Watson & Walker
zRoadshow

Session 26202

Mario Bezzi (mario@watsonwalker.com)

Frank Kyne (frank@watsonwalker.com)

Welcome/Adios

- Who are we?
  - Cheryl **Watson**, CEO of Watson & Walker Inc. since 1986, working on IBM mainframes since 1965 and her husband Tom **Walker**.
  - Brenda White, Mario Bezzi, Scott Barry, and Frank Kyne, plus Alan Murphy and Graham Horne.
- We publish Cheryl Watson's Tuning Letter (since 1991) and provide performance, sysplex, and software pricing consulting.
- Our latest addition is our Starter Pack Service Offering, where we help clients get the maximum value from their z/OS systems.
- We have two websites:
  - Watsonwalker.com   General web site, free access for everyone.
  - Watsonwalkerpublications.com   Subscriber-only – all Tuning Letters for last 29 years.
- This session provides some tidbits of information that we hope you will find valuable.

Topics - Mario

- **Our Important Messages Check**

- **Using SMF30 Counters to identify SIIS Activity**

- **Working with RAW SMF data in a workstation environment**

- **The IBM Record Generator**

- **Miscellanea**

Topics - Frank

- **SMF Record Generator**
- **z Open Automation Utilities**
- **Quick survey**
- **Some important dates**
- **User Key Common Storage Reminder for z/OS 2.4**
- **zEDC and z15 und zee IFAPRDxx member**
- **ERV refresher**
- **CPENABLE Update**
- **CICS Liberty High CPU APAR**
- **zIIPs, HiperDispatch, and Db2**
- **zPDT 'free' zIIPs**
- **Availability of FICON End-to-End Encryption**
- **DFSMSdss Tips**
- **Z Workload Interaction Correlator and z Workload Interaction Navigator**

Our Important Messages Check

The IMPORTANT_MESSAGES Check

- IBM recommends passing certain system messages to your alerting infrastructure so that the right people can be informed. However, identifying the specific messages that require this treatment, and using an available local automation package to detect and handle them is the customer's responsibility.

- The Watson and Walker IMPORTANT_MESSAGES check tracks specific system messages, raising a Health Checker exception if these messages are detected.

- It is provided with a default list of critical messages we build from Redbooks, manuals, APARs, customers' experiences. Users can extend this list with their own groups of environment-specific messages.

- We plan to periodically deliver updated versions of the important messages list, based on our continuous analysis of new software functions and on the feedback of users of the check.

What's new

- We asked **Adrian Burke** from Db2 for z/OS Development to help us identify Db2 specific messages which in his experience affected customers the most.

- 47 important Db2 messages were defined to a new Db2 specific Important Messages Check.
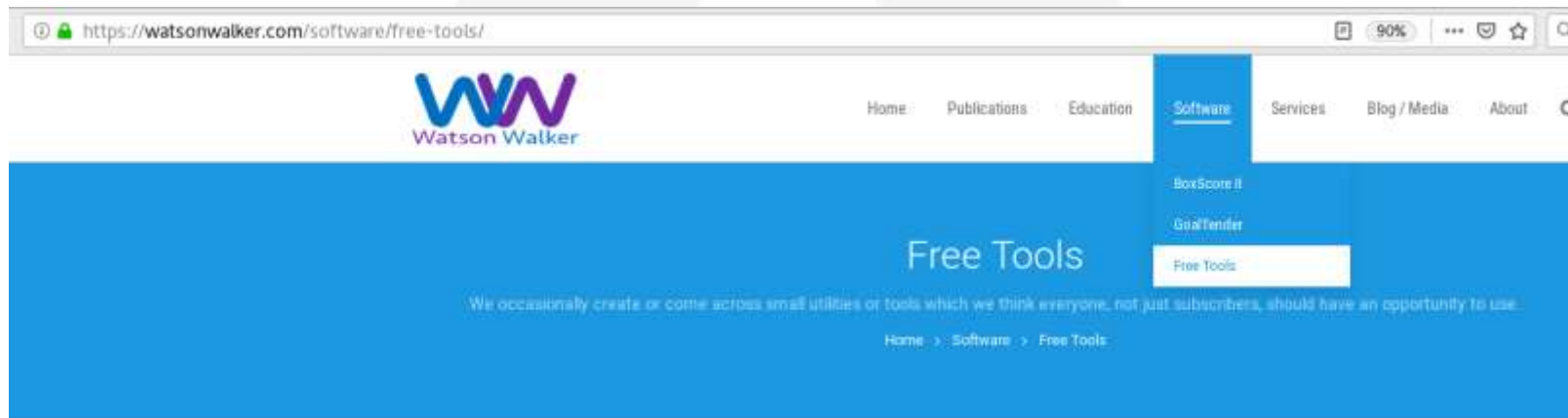
```
 Display   Filter   View   Print   Options   Search   Help
-----------------------------------------------------------------------
SDSF HEALTH CHECKER DISPLAY   S0W1                      LINE 66-84 (212)
COMMAND INPUT ===> █                                       SCROLL ===>
NP    NAME                              CheckOwner           State
      IMPORTANT_MESSAGES_CUSTOM         WATSONWALKER         ACTIVE(ENABLED)
      IMPORTANT_MESSAGES_SYS_CRIT       WATSONWALKER         ACTIVE(ENABLED)
      IMPORTANT_MESSAGES_WWD21911       WATSONWALKER         ACTIVE(ENABLED)
      IMPORTANT_MESSAGES_WWIM1905       WATSONWALKER         ACTIVE(ENABLED)
```

- These are in addition to the 152 we already track for z/OS plus the ones tagged as critical by z/OS.

Thank You Adrian!

Interested in getting it? Its free!

- We made the check available to the z/OS user community to help improve systems' availability.
- Some 70 customers downloaded it so far.



https://watsonwalker.com/software/free-tools/

Using SMF30 Counters to identify SIIS Activity

Store Into Instruction Stream (SIIS)

- Modern CPCs really don't like programs that store into the instruction stream (SIIS). As a result affected program's CPU utilization increases significantly.

- IBM has warned against this for years, but it is more of an issue now.

- Compilers are designed not to do this, however, if you have Assembler programs that do a lot of this, that could be an issue.

- CPU MF can be used to help identify the potential SIIS timeframes. This is not an exact measurement as the used metric has inherent noise, but experience has shown that when certain thresholds are exceeded, it is likely because of SIIS activity. SMF30COUNT data can be used to attempt identify the culprit by looking at individual program's CPI.

- For additional information see:

  Identifying "Store Into Instruction Stream"•(SIIS) Inefficiency by Using CPU MF Counters

  http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102806

Store Into Instruction Stream (SIIS)

SIIS Cost
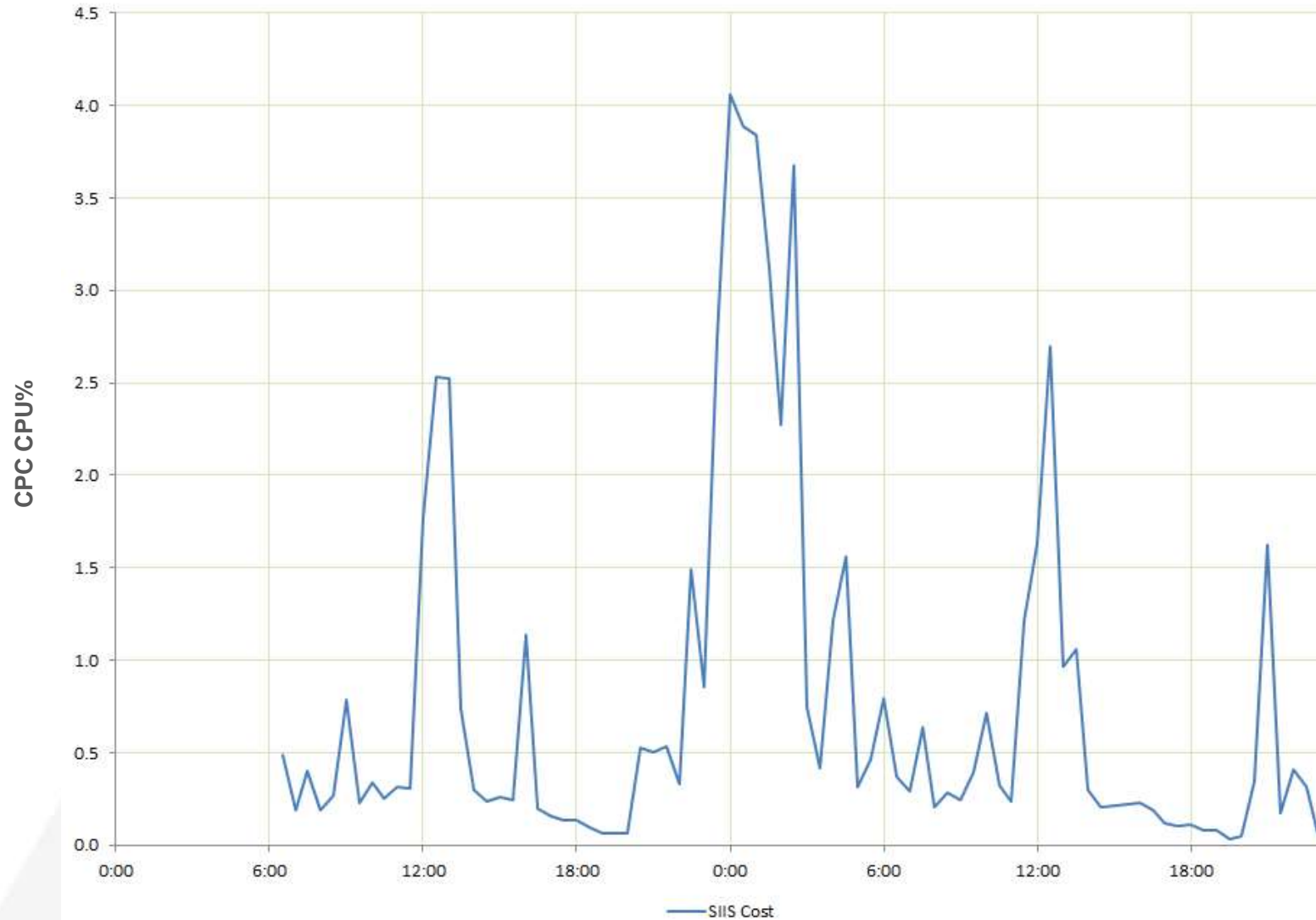
SMF 30 Instruction Counts – aka SMF30COUNT

