HOSTBRIDGE®
TECHNOLOGY

IBM Business Partner

Ready for IBM SOA Specialty

Ready for IBM Rational software

(SHARE Anaheim)

CICS Integration & Optimization: Tales from the Trenches

Russ Teubner - russ@hostbridge.com

Original Abstract (it morphed)

CICS users are loyal to their apps – and for good reason! However, they also need to <u>integrate</u> these same applications with an ever widening array of <u>web and cloud-based resources</u>. And, if that weren't enough, every year they are under pressure to add new value, and <u>reduce the cost of ownership</u>. That's a tall order. This session will highlight ~~a few~~ of the tactics and strategies that customers can use to enhance the value of the existing CICS apps while lowering the cost of ownership.

One

Customer Context (why it morphed)

- When the abstract was submitted, we had just been approached by a large CICS customer with some intriguing business/technical objectives
- Business Objectives
  - Respond to competitive pressures in their industry
  - Lower incremental cost of high-volume CICS application processing (i.e., marginal value > marginal cost)
  - Move new/additional workload to System z and reinforce CICS TS as the most cost effective platform for their business

- Technical Objective (at least their hope)
  - Reduce the CPU burn (GP) associated with socket applications and infrastructure

Perfect R&D Situation

- **Well defined business objectives**
- **An initial theory as to what the technical issues might be**
- **Very strong in-house CICS talent**
- **Load testing infrastructure in place**
- **Good CICS tools on hand**
- **Test LPAR/region available**
- **Had a spare cubicle**

Timing Was Opportune

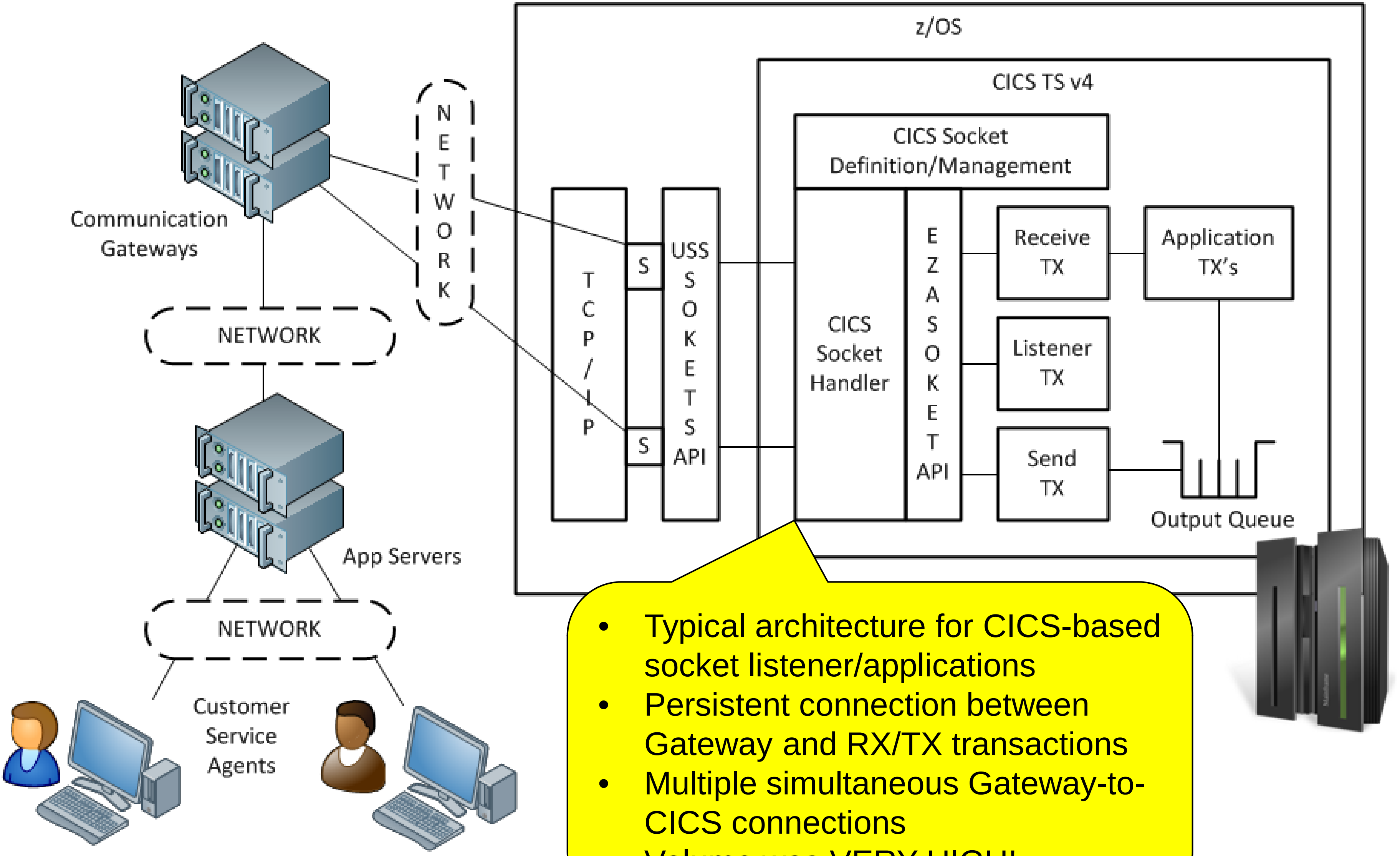
- **Customers were continuing to state their concern about doing more for less**
- **We had just delivered zIIP-enabled versions of our products, and our heads were filled with fun facts related to:**
  - **z/OS, USS, LE, WLM, SRBs, zIIP**
  - **CICS TS v4 Open Transaction Environment**
  - **Sockets**

- **Other factors:**
  - **We are zealots regarding integration of CICS apps/data as part of web/cloud-based infrastructure**
  - **We are committed to delivering functionality <u>under</u> CICS**
  - **I didn't want to stop writing code (zIIP project was too fun)**

Cut to the Chase

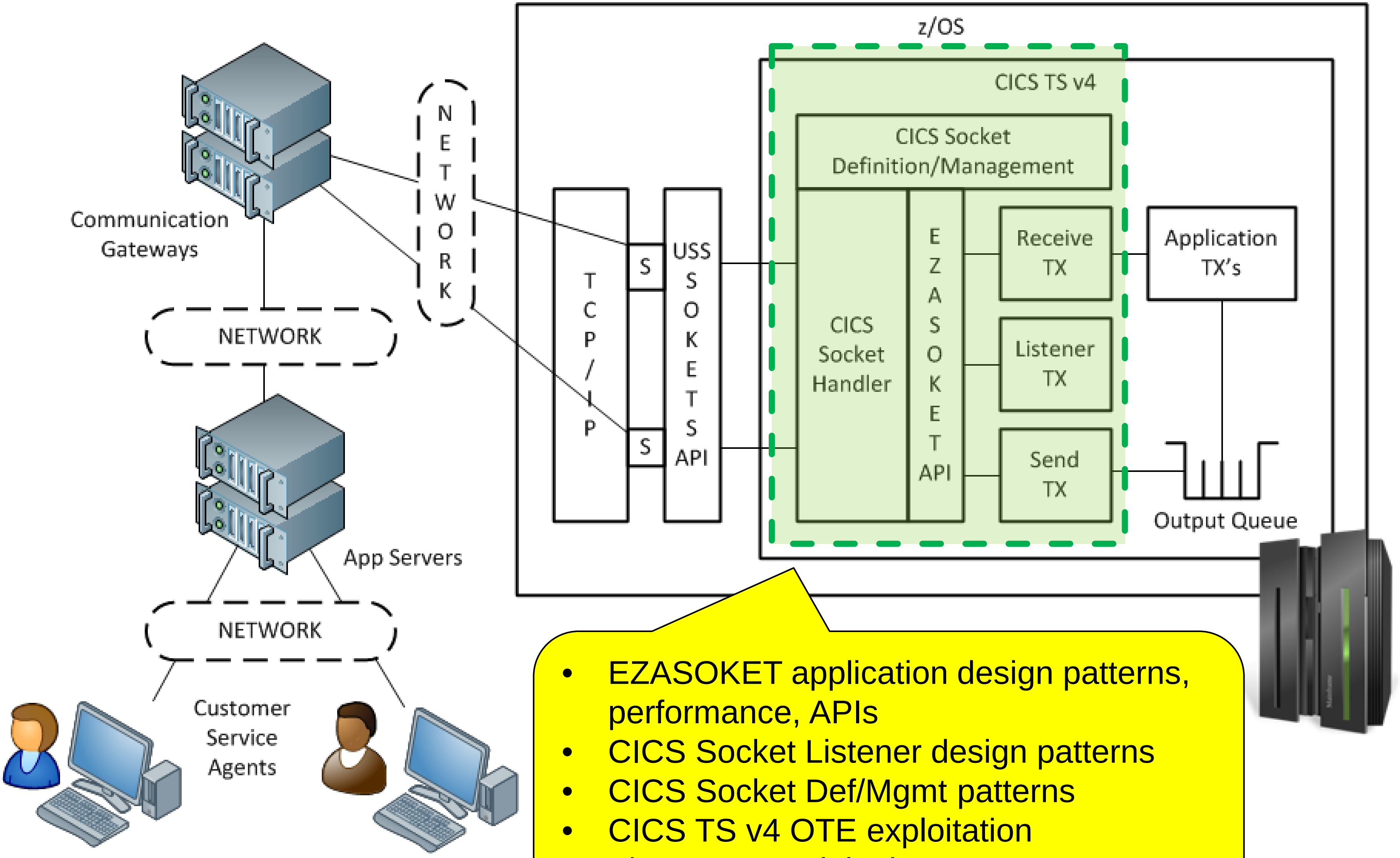
- What we learned was surprising and the results were unexpected (in a good way)
- We ended up exploiting CICS TS v4 OTE and z/OS to create a solution
- I want this to be knowledge you can use:
  - The approach is generally applicable to any CICS customer who has socket apps
  - The higher your volume, the more it matters
- Yes… I’m “a vendor” but please forget that for now – I’m speaking as a CICS developer

Initial Conditions



- Typical architecture for CICS-based socket listener/applications
- Persistent connection between Gateway and RX/TX transactions
- Multiple simultaneous Gateway-to-CICS connections
- Volume was VERY HIGH!

