interskill learning

INTERSKILL MAINFRAME TRAINING NEWSLETTER

May 2014

Inside This Issue:

Welcome to the Interskill Mainframe Training Newsletter

Interskill Mainframe Training Newsletter is an e-Zine published by Interskill Learning, which provides world-class elearning Mainframe training programs for the Information Communications and Technology industry.

As a subscriber to Interskill Mainframe Training Newsletter you will receive an edition several times per year. These editions will include z/OS related articles for technical experts and management, and information on upcoming releases of Interskill's Mainframe curriculum.

Interskill Learning Planned New Curriculum and Updates for 2014

- **z/OS 2.1 Series**
  - z/OS Concepts and Components 2.1
  - Initializing and Terminating the z/OS System 2.1
  - Monitoring the z/OS System 2.1
  - z/OS Architecture 2.1
  - Identifying z/OS System Problems 2.1
  - Resolving z/OS System Problems 2.1

- **JCL 2.1 Series**
  - JCL (z/OS) – Introduction to z/OS JCL 2.1
  - JCL (z/OS) – Using Special Data Sets in Batch Jobs 2.1
  - JCL (z/OS) – Identifying and Resolving Batch Problems in JCL 2.1
  - JCL (z/OS) – Coding Procedures and JES2 Control Statements 2.1

- **IBM Environment Introduction Series 2.1**
  - Introduction to the IBM Enterprise Environment 2.1
  - z/OS System Programming Fundamentals 2.1
  - IBM Development Environment Overview 2.1

- **The z/OS Management Facility 2.1**
- **Introduction to zEnterprise - EC12, BC12 & zBX**
- **IBM Explorer for z/OS**

- **TSO/ISPF 2.1 Series**
  - Using Online System Facilities - TSO/ISPF 2.1
  - Managing Data Files and Definitions with ISPF/PDF 2.1
  - Maintaining Data in Files with the ISPF Editor 2.1

- **System Display and Search Facility (SDSF) 2.1 Series**
  - SDSF Concepts and Operation 2.1
  - Using SDSF to Control Job Processing 2.1
  - Using SDSF to Display, Manipulate and Print Job Output 2.1
  - Using SDSF to Manage System Resources and Devices 2.1

- **z/OS Console Simulators 2.1 Series**
  - Console Simulations 2.1

- **z/OS Batch Utilities 2.1 Series**
  - General Data Set Utilities 2.1
  - Data Utilities 2.1
  - Introduction to VSAM 2.1

- **JES2 2.1 Series**
  - JES2 System Initialization & Shutdown 2.1
  - Monitoring Batch Jobs with JES2 2.1
  - Using JES2 in Scheduling Batch Jobs 2.1
  - Identify and Resolve JES2 Batch Problems 2.1
  - Identify and Resolve JES2 System Problems 2.1

- **Operations Assessment Series 2.1**
  - CICS Concepts and Operation Assessment 2.1
  - JES2 Concepts and Operation Assessment 2.1
  - JES2 Problem Resolution Assessment 2.1
  - SNA/VTAM Concepts and Operation Assessment 2.1
  - z/OS Concepts Assessment 2.1
  - z/OS Problem Diagnosis and Resolution Assessment 2.1
  - z/OS System Operation Assessment 2.1
  - JCL Concepts Assessment 2.1
  - JCL Coding Assessment 2.1
  - JCL Problem Resolution Assessment 2.1
  - TSO/ISPF Concepts Assessment 2.1
  - TSO/ISPF Operation Assessment 2.1

- **Cloud Computing for Data Centers Series 2.1**
  - Cloud Computing for Data Centers 2.1

- **IBM Mainframe Communications Series 2.1**
  - IBM Mainframe Communications Concepts 2.1
  - VTAM Commands 2.1
  - Mainframe TCP/IP Commands 2.1

- **Data Center Storage Management Series**
  - Storage Fundamentals for Data Centers 2.1
  - Using DFSMS to Manage the z/OS Storage Environment 2.1
  - Storage Networks, Administration, and DASD Management Using ICKDSF 2.1

- **IBM i Programming Fundamentals Series**
  - CL Programming
  - CL Programming Functions and Messaging
  - RPG/400 Introduction
  - RPG/400 Coding
  - RPG/400 Programming
  - RPG/400 Workstation Programming

