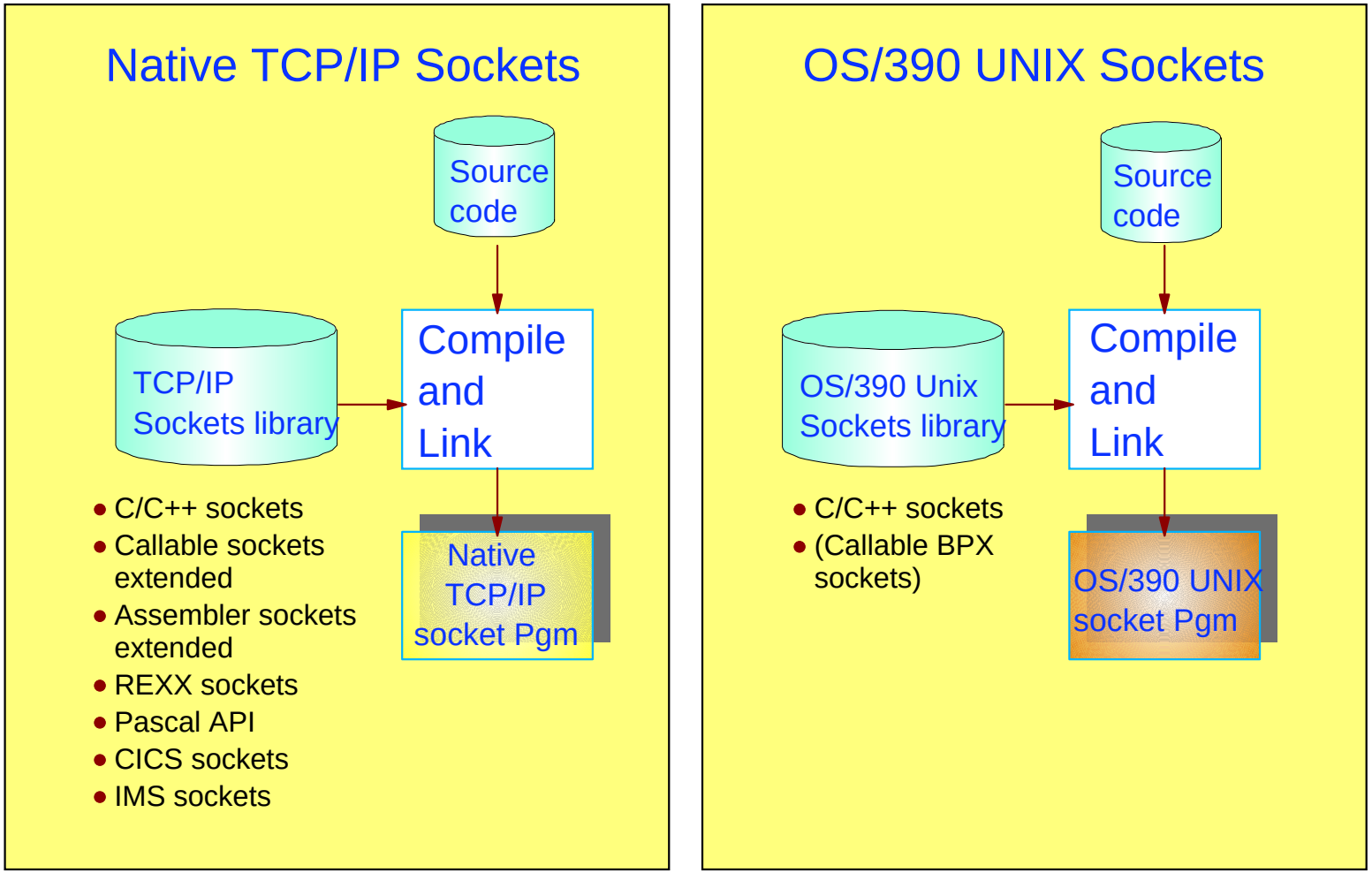
2. **UNIX** - implemented by OS/390 UNIX system services (Language Environment)
   - A UNIX socket program can use up to 8 TCP/IP protocol stacks at a time. The stacks may be a combination of any TCP/IP protocol stack that is supported by OS/390 UNIX system services - including AnyNet Sockets over SNA.

The native TCP/IP C socket library is not POSIX compliant and it should not be used for new C socket program development.

The non-C native TCP/IP socket libraries (sockets extended - call and assembler macro, REXX sockets, CICS sockets, and IMS sockets) are all current and available for development of new socket application programs.