Research Focus



- EZASOKET application design patterns, performance, APIs
- CICS Socket Listener design patterns
- CICS Socket Def/Mgmt patterns
- CICS TS v4 OTE exploitation
- z/OS USS exploitation

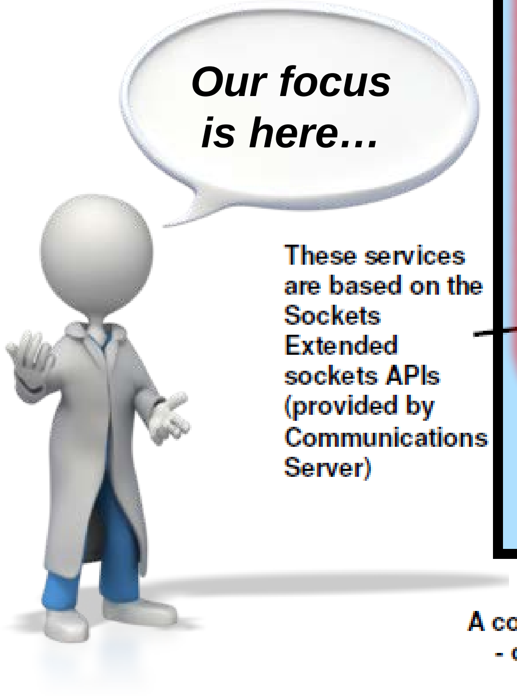
CICS Socket Support

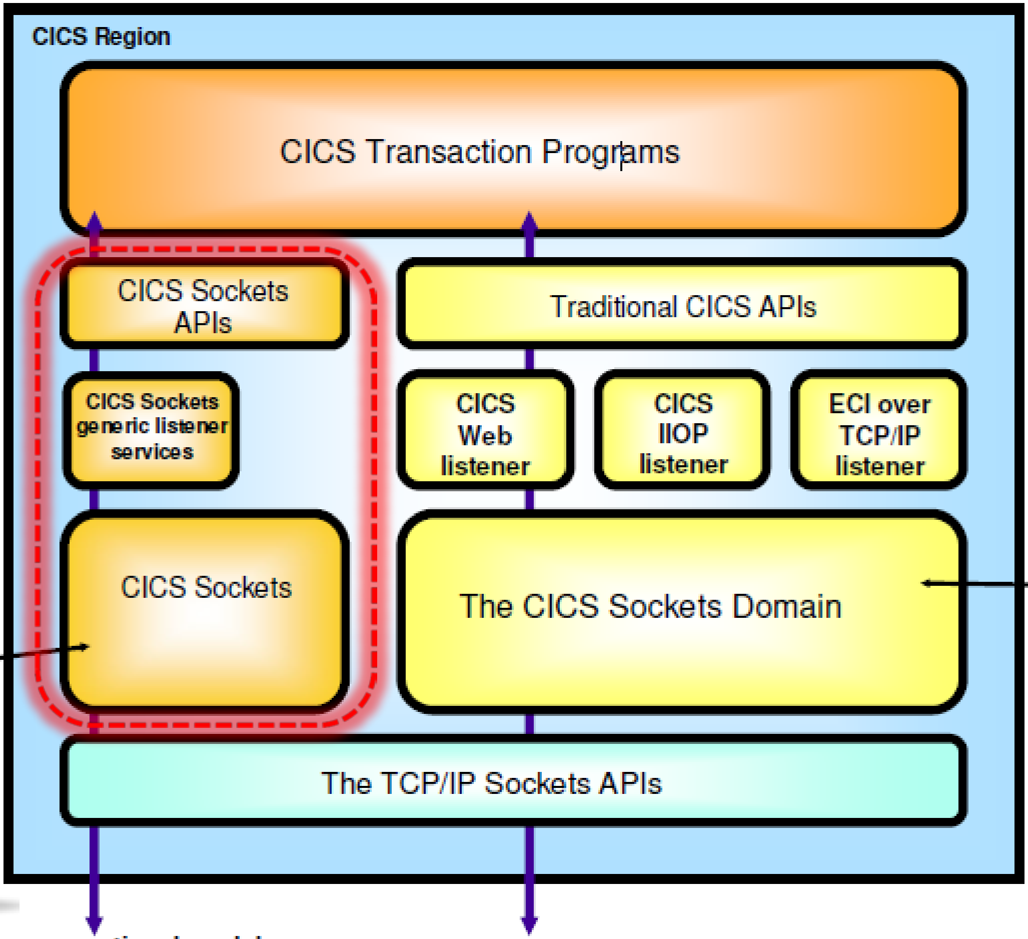
- Provided as part of z/OS Communications Server
- What it includes:
  - Socket APIs
    - C language API
    - Sockets Extended API (aka, EZASOKET or EZACICSO)
    - Original COBOL API (aka, EZACICAL)
  - Listeners: standard and enhanced (i.e., CSKL); or user-written
  - Definition and management components (e.g., EZAO)
- A <u>well-documented workhorse</u>, but…
- It's been around a long time (circa 1992)
- Older than CICS OTE
  - Thus… much of it's original architecture
- Reengineered to support OTE
  - But… the general approach of the original architecture persisted

Thus, I'm NOT referring to CICS TS features which use the CICS Sockets Domain.

CICS Sockets ≠ Sockets Domain

A CICS Sockets transaction has direct access to the TCP/IP socket and can issue native sockets calls to receive and send data over the socket.





These services are based on the Sockets Extended sockets APIs (provided by Communications Server)

A conversational model - or a request/reply model

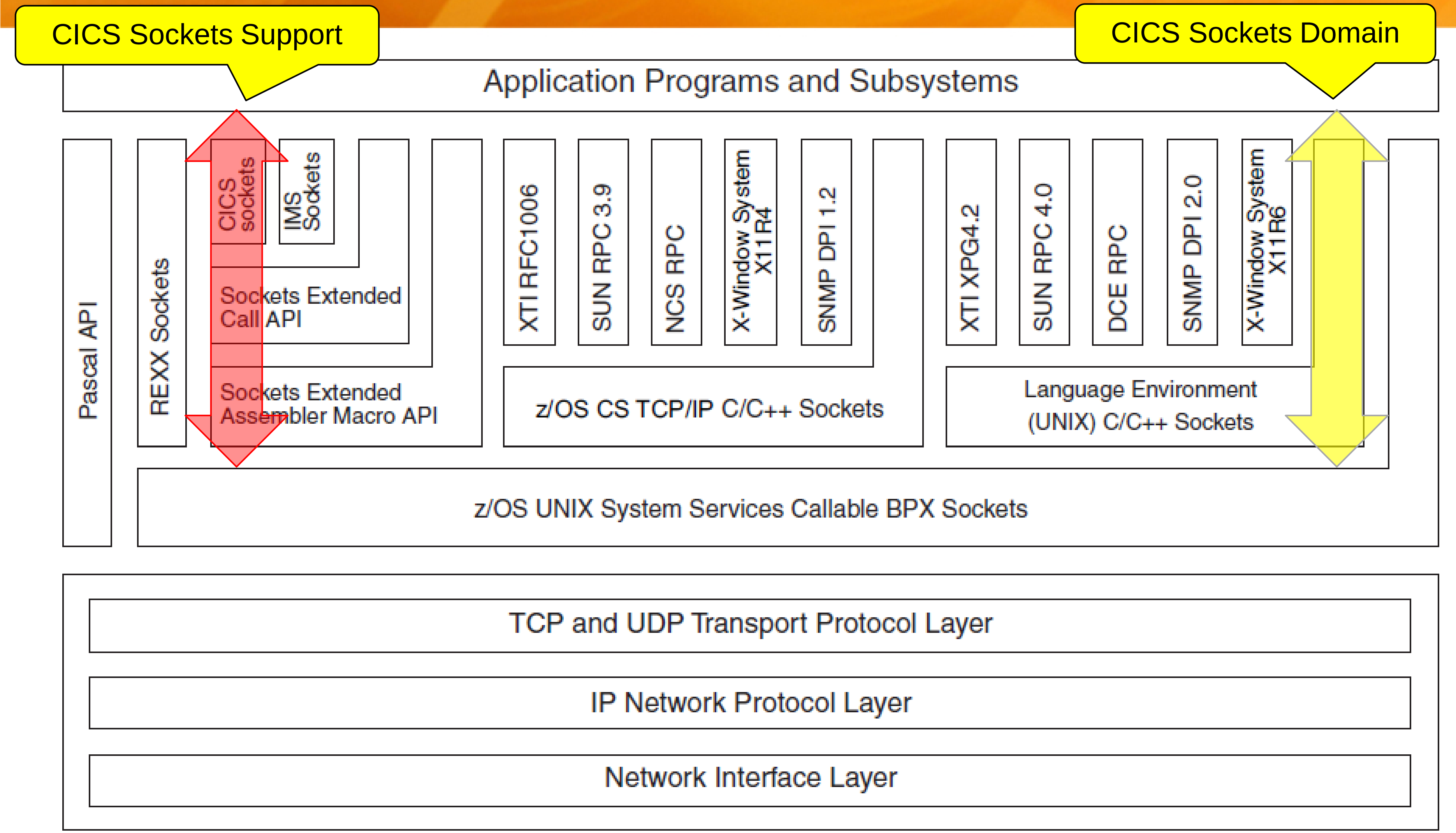
A request/reply model

A CICS Sockets Domain transaction does not have direct access to the socket, but communicates with CICS Sockets Domain services to receive a request and to send a reply over a socket.

The listeners are the 'servers' as seen from a TCP/IP perspective.

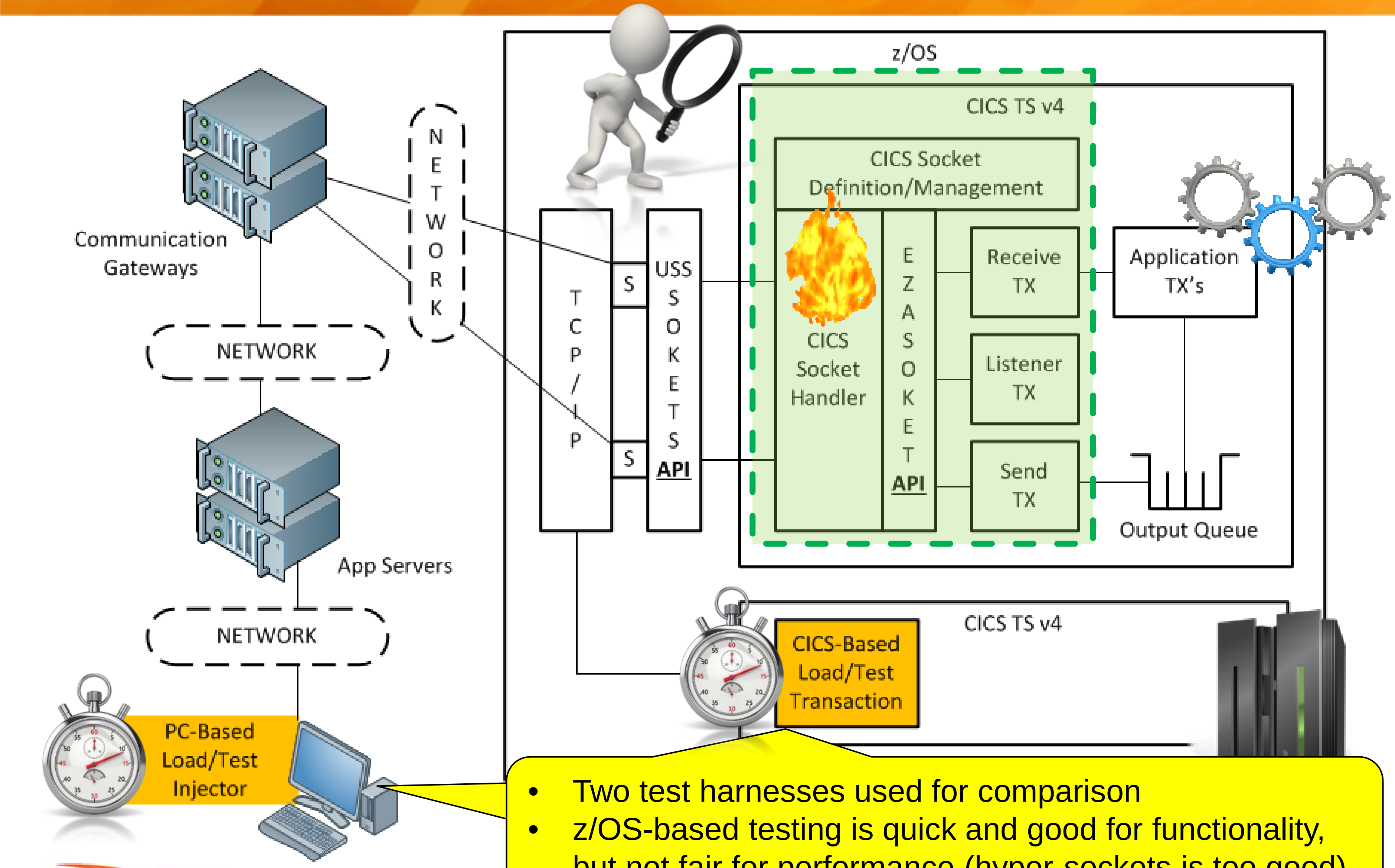
These services are based on the UNIX System Services C/C++ sockets API (provided by Language Environment) and the UNIX System Services callable APIs