- **CA OPS/MVS® Event Management and Automation Series**
  - CA OPS/MVS® Event Management and Automation - Overview, Components, and Features
  - CA OPS/MVS® Event Management and Automation - Rules and OPS/REXX
  - CA OPS/MVS® Event Management and Automation - Automating Events Using the Relational Data Framework
  - CA OPS/MVS® Event Management and Automation - Automating Events Using the SSM
  - CA OPS/MVS® Event Management and Automation - Schedule and Group Managers for Event Management

- **CA 1® Tape Management Series**
  - CA 1® Tape Management - Using Tape Media
  - CA 1® Tape Management - Identifying and Resolving Media Problems

- **CA ACF2™ Series**
  - CA ACF2™ - Introduction
  - CA ACF2™ - Protecting Data Integrity
  - CA ACF2™ - Protecting System Access
  - CA ACF2™ - Defining Environment Controls
  - CA ACF2™ - Protecting General Resources
  - CA ACF2™ - Maintaining ACF2™
  - CA ACF2™ - For Auditors

Highlights of Interskill Learning Releases from 2013

Interskill Learning releases for 2013 included the following:

- **Business Continuity and Disaster Recovery Series**
  - Ensuring Data Centre Business Continuity

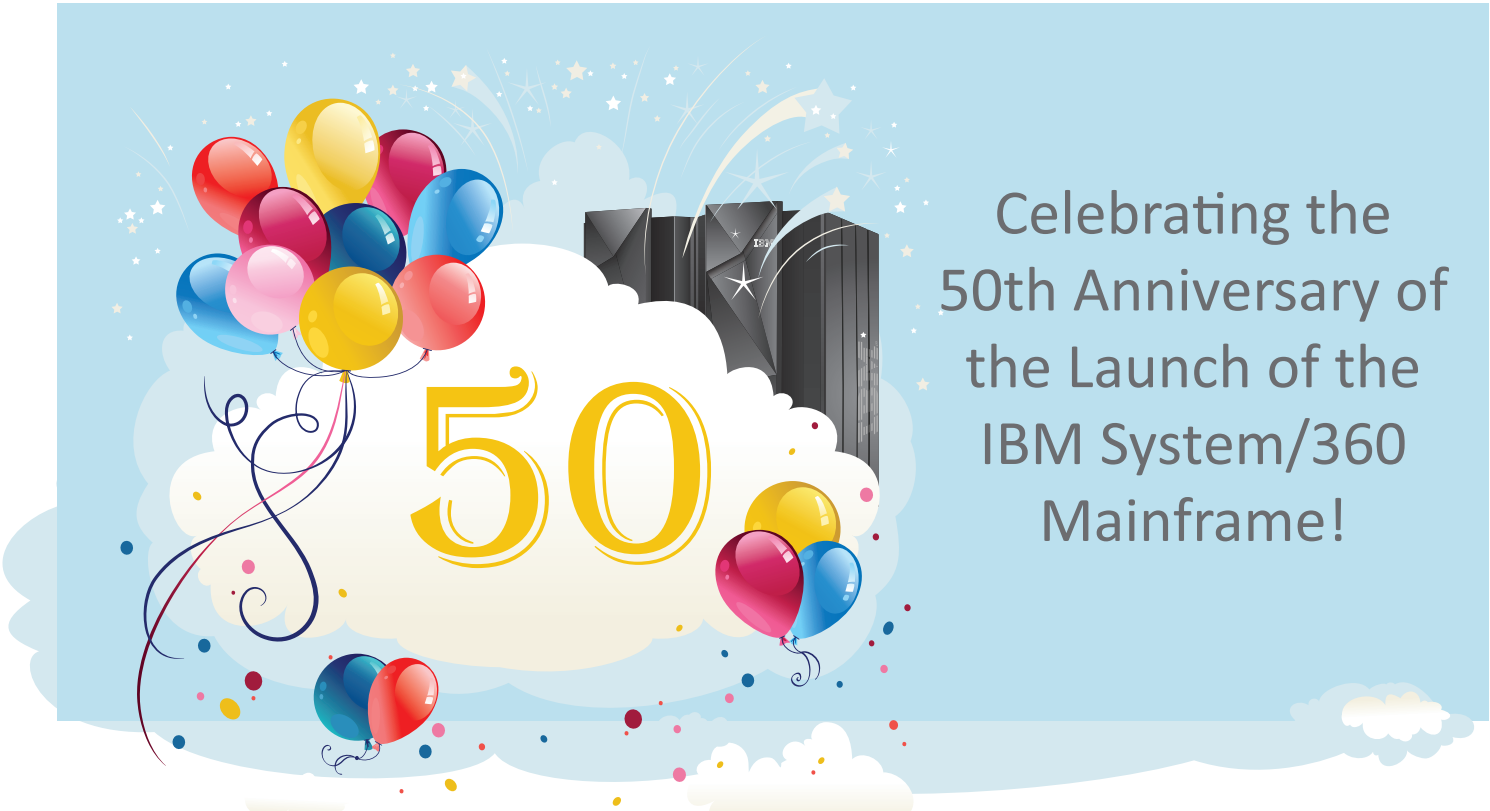
- **Installing and Managing z/OS Software Series**
  - Introduction to SMP/E