How to Select OS/390 UNIX or Native TCP/IP Sockets

e-business



Sockets API Support Matrix

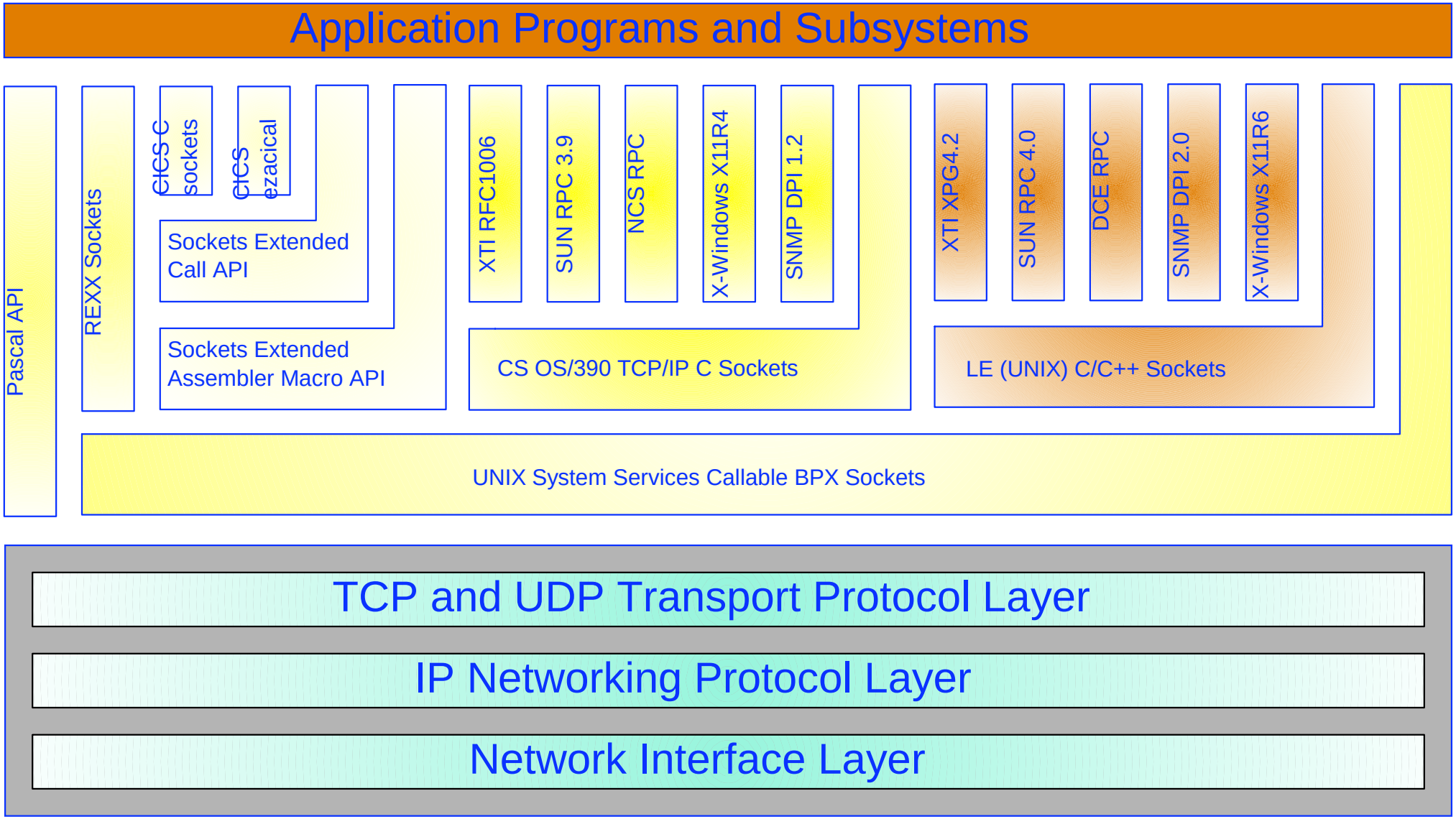
e-business

Sockets Library	TCP/IP V3R2	TCP/IP OE OS/390 V2R4	TCP/IP OS/390 V2R5+	AF_INET Support	AF_UNIX Support
OS/390 UNIX C/C++ Sockets	Yes	Yes	Yes	Yes	Yes
OS/390 UNIX Callable BPX Sockets (note 1)	Yes	Yes	Yes	Yes	Yes
TCP/IP MVS C Sockets	Yes		Yes	Yes	
TCP/IP MVS Sockets Extended ASM Macro	Yes		Yes	Yes	
TCP/IP MVS Sockets Extended Callable	Yes		Yes	Yes	
TCP/IP MVS REXX Sockets	Yes		Yes	Yes	
TCP/IP MVS ASM IUCV Socket API	Yes			Yes	
TCP/IP MVS Pascal Sockets	Yes		Yes	Yes	

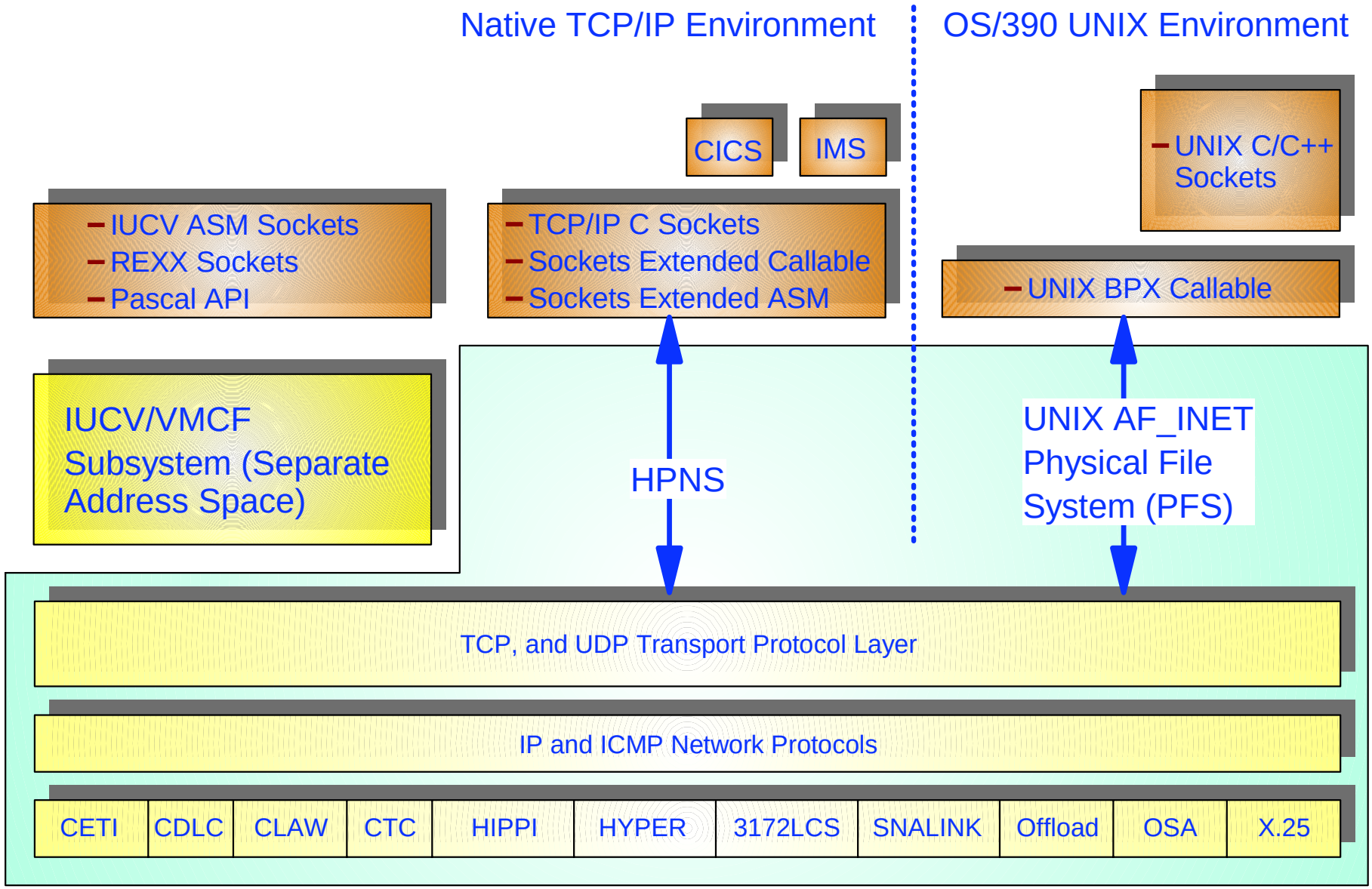
Note 1: BPX callable sockets is a low-level, high-performing assembler API without a resolver.