CICS Sockets Pathway


CICS Sockets Support
CICS Sockets Domain
Application Programs and Subsystems
Pascal API
REXX Sockets
CICS sockets
IMS Sockets
Sockets Extended Call API
Sockets Extended Assembler Macro API
XTI RFC1006
SUN RPC 3.9
NCS RPC
X-Window System X11 R4
SNMP DPI 1.2
z/OS CS TCP/IP C/C++ Sockets
XTI XPG4.2
SUN RPC 4.0
DCE RPC
SNMP DPI 2.0
X-Window System X11 R6
Language Environment (UNIX) C/C++ Sockets
z/OS UNIX System Services Callable BPX Sockets
TCP and UDP Transport Protocol Layer
IP Network Protocol Layer
Network Interface Layer

z/OS Communications Server, IP Sockets Application Programming Interface Guide and Reference

Test Methodology



- Two test harnesses used for comparison
- z/OS-based testing is quick and good for functionality, but not fair for performance (hyper-sockets is too good)

Standard Test Cycle

- **Each test cycle caused the gateway to:**
  - **Open 2 sockets via Listener TX**
  - **Send/Receive TXs started to handle socket I/O**
  - **Generate 2,500 request-response iterations (no delays)**
  - **Each request caused a LINK to a customer program**
  - **Bytes in/out modeled for average production use case**
- **Benchmarks run:**
  - **1 concurrent test cycle**
  - **5 concurrent test cycles (10 sockets and 12,500 iterations)**

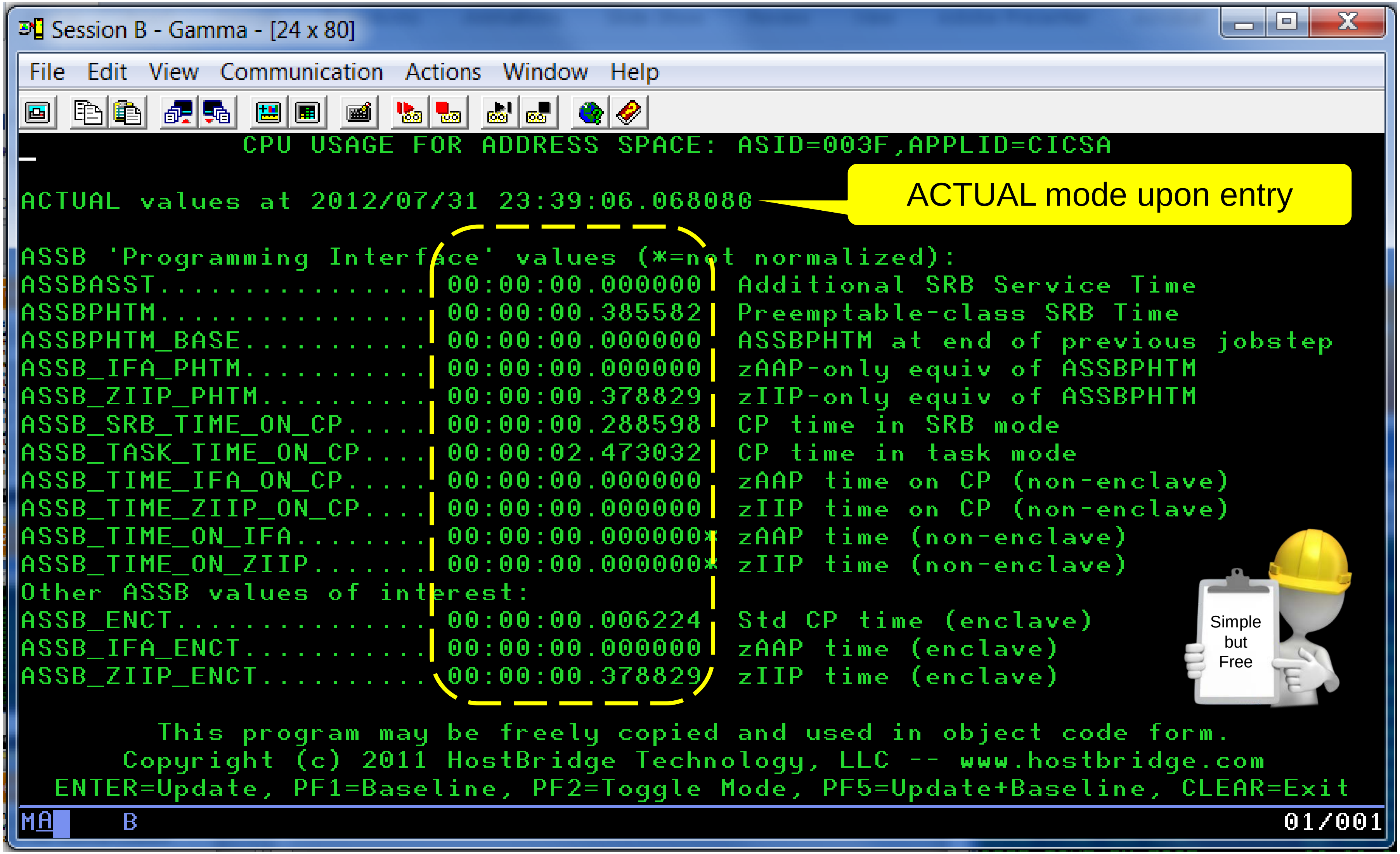
Selected to keep total region-level CPU use to a manageable level on test LPAR

- **Objectives:**
  - **Measure region-level CPU burn for various configurations**
  - **Differentiate between CPU burn associated with Socket apps and Socket infrastructure**

Tooling Developed

- It's difficult to get a snapshot of a CICS region's total resource consumption that is:
  - high-resolution (microseconds)
  - low-overhead
  - Immediate
  - Includes zIIP and zAAP
- Ended up developing four tools:
  - A CICS transaction to provide a summary of MVS ASSB timers (HBZT)
  - A CICS XMNOUT exit to log transaction metrics via WTO
  - A CICS-based test harness to generate socket activity
  - A PC-based test harness to augment the customer's
- The combination allowed us to:
  - drive testing fast
  - quickly assess results from all angles
- Special thanks to:
  - Larry Lawler (UNICOM)
  - Ed Jaffe (Phoenix Software)
  - Scott Glenn (HostBridge)
- For info on HBZT, see me after session