z/OS 2.1 introduced the ability to collect instruction counts at the job/step level, meant to provide a more consistent metric of the work done compared to CPU time.

Optionally provides Instruction Counts in SMF 30 Counter Data Section

- Requires CPUMF basic counters in HIS be active
- New keyword SMF30COUNT|NOSMFCOUNT in SMFPRMxx

Not really designed to calculate the CPI – It doesn't provide the number of cycles – but less accurate CPI values can be calculated and used to detect large variations in programs' behavior.

SMF 30 Counter Data Section

8	8	SMF30_Inst_CP_Task	8	binary	The number of instructions executed while running on a standard processor as a task when it is not eligible for an offload processor and it is not associated with an enclave.
16	10	SMF30_Inst_CP_NonPreemptSRB	8	binary	The number of instructions executed while running on a standard processor as a non-preemptable SRB.
24	18	SMF30_Inst_CP_PreemptSRB	8	binary	The number of instructions executed while running on a standard processor as a preemptable or client SRB when it is not associated with an enclave.
32	20	SMF30_Inst_Offload	8	binary	The number of instructions executed while running on an offload processor when not associated with an enclave.
40	28	SMF30_Inst_OffloadOnCP	8	binary	The number of instructions executed while running on a standard processor as eligible for an offload processor while not associated with an enclave.
48	30	SMF30_Inst_CP_Enclave	8	binary	The number of instructions executed while running on a standard processor as not eligible for an offload processor for an independent enclave.
56	38	SMF30_Inst_Offload_Enclave	8	binary	The number of instructions executed while running on an offload processor for an independent enclave.
64	40	SMF30_Inst_OffloadOnCP_Enclave	8	binary	The number of instructions executed while running on a standard processor as eligible for an offload processor for an independent enclave.
72	48	SMF30_Inst_CP_DepEnc	8	binary	The number of instructions executed while running on a standard processor as not eligible for an offload processor for a dependent enclave.
80	50	SMF30_Inst_Offload_DepEnc	8	binary	The number of instructions executed while running on an offload processor for a dependent enclave.
88	58	SMF30_Inst_OffloadOnCP_DepEnc	8	binary	The number of instructions executed while running on a standard processor as eligible for an offload processor for a dependent enclave.

SMF 30 Instruction Counts – aka SMF30COUNT

When originally introduced, early use of the provided metrics showed unexpected variability due to impact of system related activity like interrupts on recorded values.

For this reason for some time IBM recommended not using them.

The recommendation recently changed to "Use them with a grain of salt: Make sure you identify and discard measurements affected by the above issue."

But how do I do it?

Our experience with SMF30COUNT

- Asked the customer to activate the recording

- Collected one week of SMF 30 data in conjunction with SMF 7x and SMF 113

- Confirmed the SIIS pattern shown above

- Analyzed SMF 30 Instruction Counters for a large number or individual program executions, taking into account average and max CPI

The following few charts report about three different programs showing very different behaviors.

Calculating the CPI by executed program

```
 1 SELECT SMFXDTE, SMFXTME, SMF30INSTFLGS1MRS,
 2        SMF30PGM, SMF30CPT,
 3       (SMF30_INST_CP_TASK + SMF30_INST_CP_ENCLAVE + SMF30_INST_CP_DEPENC + SMF30_INST_CP_PREEMPTSRB +
 4        SMF30_INST_OFFLOADONCP + SMF30_INST_OFFLOADONCP_ENCL + SMF30_INST_OFFLOADONCP_DEPENC) as SMF30CPT_INST,
 5
 6       (SMF30CPT * 50000000.0) /   -- Note: SMF30CPT * 50000000 gives the approximate number of cycles (z13)
 7       (SMF30_INST_CP_TASK + SMF30_INST_CP_NONPREEMPTSRB + SMF30_INST_CP_PREEMPTSRB +
 8        SMF30_INST_CP_ENCLAVE + SMF30_INST_CP_DEPENC +
 9        SMF30_INST_OFFLOADONCP + SMF30_INST_OFFLOADONCP_ENCL + SMF30_INST_OFFLOADONCP_DEPENC)  as SMF30CPT_CPI,
10
11        SMF30_INST_CP_TASK,
12        SMF30_INST_CP_NONPREEMPTSRB
13        SMF30_INST_CP_ENCLAVE,
14        SMF30_INST_CP_DEPENC,
15        SMF30_INST_CP_PREEMPTSRB,
16        SMF30_INST_OFFLOADONCP,
17        SMF30_INST_OFFLOADONCP_ENCL,
18        SMF30_INST_OFFLOADONCP_DEPENC
19
20    FROM SMF304CDS
21  WHERE SMF30INSTFLGS1MRS = 0
```

Case #1 – Stable user program (19K Observations)
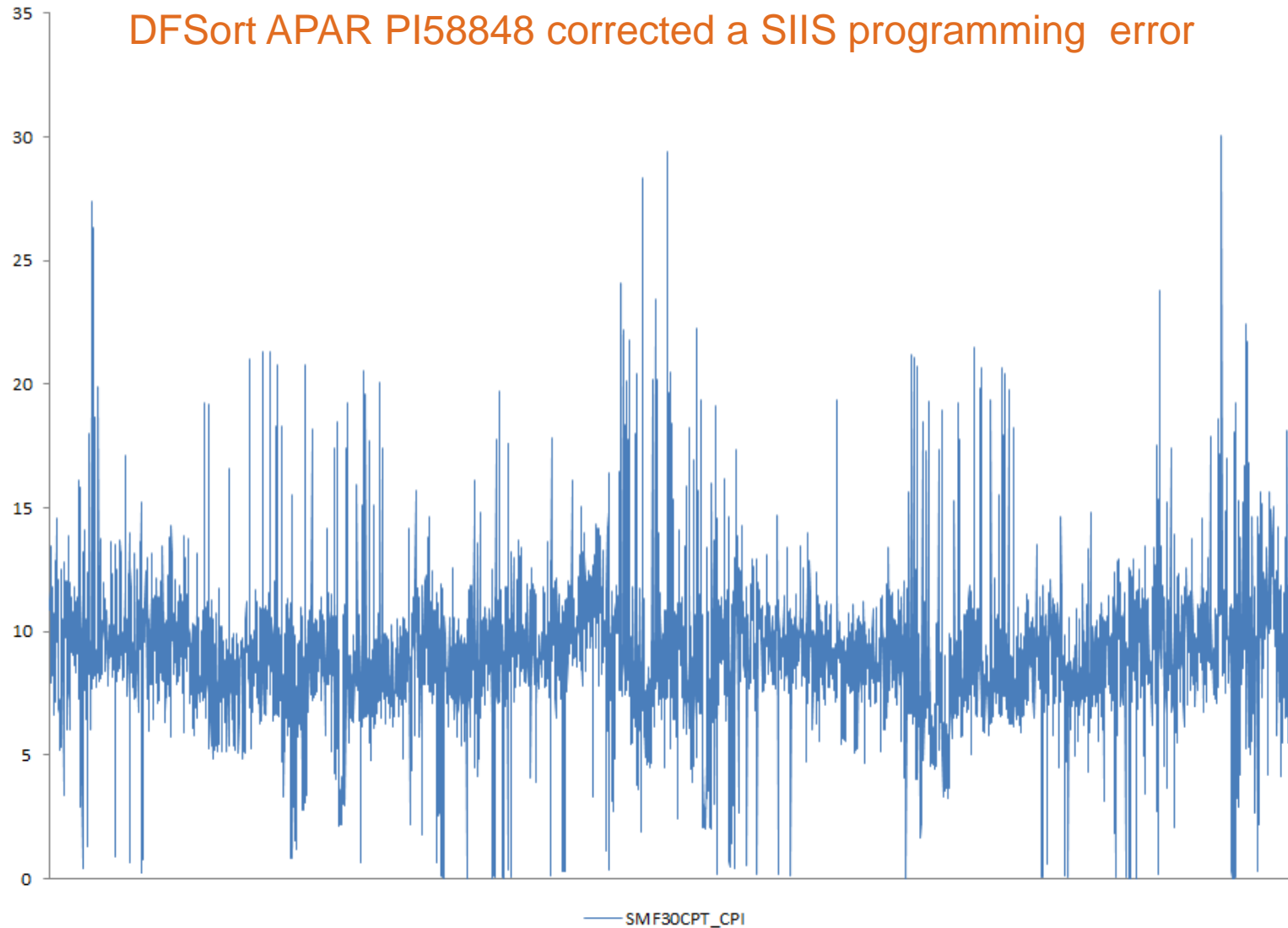