- **Data Center Storage Management Curriculum**
  - Storage Fundamentals for Data Centers
  - Managing the Storage Network
  - DFSMSrmm Operations

- **CA 7® Workload Automation Series**
  - Introduction to CA Workload Automation - CA 7® Edition
  - Scheduling Batch Processing
  - Monitoring and Managing the Batch Processing Environment
  - System Programmer Interaction with CA 7
  - Backup, Recovery, and Problem Resolution

- **CA 11 Workload Automation Series**
  - CA Workload Automation Restart Option for z/OS Schedulers Overview
  - Managing CA Workload Automation Restart Option for z/OS Schedulers

- **New courses in the Power series for IBM i System Administrators**
  - IBM i System Administration Fundamentals
  - Journal Management
  - Storage Management
  - Logical Partitioning and Virtualization
  - Security Implementation



Celebrating the 50th Anniversary of the Launch of the IBM System/360 Mainframe!

So you think you know your IBM mainframe history? In honor of the IBM mainframe turning 50 this year, we have put together some real brainteasers that will either display how smart you are… or how old you are - or possibly both!

1. On what date did IBM announce the System/360 product line of computers? You can earn one point each for the day, the month and the year.

2. IBM spent approximately how much developing their System/360?

3. What was the first operating system released for the System/360?

4. Name the chief architect involved with the System/360.

5. List the type of devices associated with these early model numbers.
   a) 3420-8
   b) 2314
   c) 1403
   d) 2540
   e) 1052

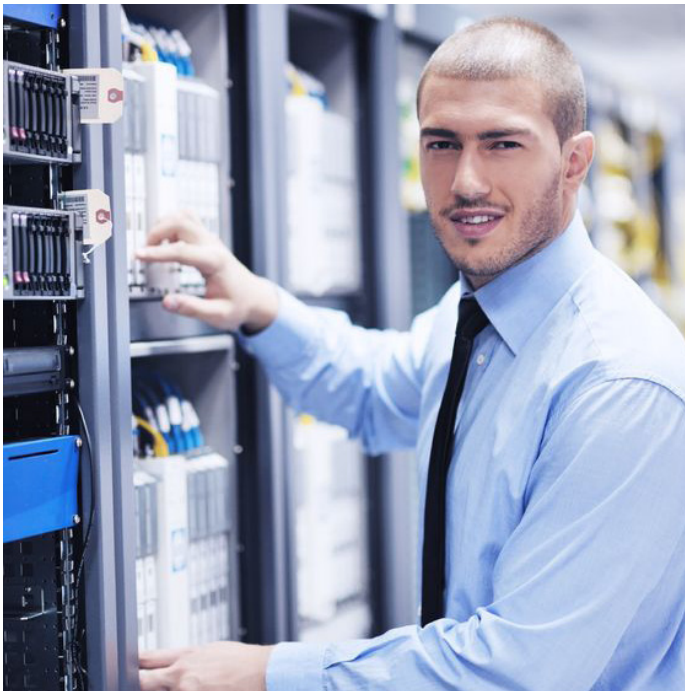
6. The names for the early IBM mainframe system coincided with the decade they were released (i.e. System/360 was in the 60s). In which decade did the mainframe not follow this naming convention?

7. List the year that the mainframe operating systems below were introduced (one point for each).
   a) MVS
   b) MVS/XA
   c) MVS/ESA
   d) OS/390
   e) z/OS

8. Which IBM product was a predecessor to JES2, performing job submission and output printing services? Receive an extra point if you can also identify the product that later evolved into JES3.