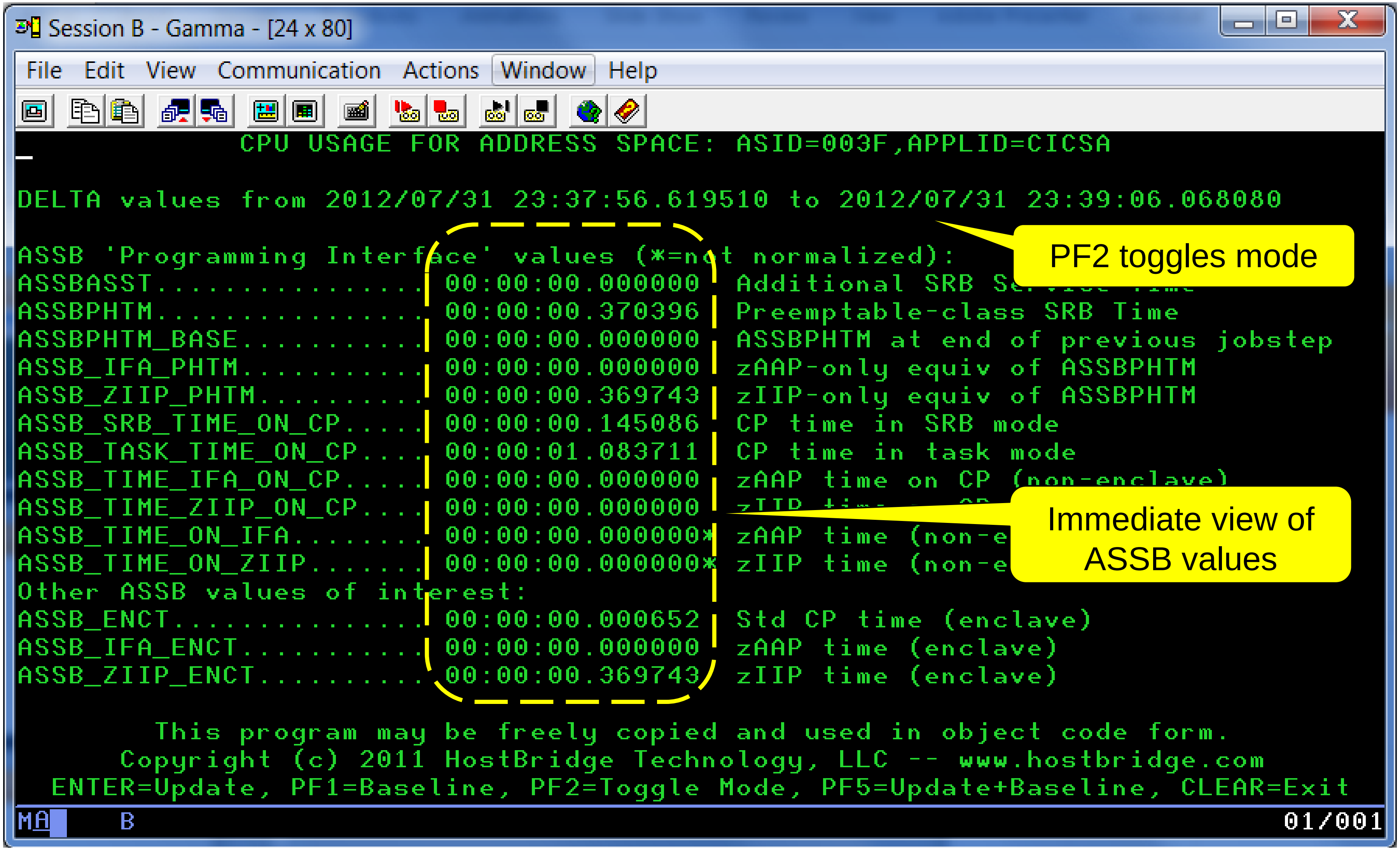
HBZT CPU Transaction


Session B - Gamma - [24 x 80]
File Edit View Communication Actions Window Help
MA B
01/001
Simple
but
Free

```
             CPU USAGE FOR ADDRESS SPACE: ASID=003F,APPLID=CICSA

ACTUAL values at 2012/07/31 23:39:06.068080

ASSB 'Programming Interface' values (*=not normalized):
ASSBASST................ 00:00:00.000000  Additional SRB Service Time
ASSBPHTM................ 00:00:00.385582  Preemptable-class SRB Time
ASSBPHTM_BASE........... 00:00:00.000000  ASSBPHTM at end of previous jobstep
ASSB_IFA_PHTM........... 00:00:00.000000  zAAP-only equiv of ASSBPHTM
ASSB_ZIIP_PHTM.......... 00:00:00.378829  zIIP-only equiv of ASSBPHTM
ASSB_SRB_TIME_ON_CP..... 00:00:00.288598  CP time in SRB mode
ASSB_TASK_TIME_ON_CP.... 00:00:02.473032  CP time in task mode
ASSB_TIME_IFA_ON_CP..... 00:00:00.000000  zAAP time on CP (non-enclave)
ASSB_TIME_ZIIP_ON_CP.... 00:00:00.000000  zIIP time on CP (non-enclave)
ASSB_TIME_ON_IFA........ 00:00:00.000000* zAAP time (non-enclave)
ASSB_TIME_ON_ZIIP....... 00:00:00.000000* zIIP time (non-enclave)
Other ASSB values of interest:
ASSB_ENCT............... 00:00:00.006224  Std CP time (enclave)
ASSB_IFA_ENCT........... 00:00:00.000000  zAAP time (enclave)
ASSB_ZIIP_ENCT.......... 00:00:00.378829  zIIP time (enclave)

        This program may be freely copied and used in object code form.
      Copyright (c) 2011 HostBridge Technology, LLC -- www.hostbridge.com
  ENTER=Update, PF1=Baseline, PF2=Toggle Mode, PF5=Update+Baseline, CLEAR=Exit
```

ACTUAL mode upon entry

HBZT CPU Transaction

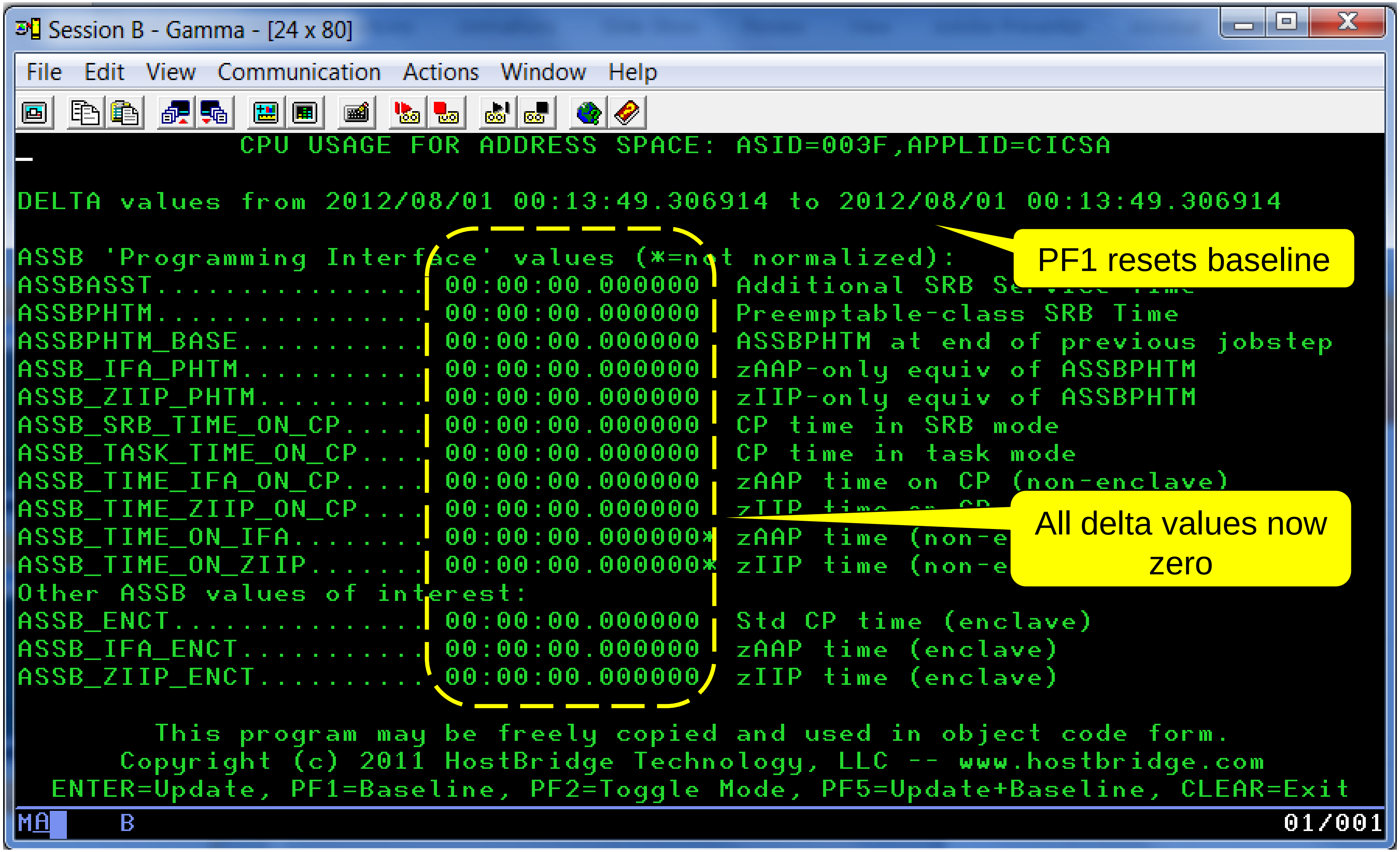

Session B - Gamma - [24 x 80]
File Edit View Communication Actions Window Help

```
             CPU USAGE FOR ADDRESS SPACE: ASID=003F,APPLID=CICSA

DELTA values from 2012/07/31 23:37:56.619510 to 2012/07/31 23:39:06.068080

ASSB 'Programming Interface' values (*=not normalized):
ASSBASST................ 00:00:00.000000  Additional SRB Se
ASSBPHTM................ 00:00:00.370396  Preemptable-class SRB Time
ASSBPHTM_BASE........... 00:00:00.000000  ASSBPHTM at end of previous jobstep
ASSB_IFA_PHTM........... 00:00:00.000000  zAAP-only equiv of ASSBPHTM
ASSB_ZIIP_PHTM.......... 00:00:00.369743  zIIP-only equiv of ASSBPHTM
ASSB_SRB_TIME_ON_CP..... 00:00:00.145086  CP time in SRB mode
ASSB_TASK_TIME_ON_CP.... 00:00:01.083711  CP time in task mode
ASSB_TIME_IFA_ON_CP..... 00:00:00.000000  zAAP time on CP (non-enclave)
ASSB_TIME_ZIIP_ON_CP.... 00:00:00.000000
ASSB_TIME_ON_IFA........ 00:00:00.000000* zAAP time (non-e
ASSB_TIME_ON_ZIIP....... 00:00:00.000000* zIIP time (non-e
Other ASSB values of interest:
ASSB_ENCT............... 00:00:00.000652  Std CP time (enclave)
ASSB_IFA_ENCT........... 00:00:00.000000  zAAP time (enclave)
ASSB_ZIIP_ENCT.......... 00:00:00.369743  zIIP time (enclave)

        This program may be freely copied and used in object code form.
      Copyright (c) 2011 HostBridge Technology, LLC -- www.hostbridge.com
  ENTER=Update, PF1=Baseline, PF2=Toggle Mode, PF5=Update+Baseline, CLEAR=Exit
MA      B                                                                01/001
```

PF2 toggles mode

Immediate view of ASSB values

HBZT CPU Transaction

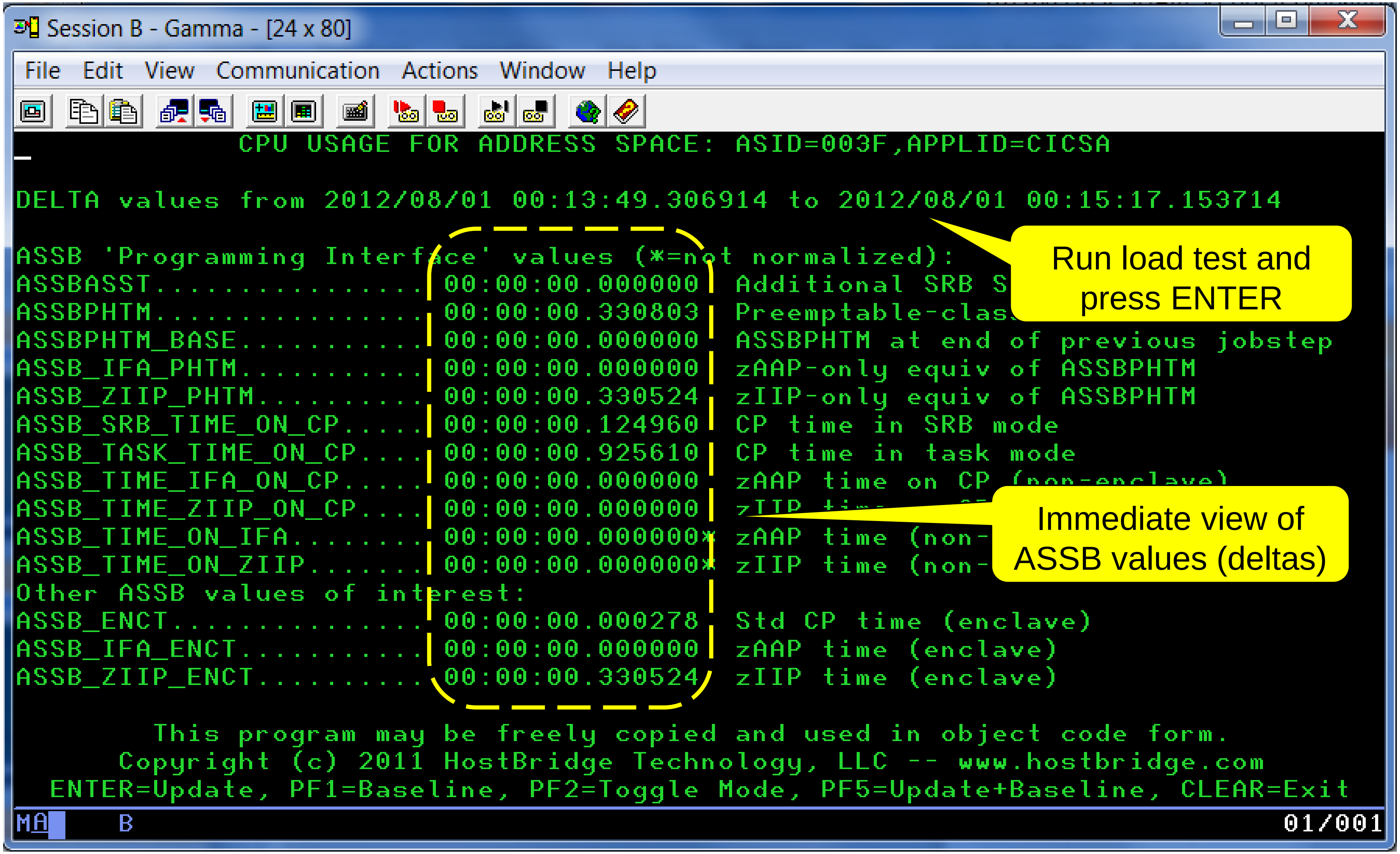


```
              CPU USAGE FOR ADDRESS SPACE: ASID=003F,APPLID=CICSA

DELTA values from 2012/08/01 00:13:49.306914 to 2012/08/01 00:13:49.306914

ASSB 'Programming Interface' values (*=not normalized):
ASSBASST................ 00:00:00.000000  Additional SRB Service Time
ASSBPHTM................ 00:00:00.000000  Preemptable-class SRB Time
ASSBPHTM_BASE........... 00:00:00.000000  ASSBPHTM at end of previous jobstep
ASSB_IFA_PHTM........... 00:00:00.000000  zAAP-only equiv of ASSBPHTM
ASSB_ZIIP_PHTM.......... 00:00:00.000000  zIIP-only equiv of ASSBPHTM
ASSB_SRB_TIME_ON_CP..... 00:00:00.000000  CP time in SRB mode
ASSB_TASK_TIME_ON_CP.... 00:00:00.000000  CP time in task mode
ASSB_TIME_IFA_ON_CP..... 00:00:00.000000  zAAP time on CP (non-enclave)
ASSB_TIME_ZIIP_ON_CP.... 00:00:00.000000  zIIP time on CP
ASSB_TIME_ON_IFA........ 00:00:00.000000* zAAP time (non-e
ASSB_TIME_ON_ZIIP....... 00:00:00.000000* zIIP time (non-e
Other ASSB values of interest:
ASSB_ENCT............... 00:00:00.000000  Std CP time (enclave)
ASSB_IFA_ENCT........... 00:00:00.000000  zAAP time (enclave)
ASSB_ZIIP_ENCT.......... 00:00:00.000000  zIIP time (enclave)

          This program may be freely copied and used in object code form.
        Copyright (c) 2011 HostBridge Technology, LLC -- www.hostbridge.com
  ENTER=Update, PF1=Baseline, PF2=Toggle Mode, PF5=Update+Baseline, CLEAR=Exit
MA      B                                                                01/001
```