SMF30CPT_CPI
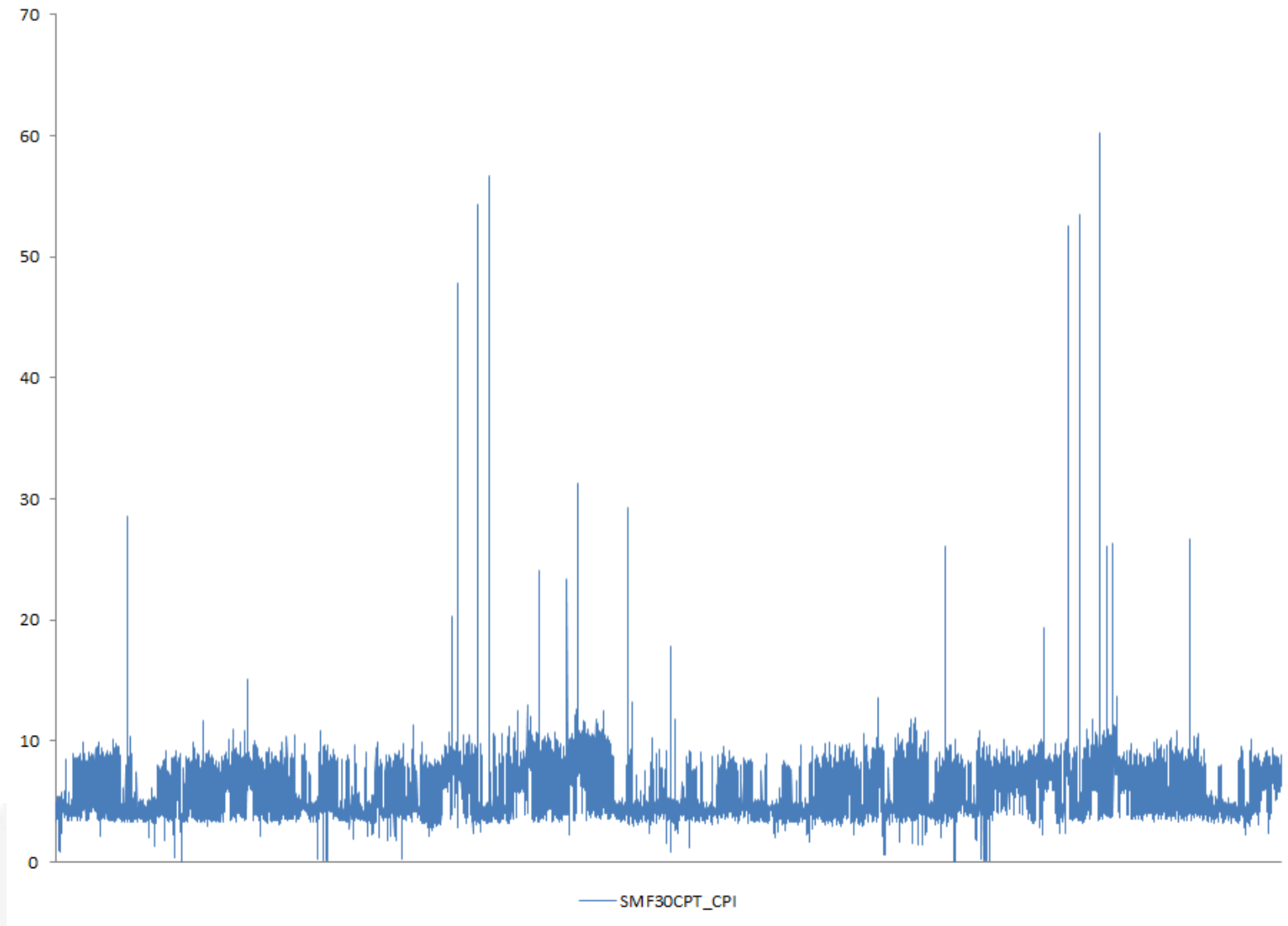
Case #2 – User program probably affected by SIIS (700 observations)
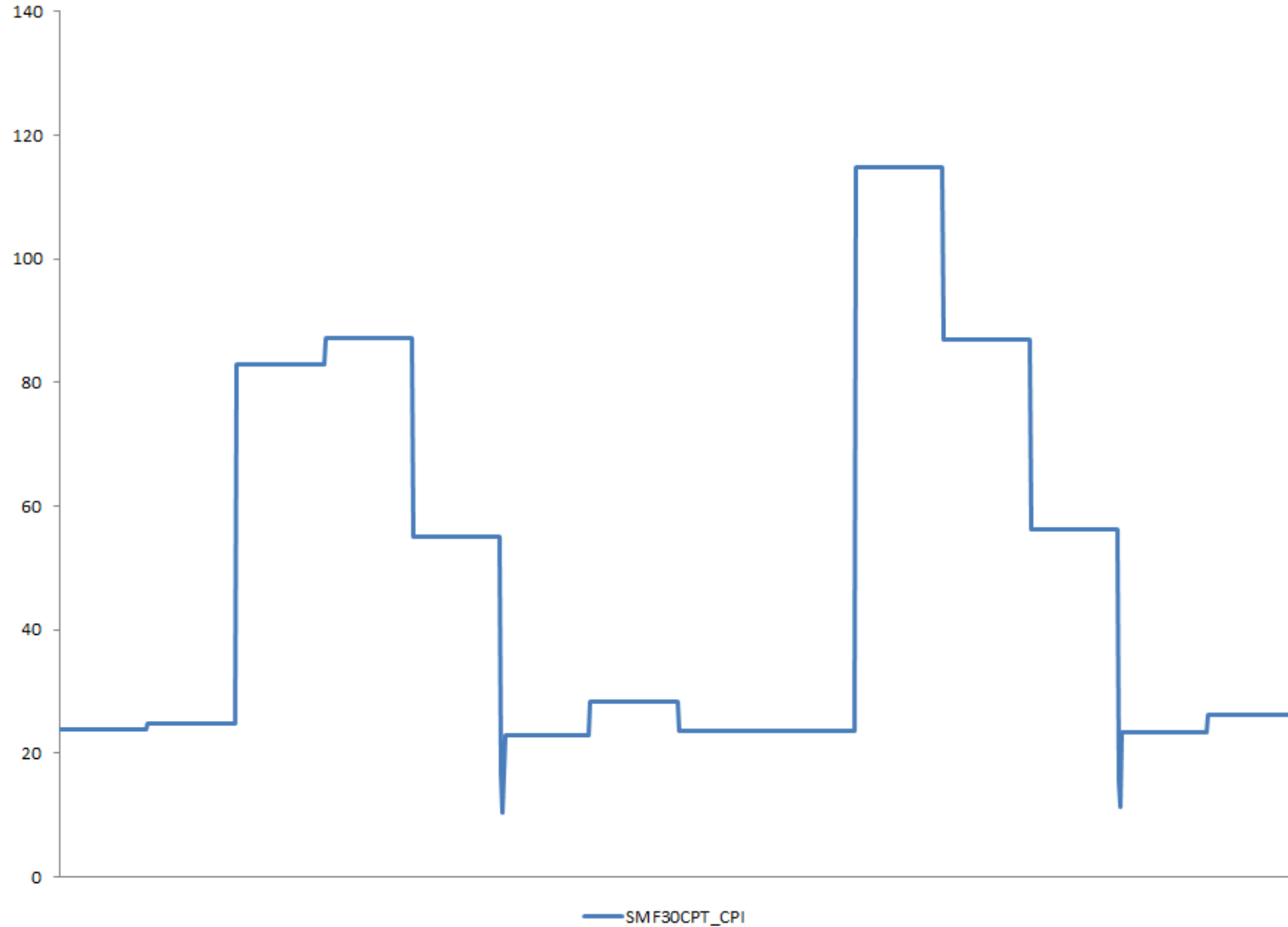
Case #3 – DFSort (5500 observations).