TCP/IP Networking API Relationship on OS/390

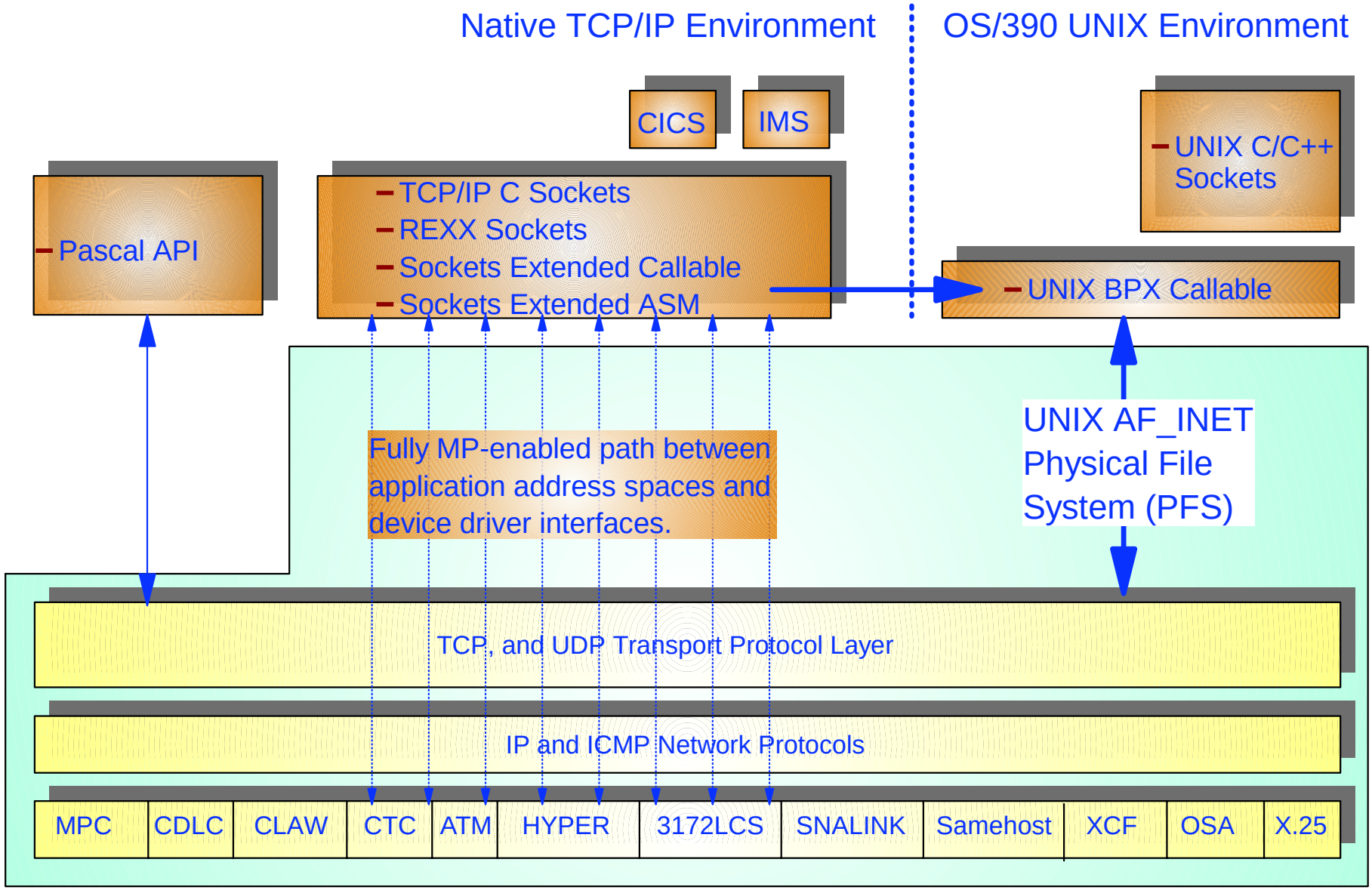
e-business



TCP/IP V3R2 API Structure



CS OS/390 V2R5+ TCP/IP API Structure

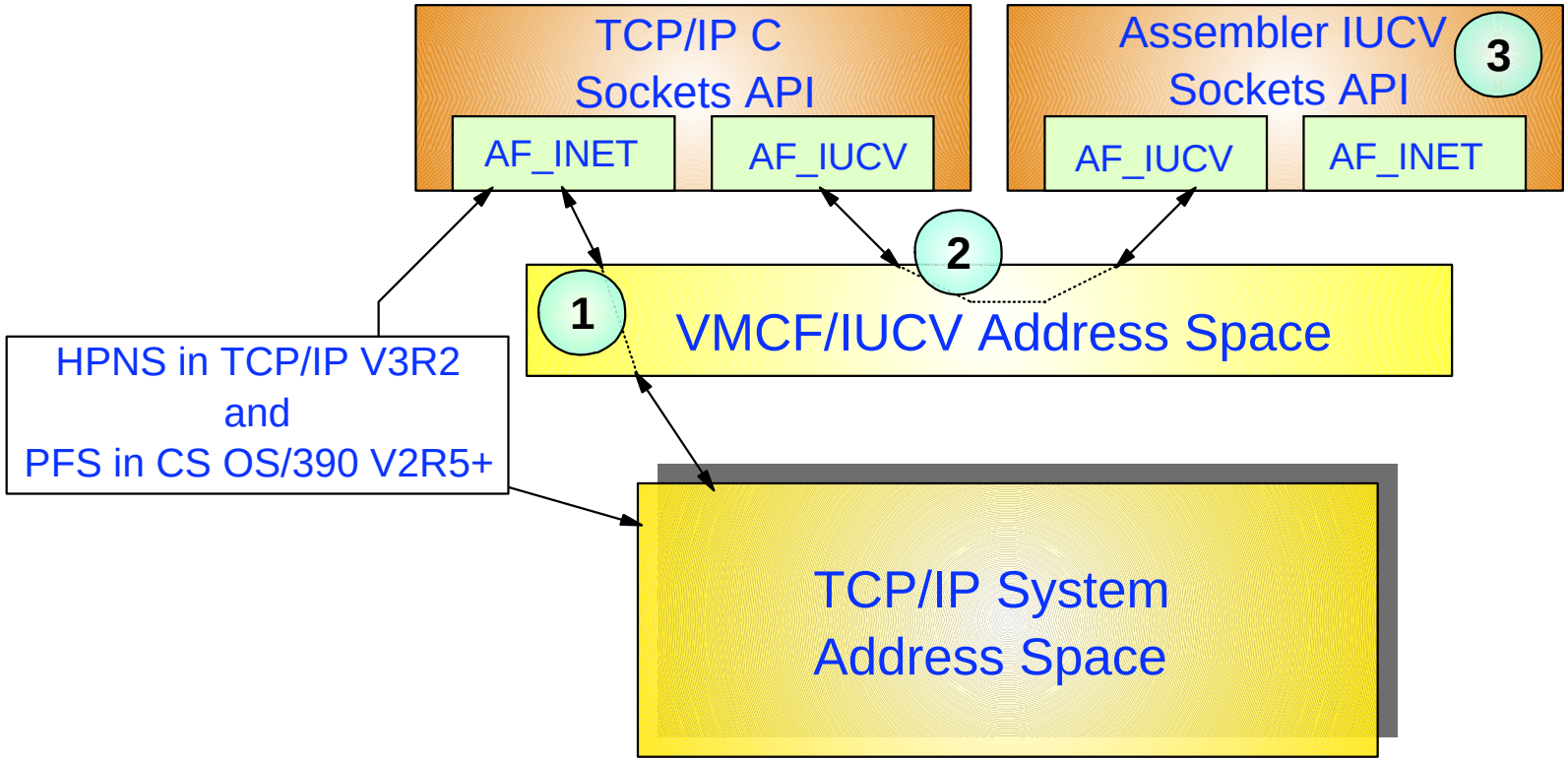


VMCF/IUCV Support

e-business

VMCF/IUCV Support consists of:

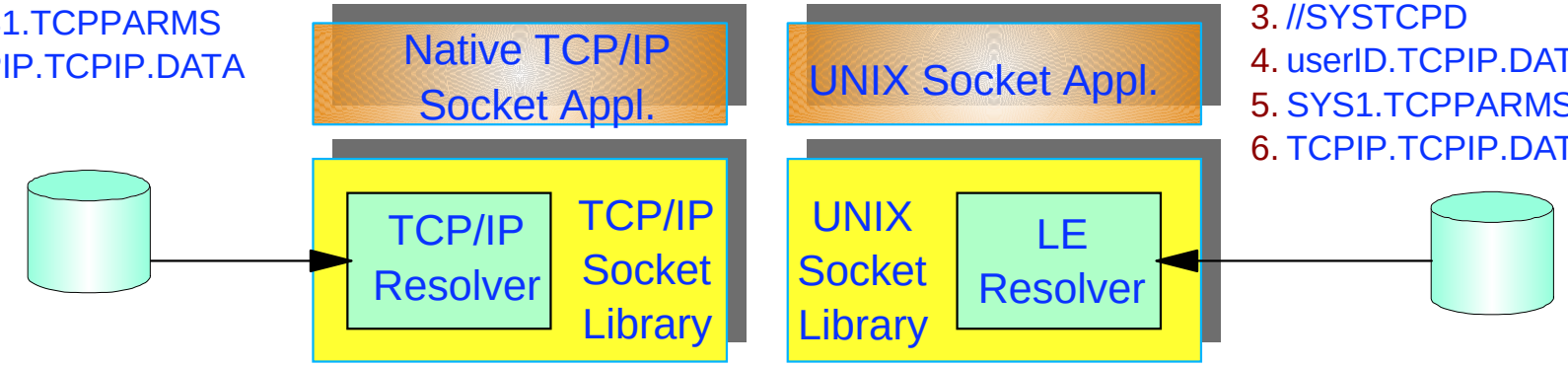
1. AF_INET services to communicate between application AS and TCP/IP AS (not used in CS OS/390 V2R5+)
2. AF_IUCV local socket communication support (still supported in CS OS/390 V2R5+, but for the TCP/IP supplied C-socket API only)
3. Assembler IUCV sockets API (not supported in CS OS/390 V2R5+)



Resolvers and Sockets Library

e-business

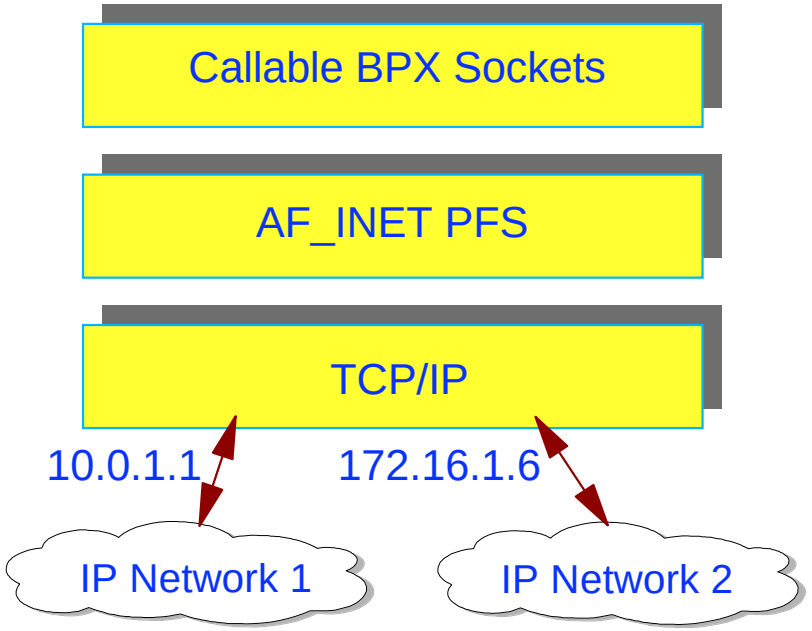
1. //SYSTCPD
2. userID/jobname.TCPIP.DATA
3. SYS1.TCPPARMS
4. TCPIP.TCPIP.DATA



1. RESOLVER_CONFIG
2. /etc/resolv.conf
3. //SYSTCPD
4. userID.TCPIP.DATA
5. SYS1.TCPPARMS
6. TCPIP.TCPIP.DATA

***No change** in setup for native TCP/IP socket programs from TCP/IP V3R1 or V3R2!*

TIP:
Member EZAGETIN in your hlq.SEZAINST library can be used to allocate and initialize the data sets that are needed by the TCP/IP resolver.



Avoid SYSTCPD DD statement for UNIX socket applications; use the RESOLVER_CONFIG environment variable instead as first search location.

Resolver Data Approaches

e-business

Two general approaches:

1. Use separate MVS data sets or files for the two resolvers - MVS data sets for the native TCP/IP resolver, and HFS files for the LE resolver.

TCP/IP Resolver:
[TCPIP.DATA]
hlq.ETC.PROTO
hlq.ETC.SERVICES
hlq.HOSTS.ADDRINFO
hlq.HOSTS.SITEINFO
hlq.STANDARD.TCPXLBIN

LE Resolver:
/etc/resolv.conf
/etc/protocol
/etc/service
/etc/hosts
hlq.STANDARD.TCPXLBIN

2. Share the same MVS data sets between both resolvers.

- Do *not* create the LE resolver files in the /etc directory
- Either use SYS1.TCPPARMS(TCPDATA) [recommended], or use the SYSTCPD DD statement for the TCP/IP resolver and point the LE resolver to your MVS data set TCPIP.DATA via the RESOLVER_CONFIG environment variable
- Let both resolvers find the remaining resolver configuration data sets via the DATASETPREFIX value in your TCPIP.DATA

How to Set the RESOLVER_CONFIG Environment Variable

e-business

- In the UNIX shell:
  - `export RESOLVER_CONFIG="//'USER1.ALFRED.CNTL(TCPDATAT)'"`
- In a BPXBATCH STDENV input file or data set:
  - `RESOLVER_CONFIG=//'USER1.ALFRED.CNTL(TCPDATAT)'`
- In a POSIX(ON) program's EXEC PARM string:

Col. 72 !!!!!

```
//FTPD  EXEC PGM=FTPD,REGION=0K,TIME=NOLIMIT,
// PARM='POSIX(ON) ALL31(ON) ENVAR("RESOLVER_CONFIG=//''USER1.ALFRED.CNX
//              TL(TCPDATAT)'"")/ PORT 621 TRACE'
```

Col. 16 !!!!!

- In a POSIX(ON) program's environment variable input file or data set:

```
//FTPD  EXEC PGM=FTPD,REGION=0K,TIME=NOLIMIT,
// PARM='POSIX(ON) ALL31(ON) ENVAR("_CEE_ENVFILE=DD:MYVARS")/ PORT 621 X
//              TRACE'
//MYVARS    DD *
RESOLVER_CONFIG=//'USER1.ALFRED.CNTL(TCPDATAT)'
/*
```

C, REXX, Pascal, and Sockets Extended API Migration to CS OS/390 V2R5+

e-business

Socket API	From TCP/IP V3R1	From TCP/IP V3R2
TCP/IP C-sockets	Relink	No change
Sockets Extended Call API	No change*	No change
Sockets Extended Assembler macro API	No change*	No change
Compiled REXX sockets	Recompile and relink	Recompile and relink
Interpreted REXX sockets	No change	No change
Pascal API	Relink	Relink

- Programs that use raw sockets require relink with AC(1) to an APF authorized load library
- For C programs, be aware that only the LE runtime libraries are supported
- Be aware that once a C program has been compiled and linked with C and LE on OS/390 V2R5, it *may* not run on previous LE releases
- Some errno values on failing socket calls have changed (ENOTCONN is now EPIPE)
- OMVS segment is required for all non-Pascal socket programs (Default OMVS segment is sufficient)
- New socket options:
  - ioctl() - SIOCGMONDATA
  - setibmsockopt() and getibmsockopt() - SO_IGNORESOURCEVIPA and SO_OPTMSS
- Some Pascal API functions have been removed or changed (see the Planning and Migration Guide for details)
- **No change*** - Relink for optimum performance

BPXPRMxx for Support of Multiple CS OS/390 V2R5+ TCP/IP Stacks

e-business

- Establishing a Common INET environment
  - Multiple stacks - Common INET file system type (CINET)
  - Sample Definition in **BPXPRMxx** parmlib member

Do NOT create a multi-stack configuration unless you REALLY need it!!