PF1 resets baseline

All delta values now zero

HBZT CPU Transaction



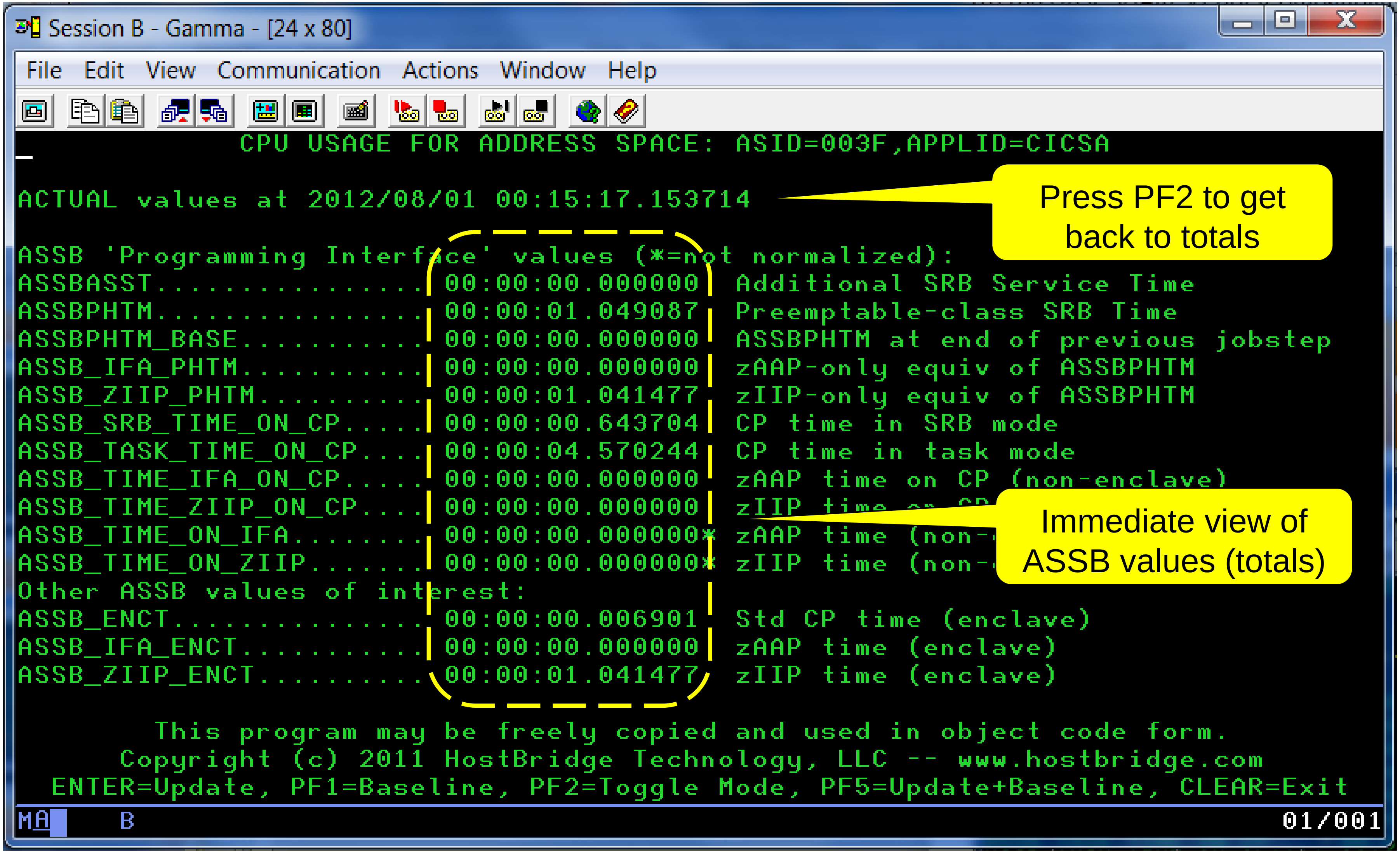
```
             CPU USAGE FOR ADDRESS SPACE: ASID=003F,APPLID=CICSA

DELTA values from 2012/08/01 00:13:49.306914 to 2012/08/01 00:15:17.153714

ASSB 'Programming Interface' values (*=not normalized):
ASSBASST................ 00:00:00.000000  Additional SRB S
ASSBPHTM................ 00:00:00.330803  Preemptable-clas
ASSBPHTM_BASE........... 00:00:00.000000  ASSBPHTM at end of previous jobstep
ASSB_IFA_PHTM........... 00:00:00.000000  zAAP-only equiv of ASSBPHTM
ASSB_ZIIP_PHTM.......... 00:00:00.330524  zIIP-only equiv of ASSBPHTM
ASSB_SRB_TIME_ON_CP..... 00:00:00.124960  CP time in SRB mode
ASSB_TASK_TIME_ON_CP.... 00:00:00.925610  CP time in task mode
ASSB_TIME_IFA_ON_CP..... 00:00:00.000000  zAAP time on CP (non-enclave)
ASSB_TIME_ZIIP_ON_CP.... 00:00:00.000000  zIIP
ASSB_TIME_ON_IFA........ 00:00:00.000000* zAAP time (non-
ASSB_TIME_ON_ZIIP....... 00:00:00.000000* zIIP time (non-
Other ASSB values of interest:
ASSB_ENCT............... 00:00:00.000278  Std CP time (enclave)
ASSB_IFA_ENCT........... 00:00:00.000000  zAAP time (enclave)
ASSB_ZIIP_ENCT.......... 00:00:00.330524  zIIP time (enclave)

        This program may be freely copied and used in object code form.
      Copyright (c) 2011 HostBridge Technology, LLC -- www.hostbridge.com
  ENTER=Update, PF1=Baseline, PF2=Toggle Mode, PF5=Update+Baseline, CLEAR=Exit
MA      B                                                                01/001
```

Run load test and press ENTER

Immediate view of ASSB values (deltas)

HOSTBRIDGE® TECHNOLOGY

HBZT CPU Transaction



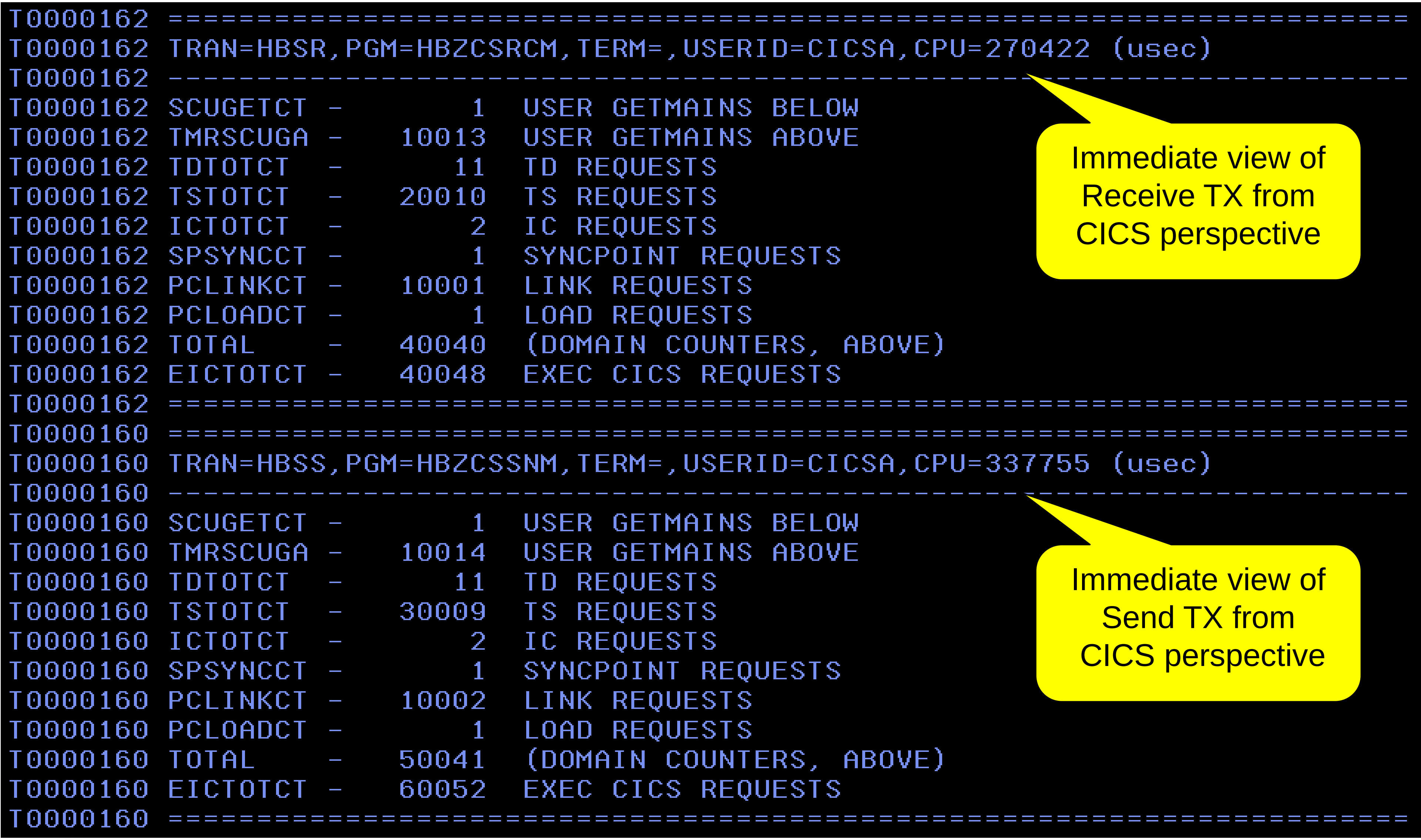
```
             CPU USAGE FOR ADDRESS SPACE: ASID=003F,APPLID=CICSA
_

ACTUAL values at 2012/08/01 00:15:17.153714

ASSB 'Programming Interface' values (*=not normalized):
ASSBASST................ 00:00:00.000000  Additional SRB Service Time
ASSBPHTM................ 00:00:01.049087  Preemptable-class SRB Time
ASSBPHTM_BASE........... 00:00:00.000000  ASSBPHTM at end of previous jobstep
ASSB_IFA_PHTM........... 00:00:00.000000  zAAP-only equiv of ASSBPHTM
ASSB_ZIIP_PHTM.......... 00:00:01.041477  zIIP-only equiv of ASSBPHTM
ASSB_SRB_TIME_ON_CP..... 00:00:00.643704  CP time in SRB mode
ASSB_TASK_TIME_ON_CP.... 00:00:04.570244  CP time in task mode
ASSB_TIME_IFA_ON_CP..... 00:00:00.000000  zAAP time on CP (non-enclave)
ASSB_TIME_ZIIP_ON_CP.... 00:00:00.000000  zIIP time on CP
ASSB_TIME_ON_IFA........ 00:00:00.000000* zAAP time (non-
ASSB_TIME_ON_ZIIP....... 00:00:00.000000* zIIP time (non-
Other ASSB values of interest:
ASSB_ENCT............... 00:00:00.006901  Std CP time (enclave)
ASSB_IFA_ENCT........... 00:00:00.000000  zAAP time (enclave)
ASSB_ZIIP_ENCT.......... 00:00:01.041477  zIIP time (enclave)

        This program may be freely copied and used in object code form.
      Copyright (c) 2011 HostBridge Technology, LLC -- www.hostbridge.com
  ENTER=Update, PF1=Baseline, PF2=Toggle Mode, PF5=Update+Baseline, CLEAR=Exit
MA     B                                                                01/001
```