DFSort APAR PI58848 corrected a SIIS programming error

SMF30CPT_CPI

Summary

- Store Into the Instruction Stream (SIIS) can impact your system efficiency.

- Hardware Instrumentation (SMF 113) can help detect it. Enable it everywhere!

- Although designed for different scopes SMF 30 Instruction Counters can be used to identify programs affected by SIIS activity.

- In our experience SMF 30 Instruction Counters proved very useful in spotting unusual behaviors. The margin of error in the measurements was low and the outliers easy to identify.

- For these reasons we recommend enabling the SMF30COUNT setting.

Working with RAW SMF data in a workstation environment

Our SMF processing flow

SMF data come in, normally in TERSE format
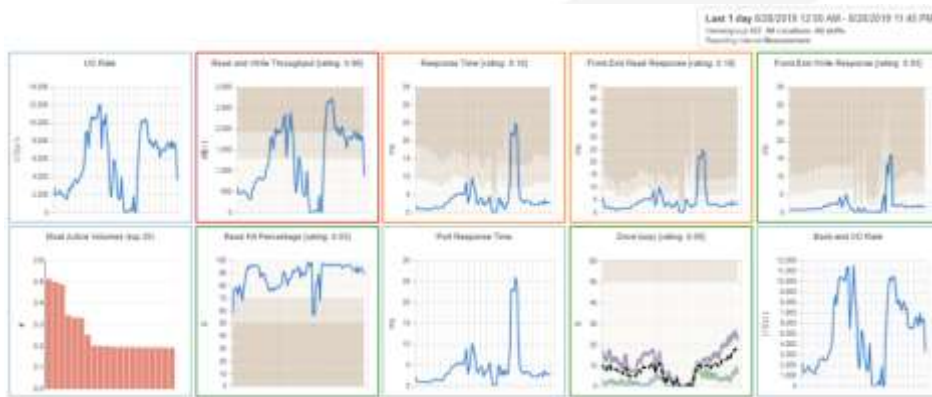
We upload it to our z/OS System

Here data is untersed, checked for validity, prepared for processing with IM Vision, run some reduction tool.

Data is downloaded to workstation for additional processing

Workstation tools we use for SMF include IM Vision, EasySMF, homegrown..

Our SMF processing needs

- Most of our SMF data processing tools run or may run in our workstations.

- Unterse, validity check and preparation for Intellimagic Vision (RMFPACK) must be performed under z/OS. This requires lengthy uploads and downloads of large amounts of data.

- To make the process more efficient we worked to see if was possible to perform these three steps in a workstation environment. To do so:

  - We created an unterse program able to properly manage VBS records.
  - We created a SMF summary report generator like the one we made available on z/OS[1].
  - In cooperation with Intellimagic we created a Java version of their RMFPACK tool.
  - We started exploring using Java to process SMF data.

[1] The z/OS version of SMFSCRUB is available at https://watsonwalker.com/software/free-tools/

Untersing SMF data on my workstation

```
mario@ubuntu > ls -l SYSA.SMFALL.TRS
-rw-r--r-- 1 mario_bezzi mario_bezzi 291506176 Jan 21 16:23 SYSA.SMFALL.TRS
```

Untersing SMF data on my workstation

```
mario@ubuntu > ls -l SYSA.SMFALL.TRS
-rw-r--r-- 1 mario_bezzi mario_bezzi 291506176 Jan 21 16:23 SYSA.SMFALL.TRS


mario@ubuntu > time unterse SYSA.SMFALL.TRS SYSA.SMFALL.SMF
```

Untersing SMF data on my workstation

```
mario@ubuntu > ls -l SYSA.SMFALL.TRS
-rw-r--r-- 1 mario_bezzi mario_bezzi 291506176 Jan 21 16:23 SYSA.SMFALL.TRS


mario@ubuntu > time unterse SYSA.SMFALL.TRS SYSA.SMFALL.SMF
** unterse   starting terse decode unpack v1.2.3  01:25:13 02/28/2020 **
** unterse   derived from github.com/openmainframeproject/tersedecompress **
** unterse   input  - filename: SYSA.SMFALL.TRS
** unterse   output - filename: SYSA.SMFALL.SMF
** unterse   blksize= 32760  lrecl=32767  packtype=PACK  recfm=VARIABLE
** unterse   input file size: 291505932 bytes output file size: 2052780682 bytes compression ratio:  14%
** unterse   terse complete decode unpack   01:25:55 02/28/2020 **
** unterse   return code: 0

real    0m42.491s
user    0m41.386s
sys     0m1.088s
```

Checking SMF data for validity

```
mario@ubuntu > time smfscrub -f SYSA.SMFALL.SMF
```

Checking SMF data for validity

```
mario@ubuntu > time smfscrub -f SYSA.SMFALL.SMF
SMFSCRUB v2.1 Processing started on Fri Feb 28 01:30:55 2020

SMF Basic Information:

  Type Sub  Sid      Records       Bytes  MinLen  AvgLen  MaxLen      Min DateTime          Max DateTime
     2   0  SYB0           1          18      18      18      18  2020-01-20 11:49:16  2020-01-20 11:49:16
     3   0  SYB0           1          18      18      18      18  2020-01-20 12:16:13  2020-01-20 12:16:13
   110   2  SYB0      377296   635966704     186    1685   32754  2019-11-03 22:22:02  2019-11-04 23:21:23
   110   2  SYB1      344381   411645644     186    1195   32756  2019-11-03 23:21:21  2019-11-04 23:13:00
   110   2  SYB2      337865   384920928     186    1139   32754  2019-11-03 23:59:37  2019-11-04 23:44:49
   110   2  SYB3      349876   620247370     186    1772   32754  2019-11-03 22:41:23  2019-11-04 23:47:11


SMFSCRUB - Processing ended on Fri Feb 28 01:31:03 2020

SMFSCRUB - Completion code is 0

real      0m7.936s
user      0m7.533s
sys       0m0.404s
```

Preparing for processing with IM Vision

```
mario@ubuntu > ls -l SYSA.SMFALL.*
-rw-r--r-- 1 mario_bezzi mario_bezzi 2052780682 Feb 28 01:25 SYSA.SMFALL.SMF
-rw-r--r-- 1 mario_bezzi mario_bezzi  291506176 Jan 21 16:23 SYSA.SMFALL.TRS
```

Preparing for processing with IM Vision

```
mario@ubuntu > ls -l SYSA.SMFALL.*
-rw-r--r-- 1 mario_bezzi mario_bezzi 2052780682 Feb 28 01:25 SYSA.SMFALL.SMF
-rw-r--r-- 1 mario_bezzi mario_bezzi  291506176 Jan 21 16:23 SYSA.SMFALL.TRS

mario@ubuntu > time java SMF2ZRF SYSA.SMFALL.SMF SYSA.SMFALL.ZRF
```

Preparing for processing with IM Vision

```
mario@ubuntu > ls -l SYSA.SMFALL.*
-rw-r--r-- 1 mario_bezzi mario_bezzi 2052780682 Feb 28 01:25 SYSA.SMFALL.SMF
-rw-r--r-- 1 mario_bezzi mario_bezzi  291506176 Jan 21 16:23 SYSA.SMFALL.TRS

mario@ubuntu > time java SMF2ZRF SYSA.SMFALL.SMF SYSA.SMFALL.ZRF
SMF2ZRF Copyright (c) 2020 IntelliMagic B.V.
  INFILE: SYSA.SMFALL.SMF succesfully opened for reading data set records
  OUTFILE: SYSA.SMFALL.ZRF succesfully opened for writing compressed data
  The SMF2ZRF compressor has been started. Version 1.0.0.
  Ready: 1409420 records processed from INFILE,
         1957.7 MB (2052780682 bytes) written to OUTFILE,
         as 187.5 MB (196560154 bytes) compressed data.
SMF2ZRF processing completed successfully

real    0m36.140s
user    0m28.417s
sys     0m7.002s
```