Indicates Common INET

Default Stack

```
FILESYSTYPE TYPE(CINET)  ENTRYPOINT(BPXTCINT)
SUBFILESYSTYPE NAME(TCPV34) TYPE(CINET)  ENTRYPOINT(EZBPFINI)
                            DEFAULT
SUBFILESYSTYPE NAME(TCPV34A) TYPE(CINET)  ENTRYPOINT(EZBPFINI)
SUBFILESYSTYPE NAME(TCPV34B) TYPE(CINET)  ENTRYPOINT(EZBPFINI)
NETWORK DOMAINNAME(AF_INET)
            DOMAINNUMBER(2)
            MAXSOCKETS(60000)
            TYPE(CINET)
            INADDRANYPORT(10000)
            INADDRANYCOUNT(1000)
```

Reserved Ports for UNIX System Services (assigned to applications performing binds to non-specified ports) - Also need to be reserved in each TCP/IP profile

SUBFILESYSTYPE NAME keyword refers to Jobname/address-space name of TCP/IP started task; not the started task user ID of the TCP/IP started task.

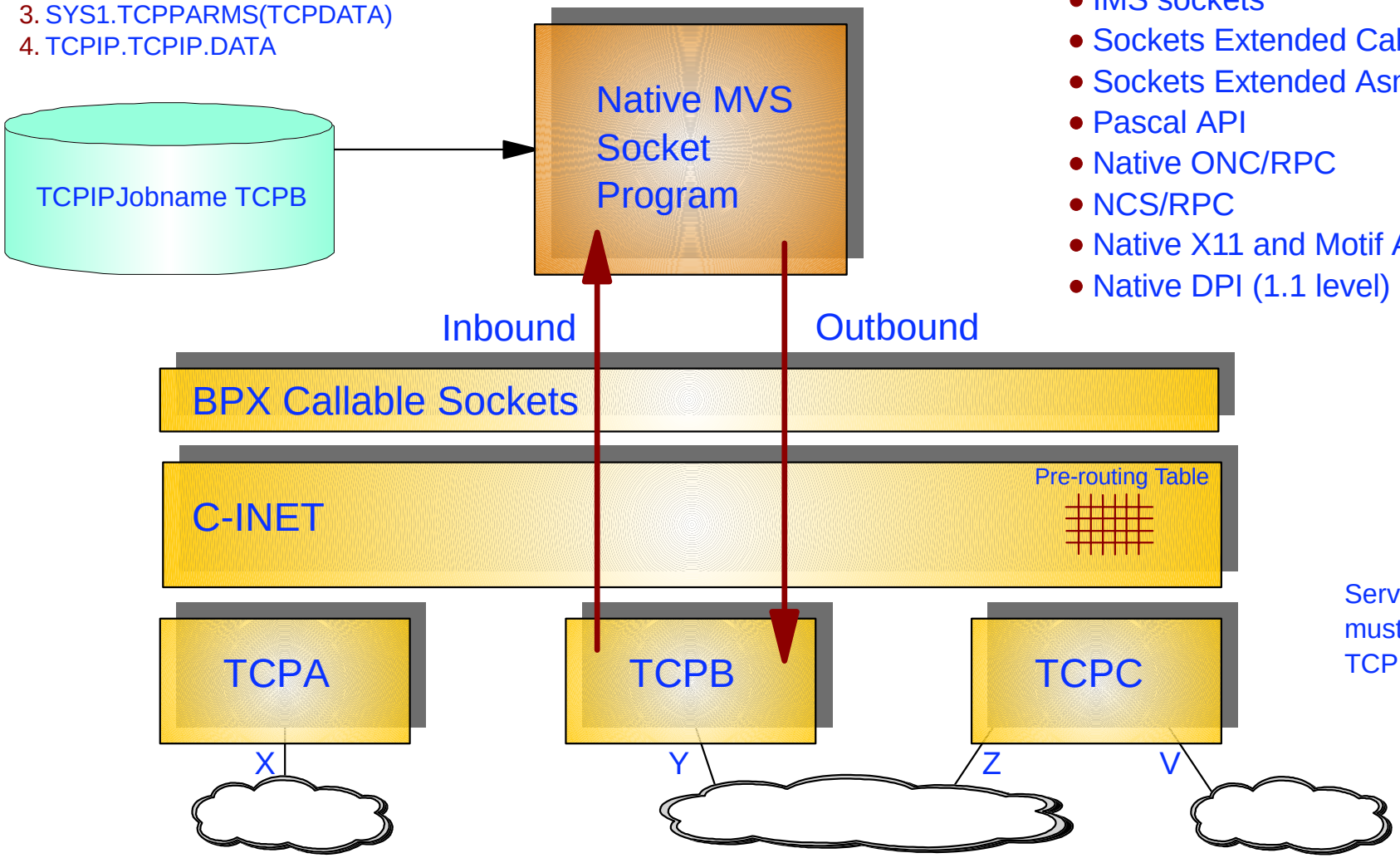
Native TCP/IP Socket Program

e-business

Standard search order for TCPIP.DATA:

1. //SYSTCPD DD statement
2. userID/Jobname.TCPIP.DATA
3. SYS1.TCPPARMS(TCPDATA)
4. TCPIP.TCPIP.DATA

- TCP/IP C-sockets
- CICS sockets
- IMS sockets
- Sockets Extended Call API
- Sockets Extended Asm. macro API
- Pascal API
- Native ONC/RPC
- NCS/RPC
- Native X11 and Motif APIs
- Native DPI (1.1 level)