Press PF2 to get back to totals

Immediate view of ASSB values (totals)

HOSTBRIDGE® TECHNOLOGY

XMNOUT Exit Metrics



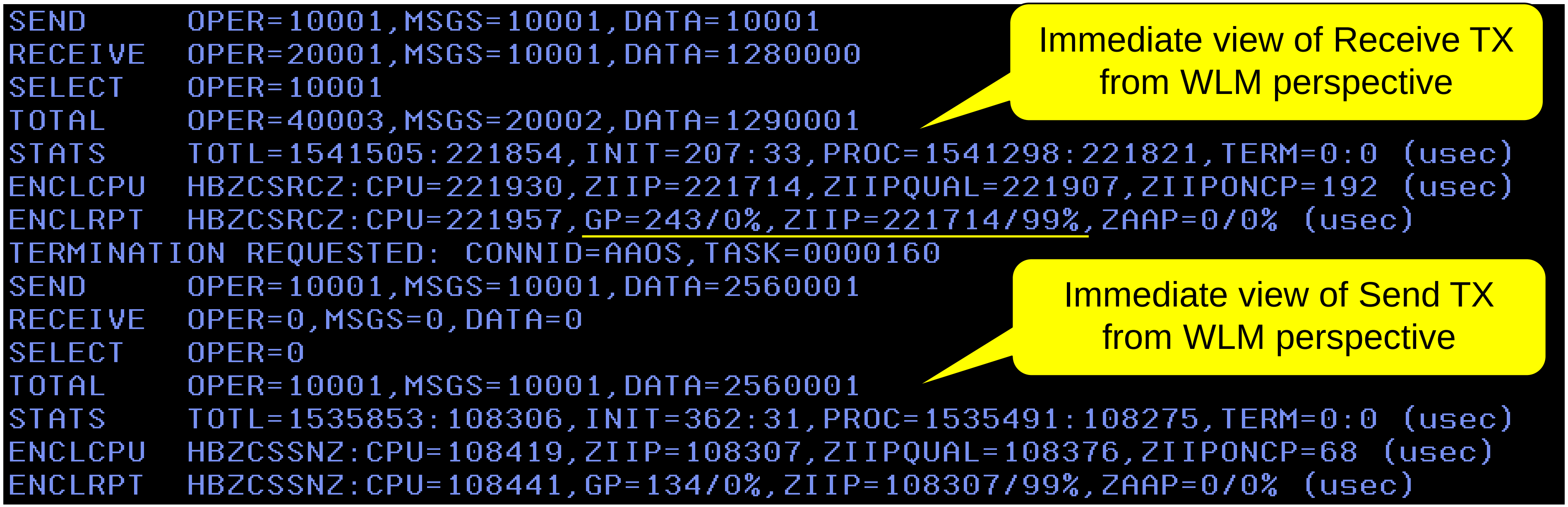
```
T0000162 ======================================================================
T0000162 TRAN=HBSR,PGM=HBZCSRCM,TERM=,USERID=CICSA,CPU=270422 (usec)
T0000162 ----------------------------------------------------------------------
T0000162 SCUGETCT -        1  USER GETMAINS BELOW
T0000162 TMRSCUGA -    10013  USER GETMAINS ABOVE
T0000162 TDTOTCT  -       11  TD REQUESTS
T0000162 TSTOTCT  -    20010  TS REQUESTS
T0000162 ICTOTCT  -        2  IC REQUESTS
T0000162 SPSYNCCT -        1  SYNCPOINT REQUESTS
T0000162 PCLINKCT -    10001  LINK REQUESTS
T0000162 PCLOADCT -        1  LOAD REQUESTS
T0000162 TOTAL    -    40040  (DOMAIN COUNTERS, ABOVE)
T0000162 EICTOTCT -    40048  EXEC CICS REQUESTS
T0000162 ======================================================================
T0000160 ======================================================================
T0000160 TRAN=HBSS,PGM=HBZCSSNM,TERM=,USERID=CICSA,CPU=337755 (usec)
T0000160 ----------------------------------------------------------------------
T0000160 SCUGETCT -        1  USER GETMAINS BELOW
T0000160 TMRSCUGA -    10014  USER GETMAINS ABOVE
T0000160 TDTOTCT  -       11  TD REQUESTS
T0000160 TSTOTCT  -    30009  TS REQUESTS
T0000160 ICTOTCT  -        2  IC REQUESTS
T0000160 SPSYNCCT -        1  SYNCPOINT REQUESTS
T0000160 PCLINKCT -    10002  LINK REQUESTS
T0000160 PCLOADCT -        1  LOAD REQUESTS
T0000160 TOTAL    -    50041  (DOMAIN COUNTERS, ABOVE)
T0000160 EICTOTCT -    60052  EXEC CICS REQUESTS
T0000160 ======================================================================
```

Immediate view of Receive TX from CICS perspective

Immediate view of Send TX from CICS perspective

Program Metrics

- **In order to triangulate all views of resource consumption, we included instrumentation in our code (provided it would not bias the results):**
  - **Simple activity & data counters**
  - **WLM Enclave metrics**



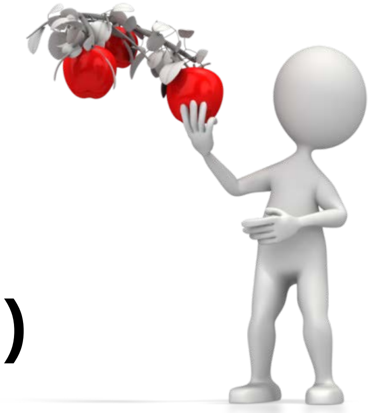
```
SEND     OPER=10001,MSGS=10001,DATA=10001
RECEIVE  OPER=20001,MSGS=10001,DATA=1280000
SELECT   OPER=10001
TOTAL    OPER=40003,MSGS=20002,DATA=1290001
STATS    TOTL=1541505:221854,INIT=207:33,PROC=1541298:221821,TERM=0:0 (usec)
ENCLCPU  HBZCSRCZ:CPU=221930,ZIIP=221714,ZIIPQUAL=221907,ZIIPONCP=192 (usec)
ENCLRPT  HBZCSRCZ:CPU=221957,GP=243/0%,ZIIP=221714/99%,ZAAP=0/0% (usec)
TERMINATION REQUESTED: CONNID=AAOS,TASK=0000160
SEND     OPER=10001,MSGS=10001,DATA=2560001
RECEIVE  OPER=0,MSGS=0,DATA=0
SELECT   OPER=0
TOTAL    OPER=10001,MSGS=10001,DATA=2560001
STATS    TOTL=1535853:108306,INIT=362:31,PROC=1535491:108275,TERM=0:0 (usec)
ENCLCPU  HBZCSSNZ:CPU=108419,ZIIP=108307,ZIIPQUAL=108376,ZIIPONCP=68 (usec)
ENCLRPT  HBZCSSNZ:CPU=108441,GP=134/0%,ZIIP=108307/99%,ZAAP=0/0% (usec)
```

Immediate view of Receive TX from WLM perspective

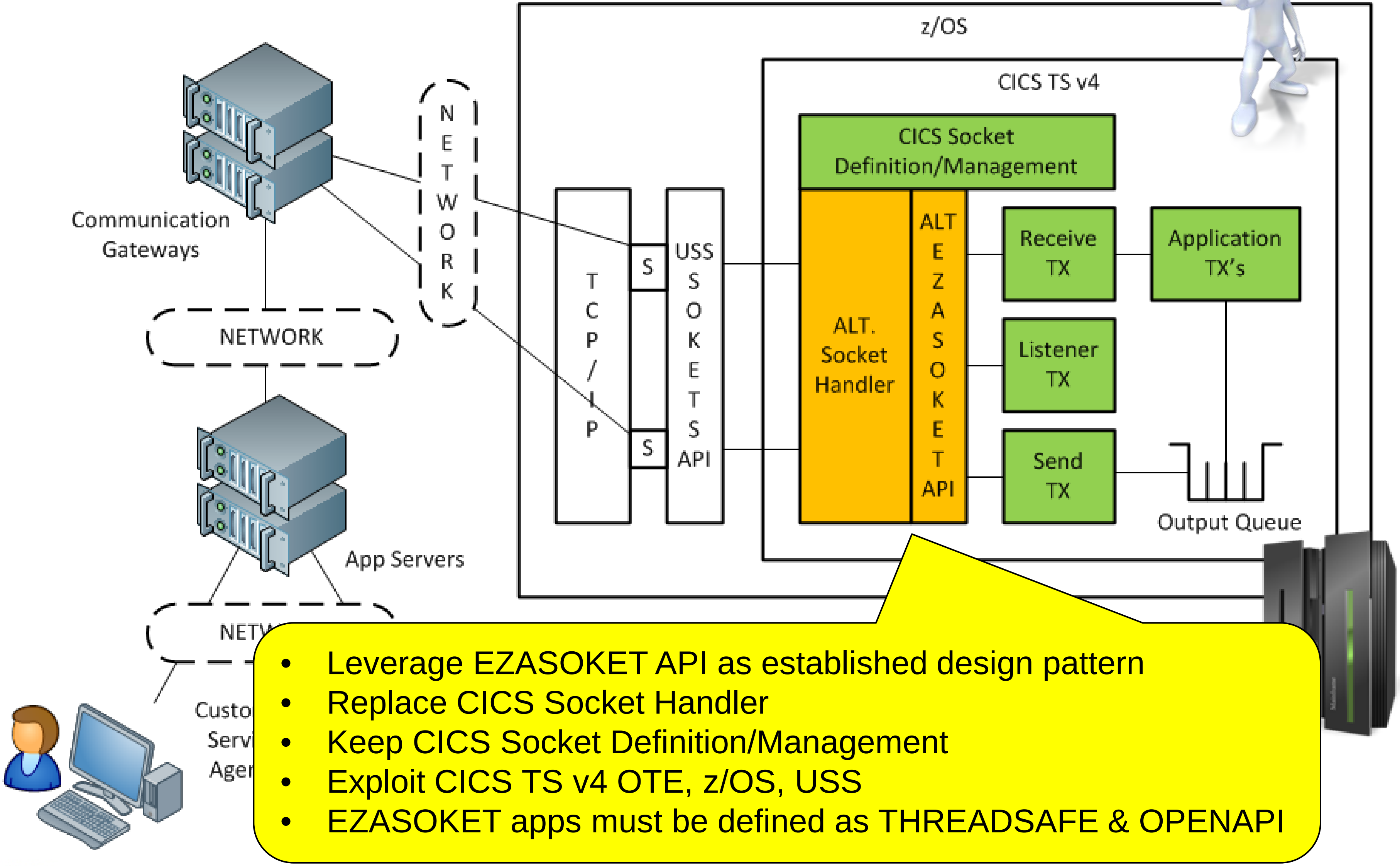
Immediate view of Send TX from WLM perspective