Preparing for processing with IM Vision

```
mario@ubuntu > ls -l SYSA.SMFALL.*
-rw-r--r-- 1 mario_bezzi mario_bezzi 2052780682 Feb 28 01:25 SYSA.SMFALL.SMF
-rw-r--r-- 1 mario_bezzi mario_bezzi  291506176 Jan 21 16:23 SYSA.SMFALL.TRS

mario@ubuntu > time java SMF2ZRF SYSA.SMFALL.SMF SYSA.SMFALL.ZRF
SMF2ZRF Copyright (c) 2020 IntelliMagic B.V.
  INFILE: SYSA.SMFALL.SMF succesfully opened for reading data set records
  OUTFILE: SYSA.SMFALL.ZRF succesfully opened for writing compressed data
  The SMF2ZRF compressor has been started. Version 1.0.0.
  Ready: 1409420 records processed from INFILE,
         1957.7 MB (2052780682 bytes) written to OUTFILE,
         as 187.5 MB (196560154 bytes) compressed data.
SMF2ZRF processing completed successfully

real     0m36.140s
user     0m28.417s
sys      0m7.002s

mario@ubuntu > ls -l SYSA.SMFALL.*
-rw-r--r-- 1 mario_bezzi mario_bezzi 2052780682 Feb 28 01:25 SYSA.SMFALL.SMF
-rw-r--r-- 1 mario_bezzi mario_bezzi  291506176 Jan 21 16:23 SYSA.SMFALL.TRS
-rw-r--r-- 1 mario_bezzi mario_bezzi  196560154 Feb 28 01:46 SYSA.SMFALL.ZRF
```

The IBM Record Generator

The IBM Record Generator for Java

- Provides a stand-alone utility to generate Java helper classes based on the associated-data (ADATA) files produced from compiling COBOL copybooks or assembler DSECTs.

- These Java helper classes can then be used in Java applications to access data from the COBOL or assembler language-specific record structures.

- The Record Generator for Java V3.0 is freely available to download for all users of z/OS V2.1, or later. License redistribution rights are provided for deployment to non-z/OS platforms to support distributed development and build.

Home Page https://www.ibm.com/support/knowledgecenter/SSMQ4D_3.0.0/documentation/welcome.html

Announcement Letter https://www-01.ibm.com/common/ssi/cgi-in/ssialias?infotype=an&subtype=ca&supplier=897&letternum=ENUS217-295

EasySMF:JE

- For our SMF processing needs we also use EasySMF by Black Hill Software.

- EasySMF offers an optional component called EasySMF:JE, which makes processing SMF in Java very easy. As part of EasySMF:JE Andrew Rowling ships ready to use helper classes for most popular SMF records.

- EasySMF:JE helper classes, which are custom built around SMF, provide more sophisticated data access methods than those created by the IBM Record Generator, which is a general purpose ADATA translator.

The IBM Record Generator for Java
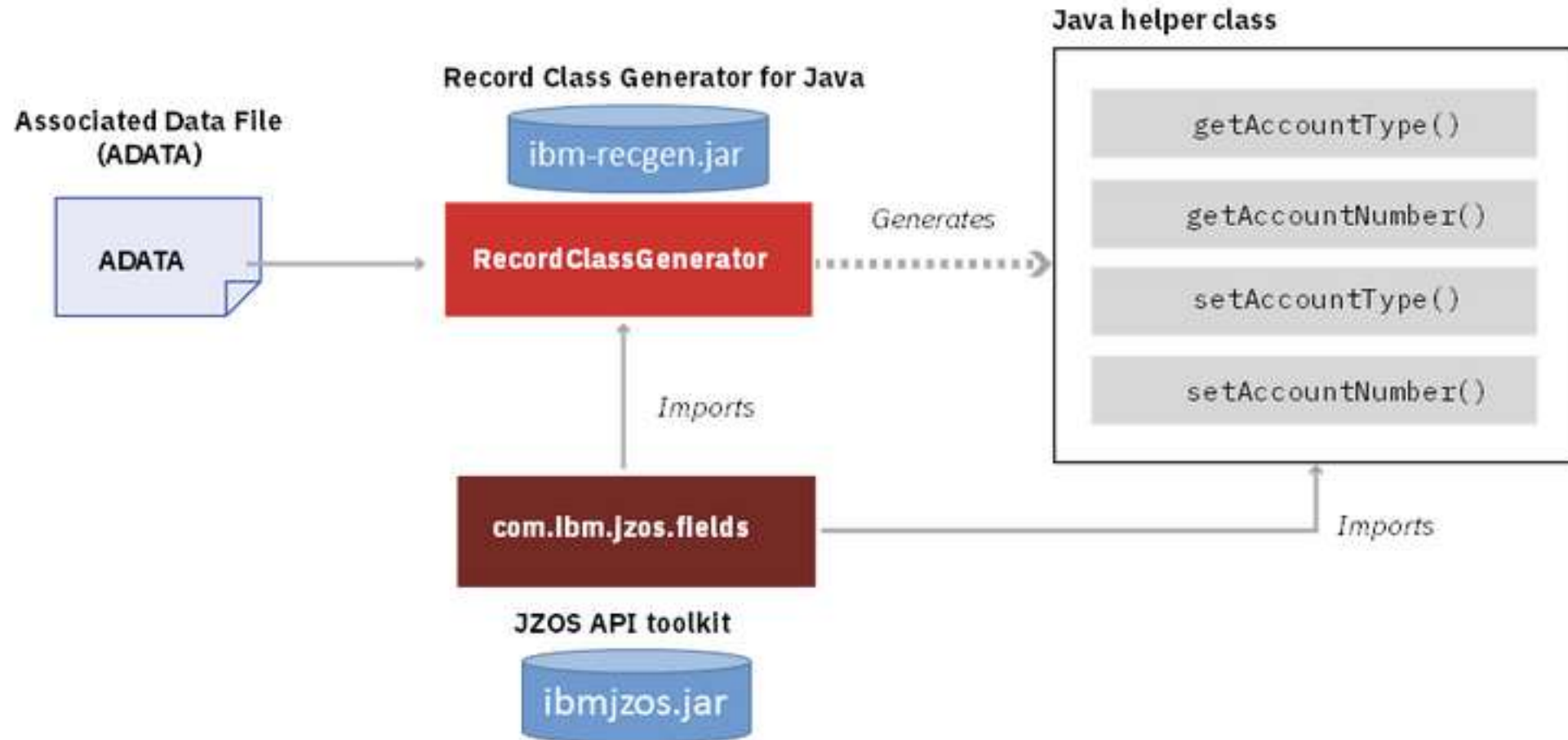
The IBM Record Generator for Java

- Download the ibmjzos-2.4.4 or later jar file from your z/OS system – in our system it is located in /usr/lpp/java/J8.0_64/lib/ext

- Download the ibm-recgen.jar from: https://developer.ibm.com/mainframe/products/record-generator-for-java/

- Add both to your Java class path.

- On z/OS Run Assembler to generate the ASMADATA you need and download it to your workstation.

- Run the IBM Record Generator to create the Java helper classes you need.