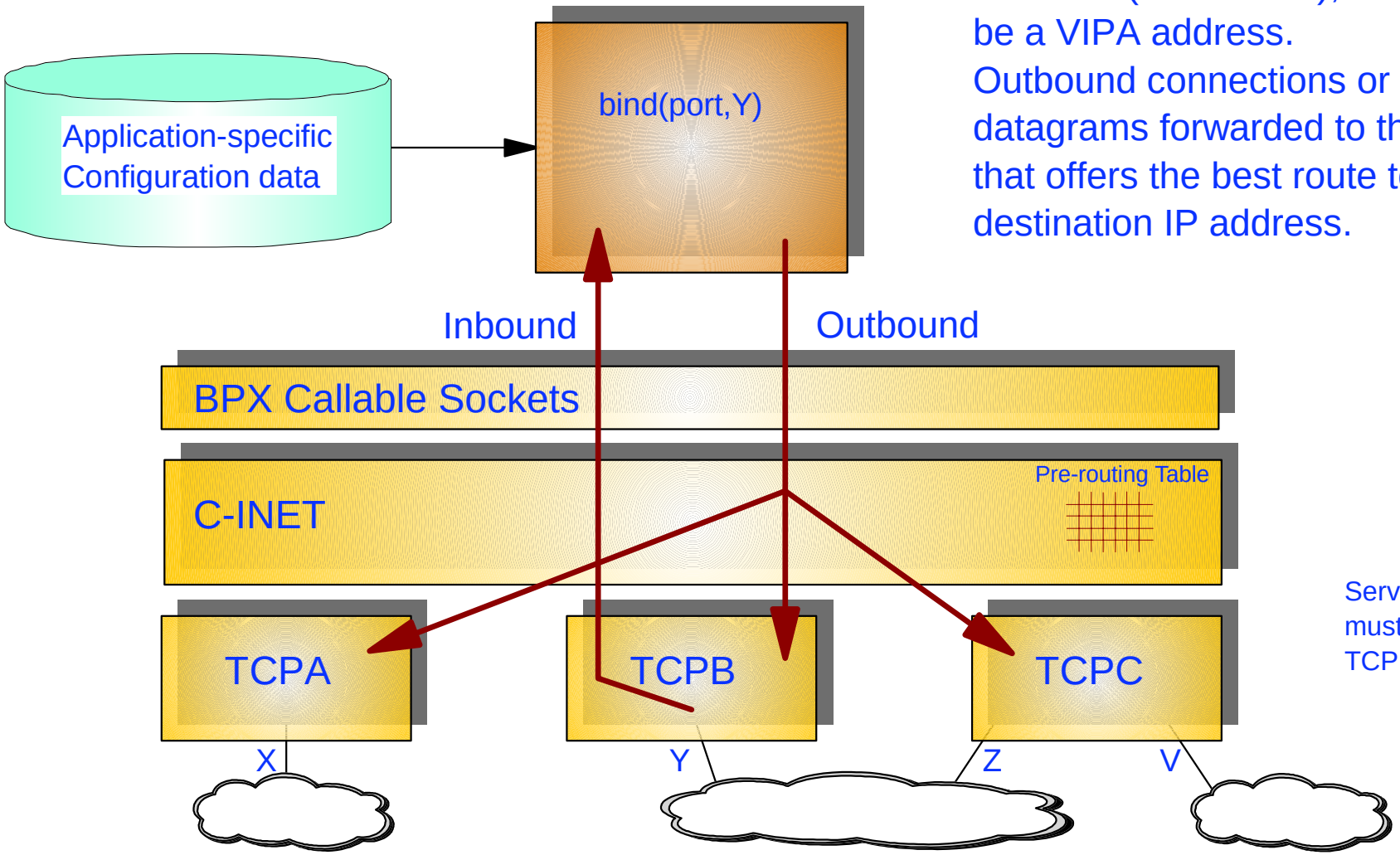


Server port number must be reserved in TCPB.

Bind-Specific UNIX Socket Program

e-business

Will accept incoming connections or UDP datagrams over specified interface (IP address), which could be a VIPA address.
Outbound connections or UDP datagrams forwarded to the stack that offers the best route to the destination IP address.

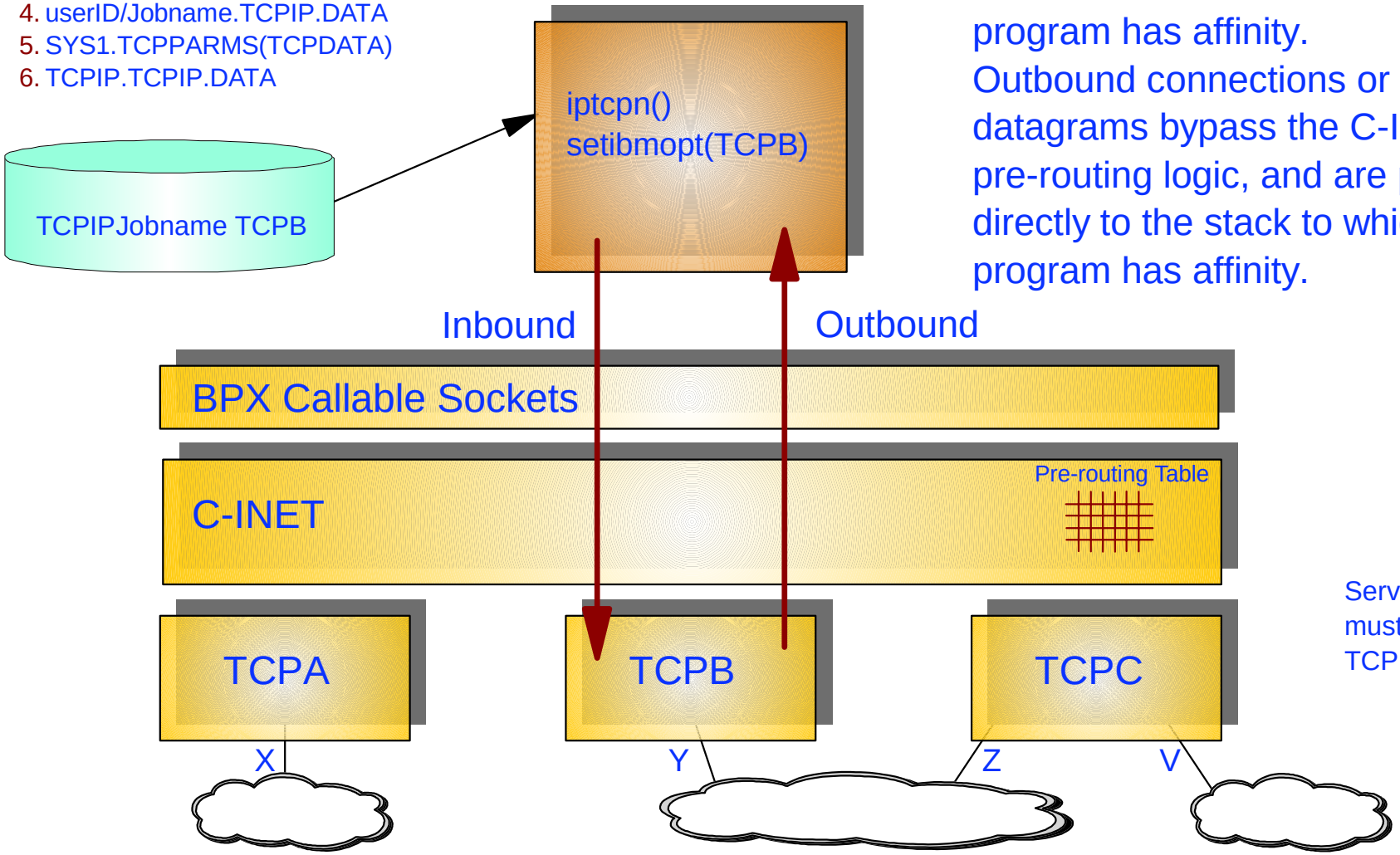


Server port number must be reserved in TCPB.

Stack-Affinity UNIX Socket Program

Standard search order for resolver configuration file:

1. RESOLVER_CONFIG environment variable
2. /etc/resolv.conf
3. //SYSTCPD DD statement
4. userID/Jobname.TCPIP.DATA
5. SYS1.TCPPARMS(TCPDATA)
6. TCPIP.TCPIP.DATA



Will accept incoming connections or UDP datagrams over any interface of the stack to which the program has affinity.
Outbound connections or UDP datagrams bypass the C-INET pre-routing logic, and are passed directly to the stack to which the program has affinity.