Where the Data Led Us

- Under volume testing, the CPU burn associated with the CICS Sockets Support was measurable and linear (confirmed customer's theory)
- I won't characterize it as "high" or "low" because the only thing that mattered was whether it could be lower (or not so linear)
- Thus, we began to:
  - Isolate various components and their impact
  - Consider how to provide alternative functionality (but complimentary to CICS TS)
- Low hanging fruit seemed to be CICS Socket Handler (via EZASOKET API)



Solution 1

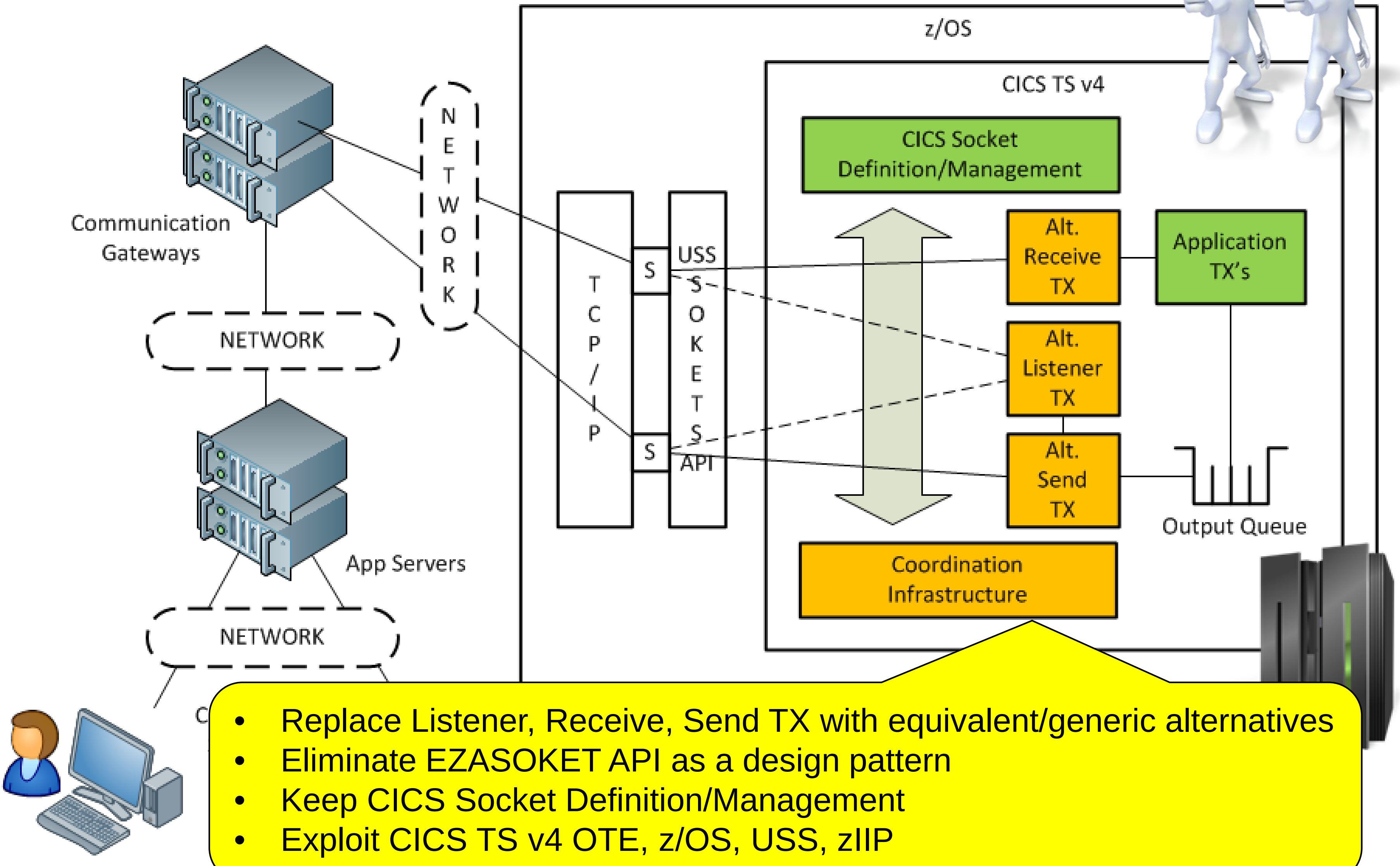


- Leverage EZASOKET API as established design pattern
- Replace CICS Socket Handler
- Keep CICS Socket Definition/Management
- Exploit CICS TS v4 OTE, z/OS, USS
- EZASOKET apps must be defined as THREADSAFE & OPENAPI

Solution 1 Assessment

- **Good…**
  - **The Alt. Socket Handler lowered GP CPU burn associated with Socket I/O**
  - **All it required was a re-link of apps that used EZASOKET API (with alternate load module)**
  - **Transparent to existing user-written Listeners, Sender and Receiver TXs**
- **However…**
  - **EZASOKET API emulation seemed to be a bit of needless overhead (e.g., parameter marshaling and transformation)**
  - **zIIP enablement opportunity wasn't optimal due to task switching**
- **But wait…**
  - **The design patterns for CICS-based Listeners, Receivers and Senders are fairly common**

Solution 2

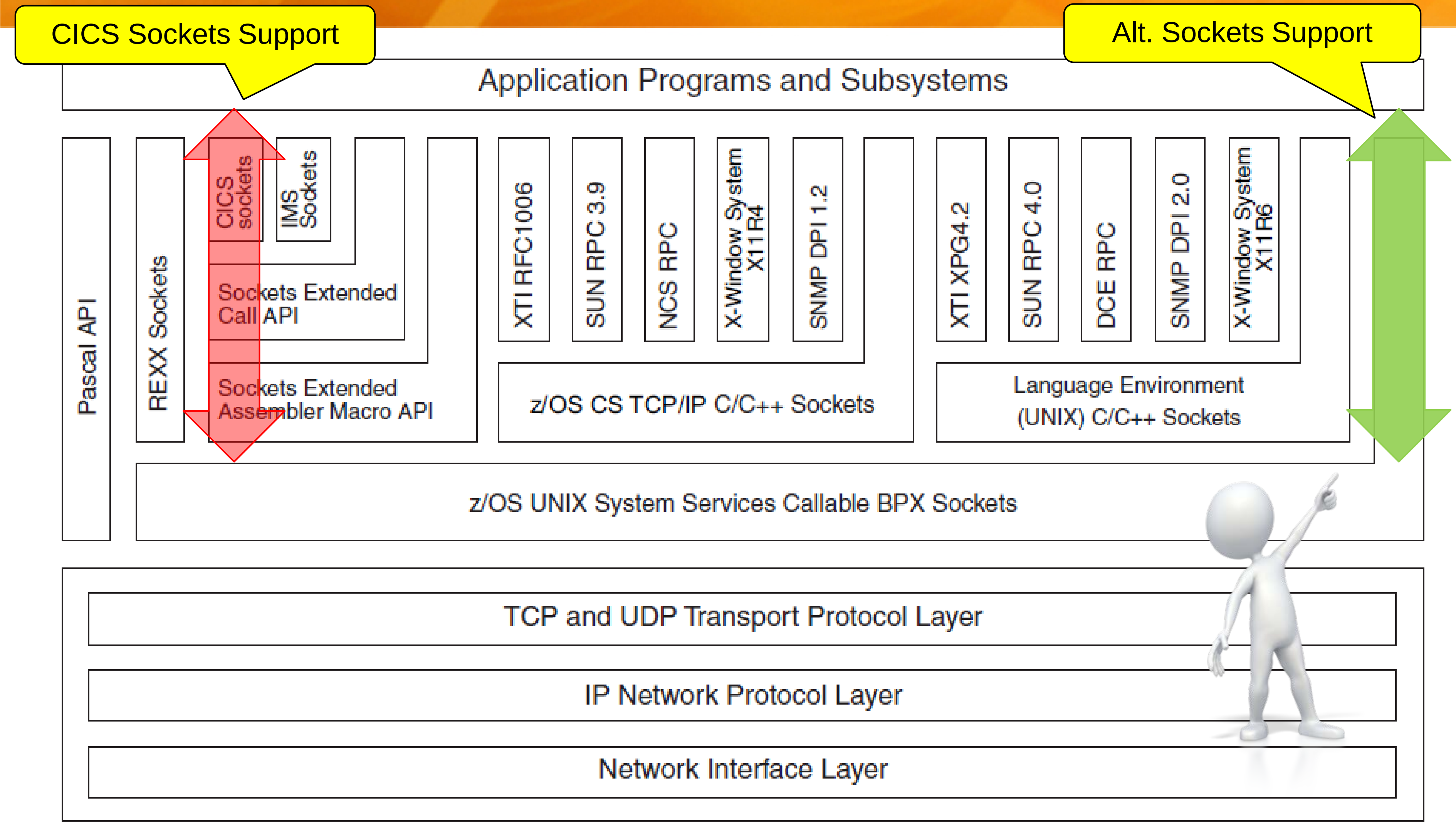


- Replace Listener, Receive, Send TX with equivalent/generic alternatives
- Eliminate EZASOKET API as a design pattern
- Keep CICS Socket Definition/Management
- Exploit CICS TS v4 OTE, z/OS, USS, zIIP