The IBM Record Generator for Java


```
mario@ubuntu > ls -l
total 7880
-rw-r--r-- 1 mario_bezzi mario_bezzi  468274 Jan 11 17:20  asmadata.rawout
drwxr-xr-x 3 mario_bezzi mario_bezzi    4096 Apr  2  2019  com
-rw-rw-r-- 1 mario_bezzi mario_bezzi   78004 Feb 26  2019  ENUS217-295.PDF
-rw-r--r-- 1 mario_bezzi mario_bezzi  565660 Feb 26  2019  ibmjzos.jar
-rw-r--r-- 1 mario_bezzi mario_bezzi  135659 Feb 27  2018  ibm-recgen.jar
-rw-rw-r-- 1 mario_bezzi mario_bezzi  371662 Feb 26  2019 'IBM Record Generator for Java English 2017.09.29 ibm-recgen.zip'
-rw-rw-r-- 1 mario_bezzi mario_bezzi  417914 Feb 26  2019 'IBM Record Generator for Java.pdf'
-rw-r--r-- 1 mario_bezzi mario_bezzi 4823657 Jan  3 16:23 'IBM Record Generator for Java V3.0 PDFs.zip'
-rw-rw-r-- 1 mario_bezzi mario_bezzi  249707 Feb 26  2019  jzos_samples.zip
-rw-rw-r-- 1 mario_bezzi mario_bezzi  900408 Feb 26  2019  jzos_users_guide_v8.pdf
```

All we really need are the output from ASSEMBLER, and the two jar files mentioned above

The IBM Record Generator for Java

```
mario@ubuntu > time java \
>       -cp ./ibm-recgen.jar:./ibmjzos.jar com.ibm.recordgen.asm.RecordClassGenerator    \
>       outputdir=./ gencache=false gensetters=false bufoffset=true                       \
>       adatafile=asmadata.rawout section=SMF71HDR                                         \
>       package=com.ibm.rmfrec71 class=JZOSSMF71HdrWrapper
```

The IBM Record Generator for Java

```
mario@ubuntu > time java \
>       -cp ./ibm-recgen.jar:../ibmjzos.jar com.ibm.recordgen.asm.RecordClassGenerator    \
>       outputdir=./ gencache=false gensetters=false bufoffset=true                        \
>       adatafile=asmadata.rawout section=SMF71HDR                                          \
>       package=com.ibm.rmfrec71 class=JZOSSMF71HdrWrapper

IBM Record Generator for Java V3.0.1 Build 20180227-1206 on: Thu Feb 27 23:58:28 CET 2020
Generating asmadata.rawout to: com/ibm/rmfrec71/JZOSSMF71HdrWrapper.java

real     0m0.112s
user     0m0.162s
sys      0m0.022s
```

The IBM Record Generator for Java

```
mario@ubuntu > time java \
>       -cp ./ibm-recgen.jar:../ibmjzos.jar com.ibm.recordgen.asm.RecordClassGenerator    \
>       outputdir=./ gencache=false gensetters=false bufoffset=true                        \
>       adatafile=asmadata.rawout section=SMF71HDR                                          \
>       package=com.ibm.rmfrec71 class=JZOSSMF71HdrWrapper

IBM Record Generator for Java V3.0.1 Build 20180227-1206 on: Thu Feb 27 23:58:28 CET 2020
Generating asmadata.rawout to: com/ibm/rmfrec71/JZOSSMF71HdrWrapper.java

real    0m0.112s
user    0m0.162s
sys     0m0.022s


mario@ubuntu > ls -l com/ibm/rmfrec71/JZOSSMF71HdrWrapper.java
-rw-r--r-- 1 mario_bezzi mario_bezzi 824 Feb 27 23:58 com/ibm/rmfrec71/JZOSSMF71HdrWrapper.java
```

A simple helper class


```java
JZOSSMF71HdrWrapper.java

 1  import com.ibm.jzos.fields.*;
 2
 3  // Generated by IBM Record Generator for Java V3.0.1 Build 20180227-1206 on: Fri Jan 03 16:24:42 CET 2020
 4
 5  public class JZOSSMF71HdrWrapper {
 6      protected static AssemblerDatatypeFactory factory = new AssemblerDatatypeFactory();
 7      static { factory.setStringTrimDefault(false); }
 8
 9      /** <pre>
10      SMF71HDR  DSECT </pre> */
11      public static final int SMF71HDR = factory.getOffset();
12
13      /** <pre>
14      SMF71HDC  DS     CL36       COMPLETE HEADER+PRODUCT SECTION TRIPL &#x0040;WA05544
15                ORG    SMF71HDC                                        &#x0040;WA05544 </pre> */
16      protected static final StringField SMF71HDC = factory.getStringField(36);
17
18      /** <pre>
19      SMF71LEN  DS     BL2        RECORD LENGTH </pre> */
20      static { factory.incrementOffset(-36); }
21      protected static final BinaryAsIntField SMF71LEN = factory.getBinaryAsIntField(2, false);
22
23      /** <pre>
24      SMF71SEG  DS     BL2        SEGMENT DESCRIPTOR </pre> */
25      protected static final BinaryAsIntField SMF71SEG = factory.getBinaryAsIntField(2, false);
26
27      /** <pre>
28      SMF71FLG  DS     BL1        HEADER FLAG BYTE </pre> */
29      protected static final BinaryAsIntField SMF71FLG = factory.getBinaryAsIntField(1, false);
```

Processing SMF record with Java

JZOSSMF71HdrWrapper.java   ReadSMFFile.java ⌗

```java
 1 import java.io.*;
 2
 3 public class ReadSMFFile {
 4
 5     private static final int recbuf_SIZE = 32768;
 6
 7     public static void main(String[] args) throws IOException {

20         SMFReader MySMFReader = new SMFReader();
21         MySMFReader.Open(InputFile, recbuf_SIZE);
22
23         System.out.format(headingFormat, "Smf71Snm", "Smf71Dte", "Smf71Tme", "Smf71Dat", "Smf71Ist", "Smf71Int");
24
25         while ((reclen = MySMFReader.GetSMFRecord(recbuf)) != -1) {
26             reccount += 1; totlen += reclen;
27
28             JZOSSMF71HdrWrapper r71Hdr = new JZOSSMF71HdrWrapper(recbuf, 0);
29             if ((r71Hdr.getSmf71Rty() == 71) & (r71Hdr.getSmf71Prn() > 0)) {
30                 JZOSSMF71ProWrapper r71Pro = new JZOSSMF71ProWrapper(recbuf, (int) r71Hdr.getSmf71Prs());
31                 System.out.format(outputFormat,
32                         r71Pro.getSmf71Snm(),
33                         SMFFields.smfjdate(r71Hdr.getSmf71Dte()),
34                         SMFFields.smftime(r71Hdr.getSmf71Tme()),
35                         SMFFields.smfjdate(r71Pro.getSmf71Dat()),
36                         SMFFields.rmfist(r71Pro.getSmf71Ist()),
37                         SMFFields.rmfint(r71Pro.getSmf71Int()));
38             }
39         }
40         System.out.println("Read " + reccount + " records, " + totlen + " bytes.");
```

Processing SMF record with Java


```
Console ⊠    JZOSSMF71HdrWrapper.java      ReadSMFFile.java

<terminated> ReadSMFFile [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (Feb 28, 2020, 12:25:05 AM)
Smf71Snm Smf71Dte Smf71Tme Smf71Dat Smf71Ist Smf71Int
SYSA 2019/10/02 12:15:00.11 2019/10/02 12:00:00 899
SYSA 2019/10/02 12:30:00.09 2019/10/02 12:15:00 899
SYSA 2019/10/02 12:45:00.09 2019/10/02 12:30:00 900
SYSA 2019/10/02 13:00:00.08 2019/10/02 12:45:00 899
SYSA 2019/10/02 13:15:00.08 2019/10/02 13:00:00 900
SYSA 2019/10/02 13:30:00.09 2019/10/02 13:15:00 900
SYSA 2019/10/02 13:45:00.17 2019/10/02 13:30:00 900
SYSA 2019/10/02 14:00:00.15 2019/10/02 13:45:00 899
SYSA 2019/10/02 14:15:00.12 2019/10/02 14:00:00 899
SYSA 2019/10/02 14:30:00.26 2019/10/02 14:15:00 900
SYSA 2019/10/02 14:45:00.33 2019/10/02 14:30:00 900
SYSA 2019/10/02 15:00:00.13 2019/10/02 14:45:00 899
SYSA 2019/10/02 15:15:00.09 2019/10/02 15:00:00 899
SYSA 2019/10/02 15:30:00.07 2019/10/02 15:15:00 899
SYSA 2019/10/02 15:45:00.07 2019/10/02 15:30:00 900
SYSA 2019/10/02 16:00:00.13 2019/10/02 15:45:00 900
SYSA 2019/10/02 16:15:00.08 2019/10/02 16:00:00 899
SYSA 2019/10/02 16:30:00.07 2019/10/02 16:15:00 899
SYSA 2019/10/02 16:45:00.07 2019/10/02 16:30:00 900
```

Summary

- With a reasonable programming effort we were able to create workstation versions of programs to unterse, validity check and prepare SMF data for processing with Intellimagic Vision.

- In our limited, early tests, Java and the IBM Record Generator utility gave promising results for ad-hoc analysis of SMF metrics otherwise not available.

- These new tools significantly reduced time and effort we spend to processing large amounts of SMF data.