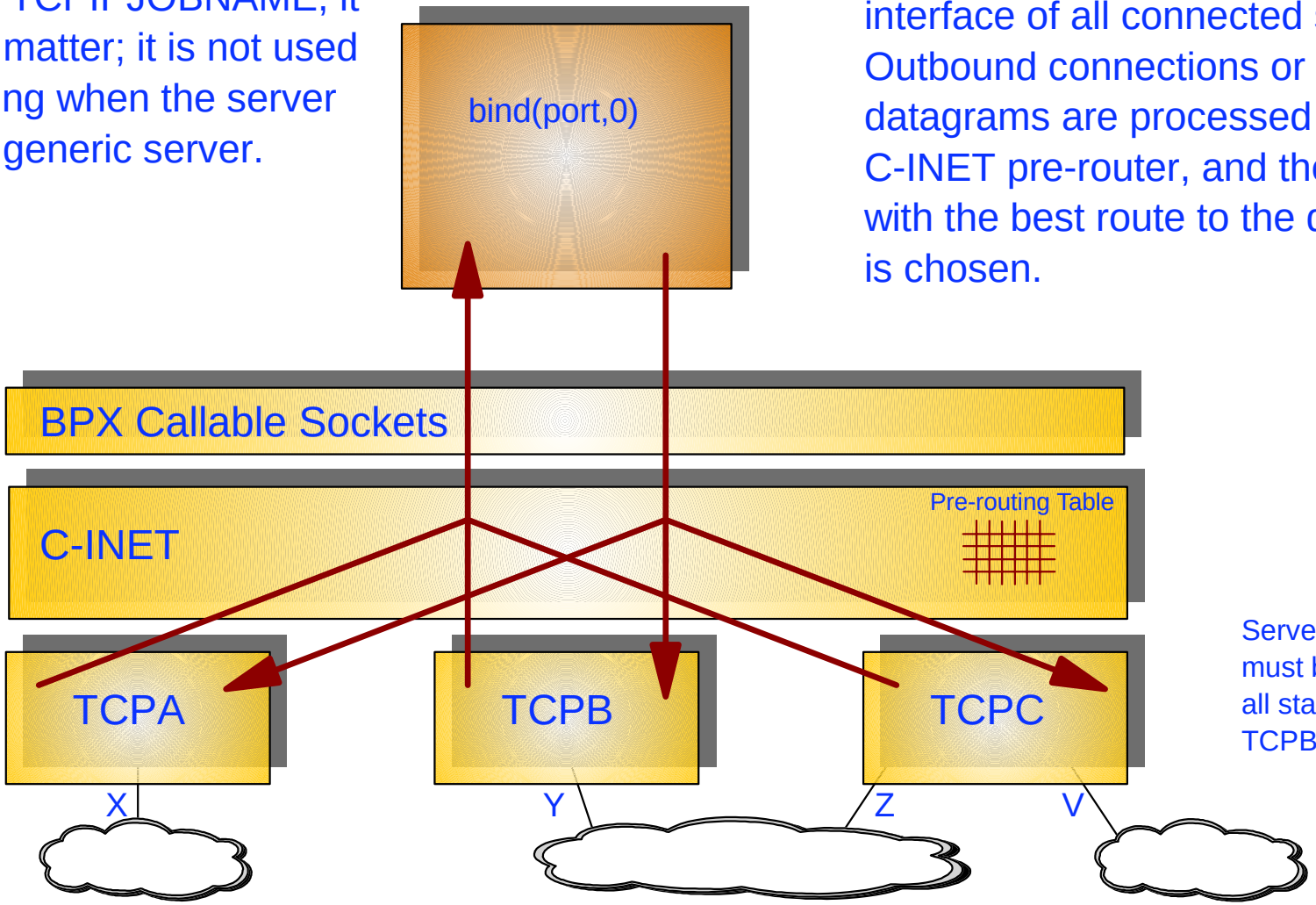
Server port number must be reserved in TCPB.

Generic UNIX Socket Program

e-business

If the chosen resolver configuration file has a TCPIPJOBNAME, it does not matter; it is not used for anything when the server is a pure generic server.

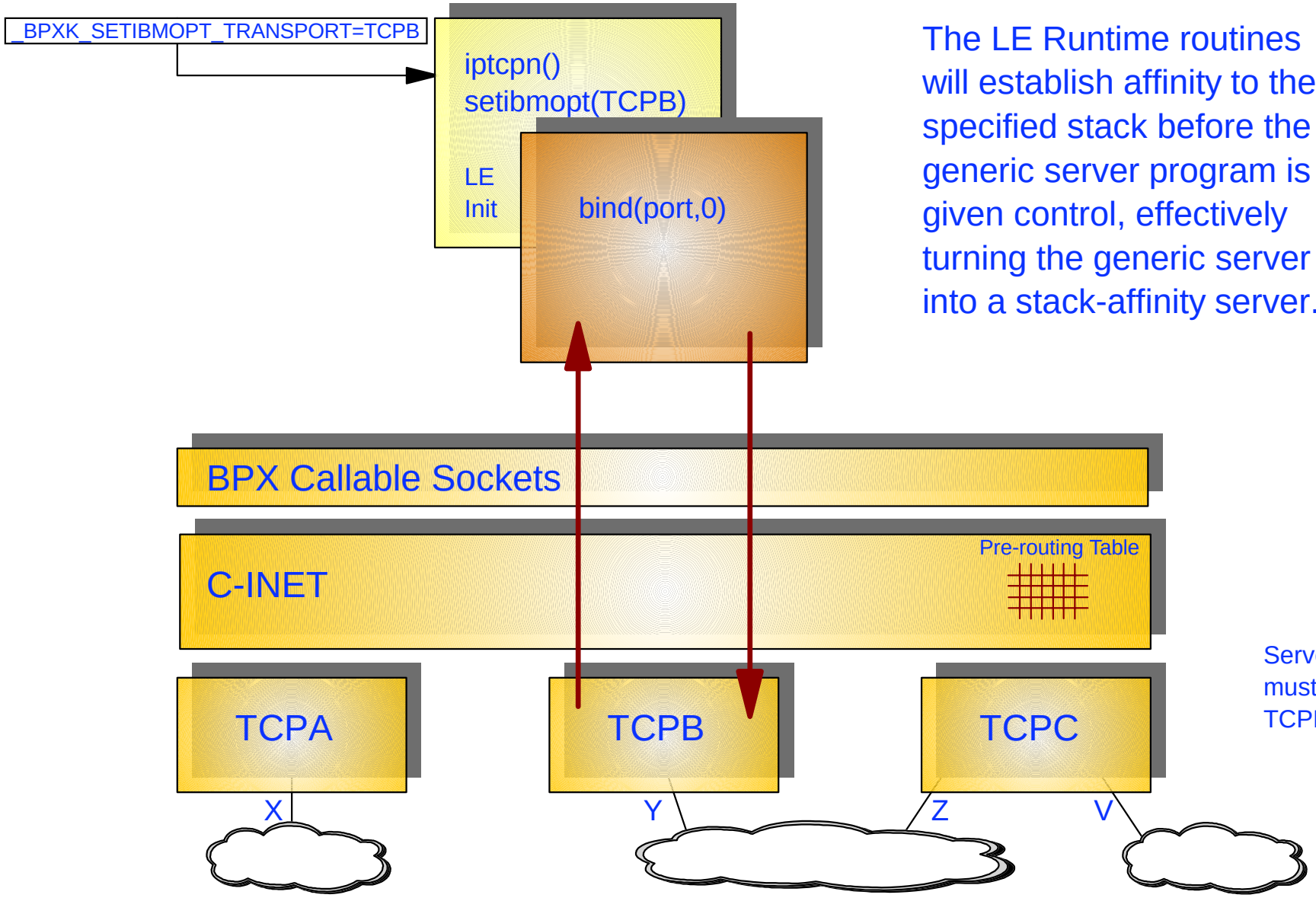
Will accept incoming connections or UDP datagrams over any interface of all connected stacks. Outbound connections or UDP datagrams are processed by the C-INET pre-router, and the stack with the best route to the destination is chosen.