Solution 2 Assessment

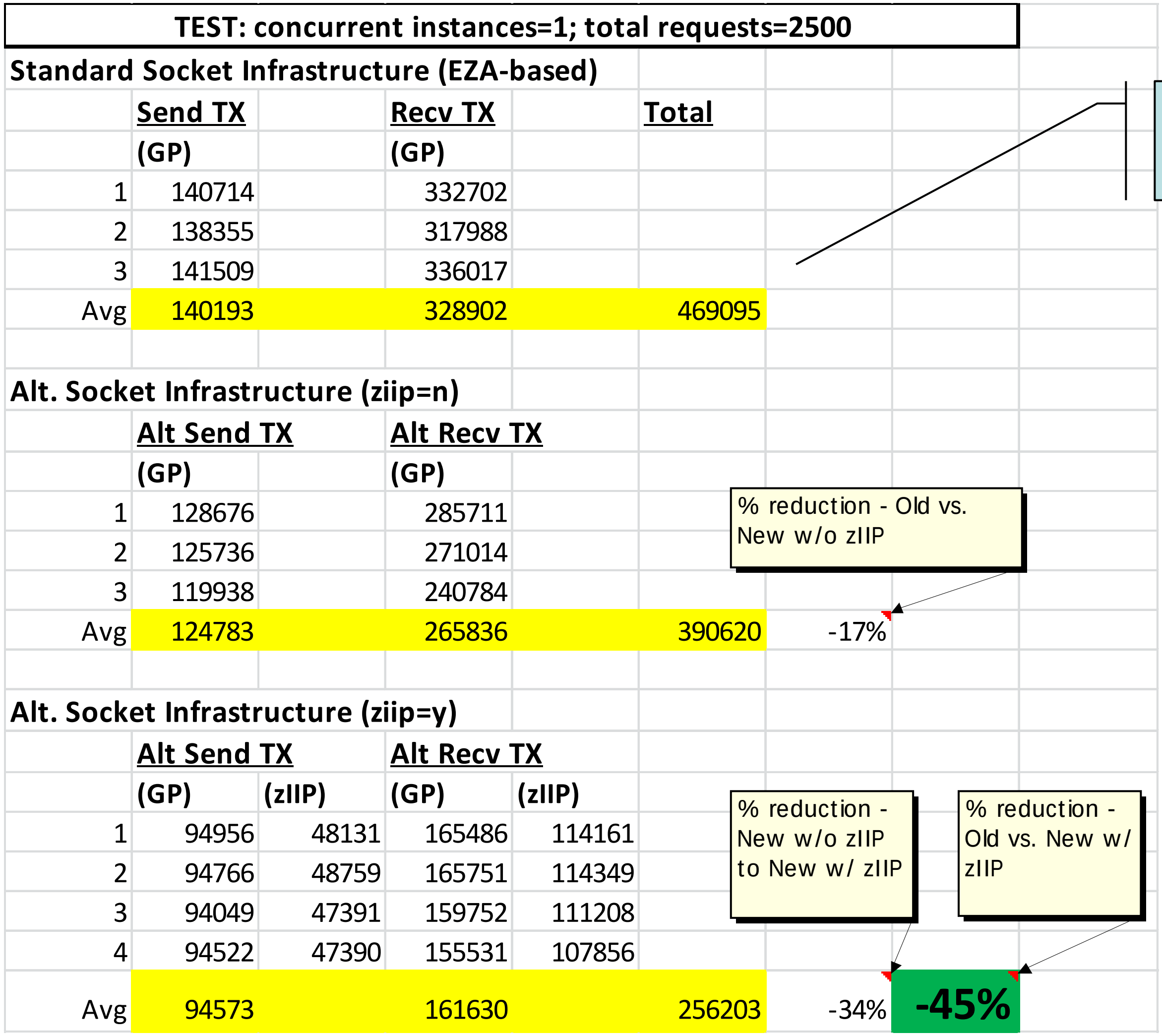
- Very Good…
  - GP CPU burn associated with Socket I/O went down further
  - EZASOKET API emulation eliminated (all components use native sockets)
  - Transparent to the customer’s applications
  - CICS Socket definition/management leveraged
    - EZAO still used to Configure, Start, or Stop Listeners
- zIIP enablement potential maximized
  - Minimal task switching
  - Customer application code not zIIP enabled (per IBM-ISV T&C’s)

Pathway - Old vs. New


CICS Sockets Support
Alt. Sockets Support
Application Programs and Subsystems
Pascal API
REXX Sockets
CICS sockets
IMS Sockets
Sockets Extended Call API
Sockets Extended Assembler Macro API
XTI RFC1006
SUN RPC 3.9
NCS RPC
X-Window System X11R4
SNMP DPI 1.2
z/OS CS TCP/IP C/C++ Sockets
XTI XPG4.2
SUN RPC 4.0
DCE RPC
SNMP DPI 2.0
X-Window System X11R6
Language Environment (UNIX) C/C++ Sockets
z/OS UNIX System Services Callable BPX Sockets
TCP and UDP Transport Protocol Layer
IP Network Protocol Layer
Network Interface Layer

z/OS Communications Server, IP Sockets Application Programming Interface Guide and Reference

Test Results



TEST: concurrent instances=1; total requests=2500							
Standard Socket Infrastructure (EZA-based)							
	Send TX		**Recv TX**		**Total**		
	(GP)		**(GP)**				
1	140714		332702				
2	138355		317988				
3	141509		336017				
Avg	140193		328902		469095		
Alt. Socket Infrastructure (ziip=n)							
	Alt Send TX		**Alt Recv TX**				
	(GP)		**(GP)**				
1	128676		285711				
2	125736		271014				
3	119938		240784				
Avg	124783		265836		390620	-17%	
Alt. Socket Infrastructure (ziip=y)							
	Alt Send TX		**Alt Recv TX**				
	(GP)	**(zIIP)**	**(GP)**	**(zIIP)**			
1	94956	48131	165486	114161			
2	94766	48759	165751	114349			
3	94049	47391	159752	111208			
4	94522	47390	155531	107856			
Avg	94573		161630		256203	-34%	**-45%**

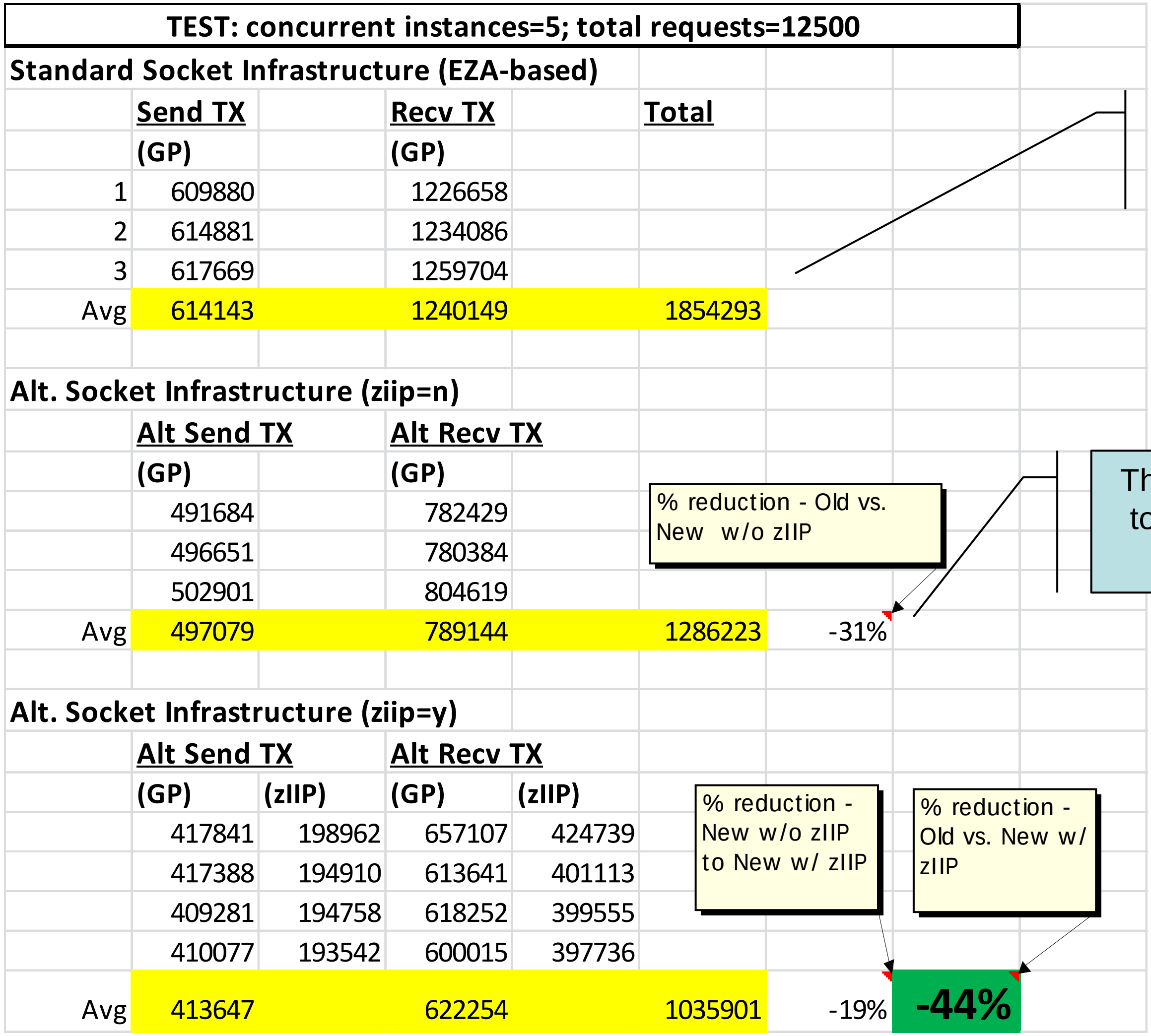
All times in microseconds

% reduction - Old vs. New w/o zIIP

% reduction - New w/o zIIP to New w/ zIIP

% reduction - Old vs. New w/ zIIP

Test Results (w/ Concurrency)



TEST: concurrent instances=5; total requests=12500							
Standard Socket Infrastructure (EZA-based)							
	Send TX		Recv TX		Total		
	(GP)		(GP)				
1	609880		1226658				
2	614881		1234086				
3	617669		1259704				
Avg	614143		1240149		1854293		
Alt. Socket Infrastructure (ziip=n)							
	Alt Send TX		Alt Recv TX				
	(GP)		(GP)				
	491684		782429				
	496651		780384				
	502901		804619				
Avg	497079		789144		1286223	-31%	
Alt. Socket Infrastructure (ziip=y)							
	Alt Send TX		Alt Recv TX				
	(GP)	(zIIP)	(GP)	(zIIP)			
	417841	198962	657107	424739			
	417388	194910	613641	401113			
	409281	194758	618252	399555			
	410077	193542	600015	397736			
Avg	413647		622254		1035901	-19%	-44%

% reduction - Old vs. New w/o zIIP

% reduction - New w/o zIIP to New w/ zIIP

% reduction - Old vs. New w/ zIIP

All times in microseconds

The TCP/IP stack seems to get more efficient the harder you load it

Summary

- CICS Socket Support has been a workhorse for a long time -- it’s earned it’s keep!
- CICS TS Open Transaction Environment continues to evolve and permits new opportunities for customers and ISV’s -- thank you Hursley Lab
- An example is the Alternate Socket Handler described in this presentation
- This approach is applicable to any customer who relies heavily on CICS Socket Support
  - zIIP support can only be provided by a licensed ISV
- The final estimate was that the solution would reduce the customer’s CPU usage (GP) associated with CICS Sockets I/O by:
  - 20% without zIIP-enablement
  - 40% with zIIP-enablement
- They were pleased

HOSTBRIDGE® TECHNOLOGY