Miscellanea

Interesting sessions we attended in Fort Worth

IBM WSC Hot Topics by Kathy Walsh

https://events.share.org/Winter2020/Public/SessionDetails.aspx?&SessionID=10355

Understanding LPAR Controls for Better Performance by Kathy Walsh

https://events.share.org/Winter2020/Public/SessionDetails.aspx?&SessionID=10303

IBM z15 System Recovery Boost by Dave Surman

https://events.share.org/Winter2020/Public/SessionDetails.aspx?&SessionID=10239
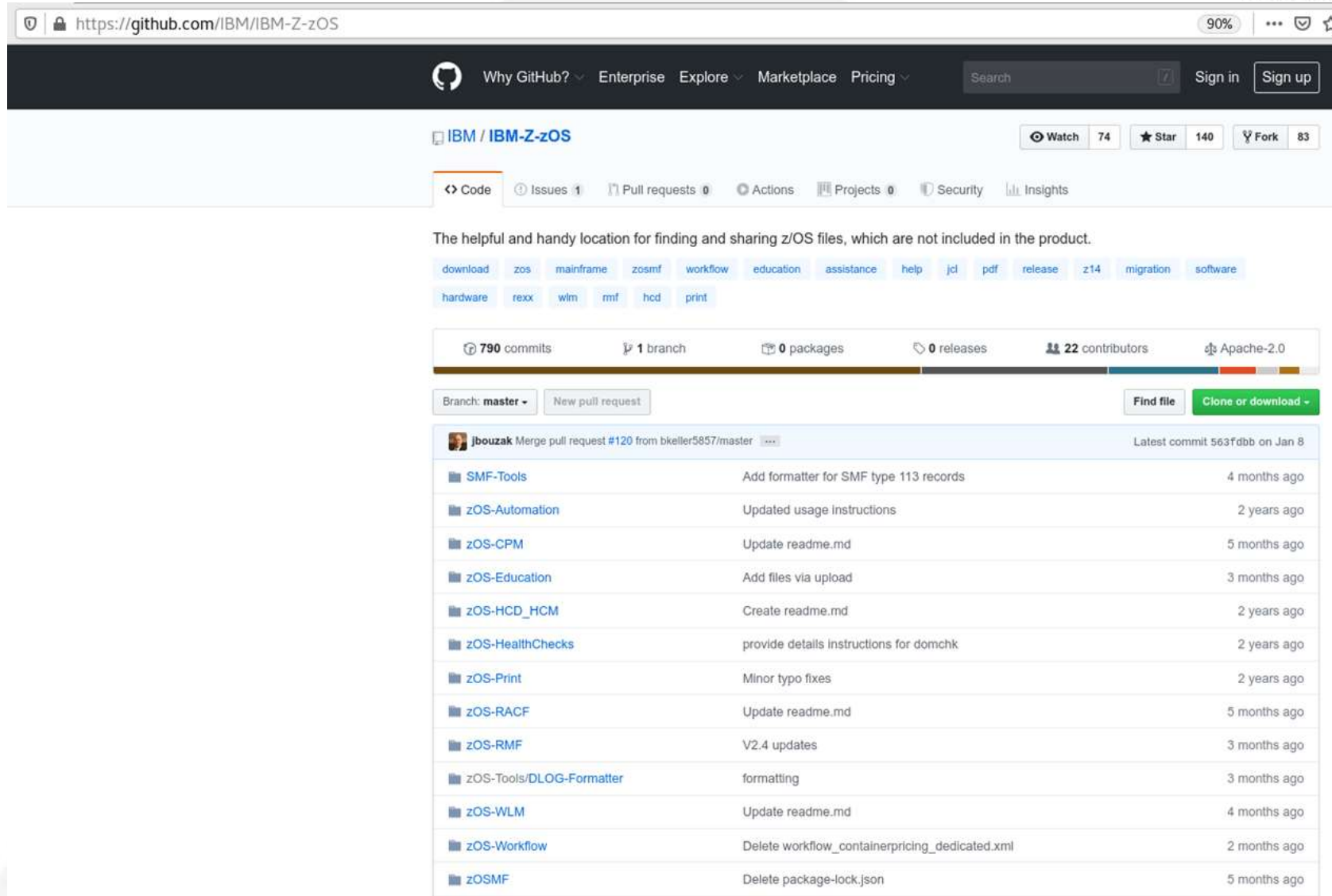
What's New in z/OS Communications Server? by Gus Kassimis and Sam Raynolds

https://events.share.org/Winter2020/Public/SessionDetails.aspx?&SessionID=9520

The Hitchhiker's Guide to MIPS and Capacity Planning by David Hutton

https://events.share.org/Winter2020/Public/SessionDetails.aspx?&SessionID=10277

z/OS GitHub

Educational Assistance Modules for z/OS

Frank

End of Support Quick Reference

-

Prod Name	Prod Number	Release	GA Date	EoS Date
z/OS	5650-ZOS	2.1	2013-09-30	2018-09-30
z/OS	5650-ZOS	2.2	2015-09-30	2020-09-30
z/OS	5650-ZOS	2.3	2017-09-30	
z/OS	5650-ZOS	2.4	2019-09-30	
CICS	5655-Y04	5.1	2012-12-14	2019-07-01
CICS	5655-Y04	5.2	2014-06-13	2020-12-31
CICS	5655-Y04	5.3	2015-12-11	
CICS	5655-Y04	5.4	2017-06-16	
CICS	5655-Y04	5.5	2018-12-14	
CICS	**5655-Y04**	**5.6**	**Open Beta**	
CICS	5655-S97	4.1	2009-06-26	2017-09-30
CICS	5655-S97	4.2	2011-06-24	2018-09-30
Db2	5605-DB2	10.1	2010-10-22	2017-09-30
Db2	5615-DB2	11.1	2013-10-25	2020-09-30
Db2	5650-DB2	12.1	2016-10-21	

End of Support Quick Reference

-

Prod Name	Prod Number	Release	GA Date	EoS Date
IMS	5635-A04	13.1	2013-10-25	2018-11-06
IMS	5635-A05	14.1	2015-10-30	2020-11-30
IMS	5635-A06	15.1	2017-10-27	
WebSphere MQ	5655-R36	7.1	2011-11-25	2017-11-06
MQ for z/OS	5655-W97	8.0	2014-06-13	**2020-04-30**
MQ for z/OS	5655-MQ9	9.0	2016-06-17	
MQ for z/OS	5655-MQ9	9.1	2018-07-03	
Enterprise COBOL	5655-S71	4.1	2007-12-14	2014-04-30
Enterprise COBOL	5655-S71	4.2	2009-08-28	2021-09-30
Enterprise COBOL	5655-W32	5.1	2013-06-21	**2020-04-30**
Enterprise COBOL	5655-W32	5.2	2015-02-27	**2020-04-30**
Enterprise COBOL	5655-EC6	6.1	2016-03-18	
Enterprise COBOL	5655-EC6	6.2	2017-09-08	
Enterprise COBOL	5655-EC6	6.3	2019-09-06	

User Key Common Storage Reminder

CICS/TS 5.2 is Minimum CICS level for z/OS 2.4

- z/OS 2.3 was the last z/OS release to support user key common storage.
  - It is being removed because it represents a security exposure.
- To provide *temporary* relief for customers that still have programs that rely on user key common storage, APAR OA56180 for z/OS 2.1/2.2/2.3 delivered a function called Restricted Use CSA (RUCSA).
  - This lets you control who can use user key CSA, but it doesn't eliminate the security concern.
  - RUCSA is enabled by adding RUCSA=(x,y) to your IEASYSxx member.
- To further encourage customers to completely eliminate the use of user key CSA, RUCSA is a <u>chargeable z/OS option in z/OS 2.4</u>.
  - This means that you must enable it in your IFAPRDxx member.
    - Because this is a chargeable feature, you must inform IBM that you plan to use it. This should result in your ServerPac order including the appropriate definitions in the IFAPRDxx member that is supplied as part of your ServerPac order.
  - If your z/OS 2.4 IEASYSxx specifies RUCSA, but your IFAPRDxx does not, your system will wait state when you IPL.
- See TL article *Bye Bye User Key Common Storage* for more info about RUCSA.

zEDC Changes on z15

- On z15, the zEDC PCIE card is replaced by the on-chip zEDC Accelerator.
  - Unlike the zEDC PCIE cards, there is no charge for the accelerator.
  - However, there IS still a zEDC chargeable feature on z/OS.
- On z15, the zEDC compression/decompression function can be called Synchronously or Asynchronously – this differentiation is new with z15.
  - Synchronous callers do NOT require the chargeable z/OS feature. So no changes are required to IFAPRDxx.
  - Asynchronous requests are handled by a SAP processor. These DO require that the IFAPRDxx member is updated to reflect that you are using the chargeable feature.
- To our knowledge, the only IBM exploiters of the Synchronous interface are zlib, MQ, and Connect Direct. All other IBM exploiters (BSAM/QSAM, HSM, DSS, SMF, zFS, etc) use the asynchronous interface and therefore DO require the chargeable feature.
- If you use ISV products that exploit zEDC, consult your vendor to see which flavor they use.