Server port number must be reserved in all stacks: TCPA, TCPB, and TCPC.

Turning a Generic UNIX Socket Program into a Stack-Affinity Program

e-business



The LE Runtime routines will establish affinity to the specified stack before the generic server program is given control, effectively turning the generic server into a stack-affinity server.

Server port number must be reserved in TCPB.

Controlling Applications in a Multi-Stack Environment

e-business

To control which stack(s) a socket program uses in a multi-stack environment, the following information is needed:

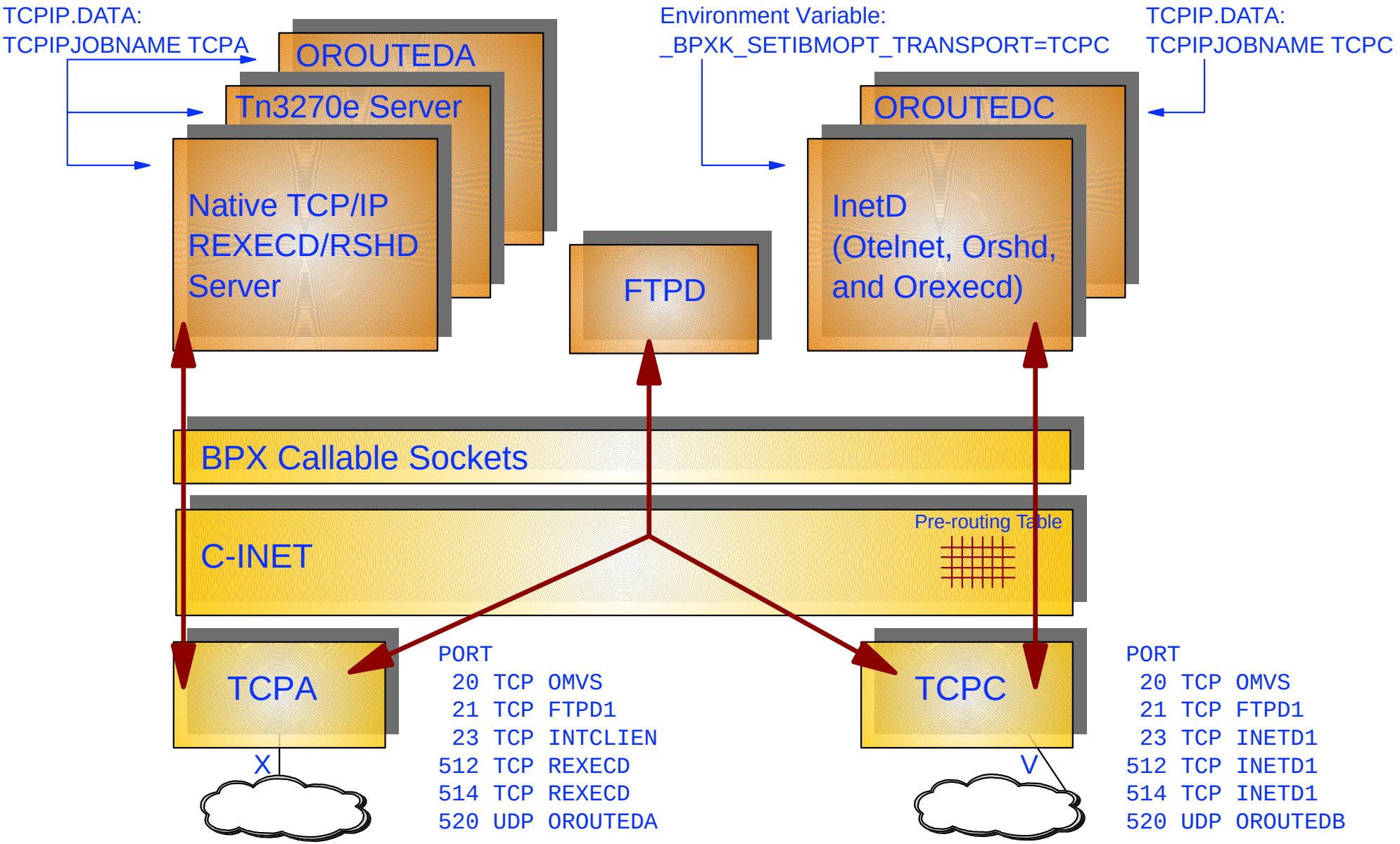
1. Is the program a native TCP/IP socket program?
   - ➡ Use standard TCPIP.DATA with TCPIPJOBNAME.

2. Is the program a bind-specific UNIX socket program?
   - ➡ Use application's configuration options to specify which IP address to accept incoming traffic over. Optionally use _BPXK_SETIBMOPT_TRANSPORT to establish stack-affinity for outbound traffic (can be used in JCL only!).

3. Is the program a stack-affinity UNIX socket program?
   - ➡ Use resolver configuration file with TCPIPJOBNAME to specify which stack to establish affinity to.

4. Is the program a generic UNIX socket program?
   - ➡ Use _BPXK_SETIBMOPT_TRANSPORT to establish stack-affinity (if this is desired; you may want some generic servers to execute as generic servers).

Native TCP/IP and UNIX Servers on Same Port Numbers



Summary

e-business

- Sockets APIs are implemented by sockets libraries
- There are two sockets execution environments on OS/390:
  1. Native TCP/IP - supported by the sockets libraries that are supplied by TCP/IP
  2. UNIX System Services - supported by the UNIX sockets libraries that are supplied by UNIX System Services (Language Environment)
- You choose sockets execution environment when you compile and link your socket program
- Native TCP/IP socket programs use the TCP/IP stack that is specified in the TCPIP.DATA resolver configuration data set on the TCPIPJOBNAME keyword.
- UNIX socket programs in a multi-stack configuration use one or more TCP/IP stacks, depending on how the socket program has been developed and/or the configuration options you specify when the program is executed:
  1. Bind-specific - single IP address (and TCP/IP stack) for incoming requests and all available IP addresses and stacks for outbound requests
  2. Generic - all available IP addresses and TCP/IP stacks for both incoming and outbound requests
  3. Stack-affinity - all IP addresses in a single TCP/IP stack for both incoming and outbound requests

For More Information....

e-business

URL	Content
http://www.software.ibm.com/network	eNetwork Software
http://www.software.ibm.com/network/commserver	eNetwork Communications Server
http://www.software.ibm.com/network/commserver/support	eNetwork Software Support
http://www.software.ibm.com/network/on-demand	eNetwork On-Demand Server
http://www.software.ibm.com/network/hostondemand	Host On-Demand
http://www.software.ibm.com/network/pcomm	Personal Communications
http://www.software.ibm.com/network/technology	Network Technologies
http://www.networking.ibm.com/netprod	Networking Hardware