ERV Refresher

- The ENQUEUE Residence Value (ERV) specifies how many service units a unit of work should receive a temporary dispatching priority boost if it is holding an ENQ that is being requested by a higher importance unit of work.
  - This is intended to improve availability of high importance work.
  - It does NOT remove the need for you to identify and address the causes of high importance work being delayed by ENQ contention with lower importance work.
- The default ERV value is 500 service units. The IBM recommendation is to set this to 50,000 service units <u>AND to address the cause of the contention</u>.
- You control the length of the temporary boost using the ERV parm, but you do NOT have any control over the target dispatching priority.
- The system dynamically identifies a DP that should place the unit of work that is holding the ENQ above about 40% of the work in the system.
  - If you have an environment where > 55% of the system capacity is being used by high importance work, this could result in the lower importance work impacting the capacity that is available to your high importance work.
  - The 'PROMOTED' section of the RMF Workload Activity report shows the CPU seconds used by promoted work units. If this is high, ADDRESS THE ROOT CAUSE of the contention.
- Thanks to **Kathy Walsh** for providing this info. See *User Experience* in latest TL for more info.

CPENABLE Changes

- The CPENABLE parm in IEAOPTxx specifies the low and high thresholds that should be used to determine whether to enable more CPs to process I/O interrupts.

- The *default* values are (0,0), meaning that all CPs should be enabled for I/O interrupts.
  - However for the longest time, the *recommended* values have been (10,30).

- Due to the speed of recent CPCs, and to maintain a balance between responsiveness and throughput, the WSC is changing the recommended values for z14 and z15 to be (5,15).

- WSC FLASH10337 (currently being updated) provides guidance for CPCs all the way back to z890 and will soon contain the new recommendations.

- For much more information on this topic, as well as tips about fine-tuning, refer to **Kathy Walsh's** *Understanding LPAR Controls for Better Performance* session 27021.
  - Also see related APAR OA57270.

CICS Liberty High CPU APAR

- CICS APAR PH20006 (still **OPEN**) addresses a situation whereby CICS automatically enabled some new tracing options provided in WebSphere Liberty
  - *"After migrating to Liberty 19.0.0.3 CICS experiences an increase in CPU consumption. Health Center data shows that method org.eclipse.osgi.internal.log.LoggerImpl.getLocation() is consuming a large **(50%+)** amount of the CPU cycles of those sampled."*
  - CICS was moved to Liberty 19.0.0.3 by APAR PH09600, delivered in July/Sept 2019.

- The APAR text provides a temporary workaround (manual update to server.xml file) until the CICS PTF is provided.

- Thanks to **Dave Follis** for help with this.

zIIPs, HiperDispatch, and Db2

- In our customer performance reviews, we frequently find very finely-tuned LPAR CP weights like 346, 297, 275, 32, 27, and 23. And the same LPARs will have zIIP weights of 10, 10, 10, 10, 10, and 10!
  - We *rarely* see Vertical High zIIPs.
- However, in every release, Db2 is making more and more of its code zIIP-eligible.
- The combination of lots of zIIP-eligible Db2 work and no VH zIIP engines is becoming a more common cause of Db2 performance issues:
  - The Managing Db2 Performance book explicitly calls out (in an 'Important' box!) the importance of having VH zIIPs available to Db2 if the performance/availability of that Db2 is important.
  - To reinforce this message, a new Health Check (OA58654) is on the way that will complain if it finds that the LPAR does not have at least one VH zIIP.
- The message – zIIPs ARE CHEAP, this is *not* the place to scrimp!
- For more info, refer to Db2 Performance Redbook and *User Experience* in TL 19-4.

'Free' zIIPs, zPDT, and z/OSMF

For ISVs

- Starting with zPDT GA8, zPDT provides the ability to have free zIIP engines, in addition to whatever number of general purpose CPs you are licensed for (limited by the number of cores in your PC).

- Depending on the underlying **PC**, a fully-configured z/OSMF can take > 20 minutes to start.

- So ~~we~~ I decided that adding a zIIP might help, but couldn't hurt.

- WRONG! The startup time increased from 25 minutes to over 1 hour!

- Why? The WLM policy that is provided with ADCD defaults the IZUSVR1 address space to a Discretionary Service Class. zIIP processors will not issue a Needs Help signal for discretionary work. So our 3 CPs were twiddling their thumbs, while our 1 zIIP ran at a solid 100%, processing nearly all the work that was previously handled by our 3 CPs.

- The lesson – zPDT free zIIPs are fine, but beware the use of discretionary service classes for zIIP-eligible work.

FICON End-to-End Encryption

- The IBM z15 announcement included a Statement of Direction for End-to-End encryption support on FICON channels.
- That support is available now (as of Feb 27):
  - Formally announced in IBM announcement letter 120-013 on January 14, 2020.
  - Only available on z15
    - Requires Endpoint Security Enablement (feature code 1146), and CPACF enablement (FC3863)
  - Requires the latest FICON Express 16SA+ adapters for encryption and authentication
    - Although authentication will be supported on the FICON Express 16S+ adapters (introduced with z14 CPC).
  - The only supported DASD are the recently-announced IBM DS8900F.
  - Supported in z/OS 2.2/2.3/2.4 with APAR OA56294.
- Note that zHyperLinks do NOT support End-to-End Encryption.

DFSMSdss Tips

- DFSMSdss standalone restore does NOT support restore from full volume dumps created with COMPRESS, HWCOMPRESS, or ZCOMPRESS options.
    - This is currently not documented, but OA58859 will update DSS manual with this information.

- In z/OS 1.12, DFSMSdss made changes to improve the efficiency of logical data set dumps to tape. However, those changes did not apply to Extended Format data sets – this results in those data sets having 5x as many write I/Os as non-EF data sets.
    - New Function APAR OA58019 extends this enhancement to **Version 2** format single-stripe Extended Format data sets.

- Thanks to **Rob Gensler** for this information. If you are a DFSMSdss user, check out Rob's Session 26800 for this and other valuable information.

- And ALWAYS check out **Barb McDonald's** information-packed "*What's New in DFSMS*" session for a list of the latest enhancements to DFSMS.

zWIC and zWIN

- IBM Announcement Letter 220-032 announced a new <u>chargeable z/OS feature</u> called z Workload Interaction Correlator.
- This product is based on the existing High Frequency Throughput Statistics (SMF 98) function, but provides an infrastructure that can be used by other z/OS components and other products.
  - The only currently-announced z/OS exploiter is Supervisor – that is the support that was delivered in z/OS 2.2. We assume that other z/OS components will be added in the future.
- Information is gathered at 5-second intervals, providing visibility of problems or relationships that disappear in a typical 15-minute SMF/RMF interval.
- Its GA date is March 20, 2020.
  - The z/OS function will be available for z/OS 2.3 and 2.4 via APAR OA56175.
    - It will only be available on z14 or later.
    - The Correlator is a chargeable z/OS feature, so you will need to discuss with IBM and presumably update your IFAPRDxx member.
  - IMS Support will be shipped by APAR PH15072.
  - There will also be CICS support, but we don't have the APAR number yet.

Recommended HFTSINTVL is 5 Seconds

zWIC and zWIN

- zWIC provides the data gathering infrastructure. The analysis and presentation of that information will be provided by another (chargeable (IPLA)) product called z Workload Interaction Navigator (zWIN) – 5698-WKN.
  - Announced on Feb 25, 2020 in IBM US Announcement Letter 220-100.
- **zWIN requires Zowe**.
- Also planned to be available on March 20, 2020.

- Existing HFTS SMF Records have already proved valuable in investigating issues that disappear in a normal 15-minute SMF interval.

Quick Links to z/OS Announcement Letters

- z/OS 2.3 Announcement, July 17, 2017.
- z/OS 2.3 Enhancements, November 21, 2017.
- z/OS 2.3 Enhancements, March 6, 2018.
- z/OS 2.3 Enhancements, May 15, 2018.
- z/OS 2.3 Enhancements, August 7, 2018.
- z/OS 2.3 Enhancements, November 13, 2018.
- z/OS 2.4 Preview, February 26, 2019.
- z/OS 2.3 Enhancements, March 5, 2019.
- z/OS 2.3 Enhancements, April 23, 2019.
- z/OS 2.4 Announcement Letter, July 23, 2019.
- z/OS 2.4 Enhancements, December 10, 2019.

Thank You!

- If you have any questions, suggestions, comments, or general abuse, please email us at technical@watsonwalker.com

- Have a safe trip home, thank you for coming, and we hope to see you in Boston in August. Enjoy your summer!

- Please complete the online evaluation!

Thanks!!!