9. JES2 was first introduced with which operating system?

To see how you fared, check your answers against the solution at the end of the newsletter.



What makes a great Data Center employee?

By Greg Hamlyn

Today's IT specialist needs more than just technical smarts. I have met and worked with a wide range of Data Center people over the last 25 years (the stories alone I am saving for my memoirs), with the majority having good technical knowledge. But is that enough today to be considered a valuable employee?

In this article we look at the skills and personal traits that can make you a great Data Center employee.

1. Technical

It is logical that having technical skills and knowledge is a must for anyone wanting to work in a Data Center. Having said that, many organizations today are focusing less on employing people with a deep and specific skill-set. Instead they prefer those with a broader technical knowledge who have shown that they can pick up new technical information quickly.

You should consider the following in relation to your own technical skills and knowledge requirements:

- Do not rely on the organization to supply all of your technical learning requirements. Some Data Center staff are quite happy to sit back and just go about their day-to-day business, and wait for management to decide the training (if any) that they need to attend. These people are likely to be left stranded if there is a shift in IT focus away from their core tasks. You need to look at any opportunities that result in learning (not necessarily training courses) and present them as opportunities to your boss.
- You need to employ a method that you can use to keep abreast of latest products and trends. While your work may require you to be fighting fires all day, you still need to put time aside to identify trends and opportunities.
- Present findings to your boss/ management indicating how the latest technological advances will improve the business. This is a great way of showing that you are adding value to the organization.
- Join specialty technical groups with the same interest. This will allow you to build your own network, which can be very beneficial if you do not have any internal staff that you can bounce ideas off or use help resolve a problem, as is the case with many IT areas.

2. Communication

You can have the greatest technical skills in the world, but if you are unable to get your

message across, or understand what work a customer requires, then no-one will want to deal with you.

A person that has technical knowledge and communication skills is worth their weight in gold. These people ask the right questions, clarify requirements, and listen carefully to what customers (internal and external) are saying. They are able to present a need to the organization (whether this is another data center group, or management), and the solution.

Communication skills are also important if your organization has implemented a coaching or mentoring program. If you are chosen as a coach or mentor then you will need to impart your skills and knowledge to other people, something that many technical people find difficult.

3. Enthusiasm

You probably know or work with someone who just wants to keep the status quo. This type of person greets new ideas with skepticism and these ideas are normally downplayed, casting them in a negative light.

This negativity creates an undesirable scenario for people working with this type of person, especially if they are a manager. It dampens enthusiasm, often to a point where some possibly beneficial IT solutions may not even be investigated. Organizations want to keep people that are innovative and enthusiastic, and they often place this trait higher than technical skills.

4. Openness

Some Data Center staff I have worked with throughout the years like to keep all of their knowledge to themselves. This could be so that they are seen as invaluable and no-one can replace them (or the fact that they aren't doing too much and don't want anyone to find out). Organizations can certainly benefit by sharing, knowledge so you should jump at the opportunity to share what you know and in the process look to gather some knowledge yourself.

Openness can benefit you and the organization in the following ways:

- Cross-training provides a multi-skilled workforce for the organization, ideal in the situations where people leave the organization, or move into another position internally. It could also provide an opportunity for you to move into an area you are more interested in, or that has a longer career span.
- During disaster recovery, you will never know which staff members will be available, so knowing more is going to be beneficial to you and the organization.
- It is well noted that many experienced Data Center personnel are approaching retirement age. This huge bank of skills and knowledge does not need to be lost if proper succession planning is put in place (of course this requires that the person who is retiring is willing to impart this information).

Summary

What constitutes a great Data Center employee will probably vary between organizations, but what is clear is that versatility is becoming more preferable to pure technical expertise. When an organization has a limited number of staff, and a limited budget, they will want to be able to spread them as the need arises and still have the type of employee that can contribute new ideas. If you have all of these

attributes, you can feel safe that you will be in demand.

Greg Hamlyn, a senior technical course designer and developer with Interskill, has over 25 years experience working in the mainframe and z/OS data center environments, having performed in numerous operational roles ranging from computer operator to national mainframe technical training manager for a large Government agency. He has presented at AFCOM and has written a number of articles relating to training methods, trends, and everything that is z/OS.

You can contact him at g.hamlyn@interskilllearning.com

Vendor Briefs

In this issue of Vendor Briefs we will look at not only the products that have been recently released by the major mainframe players since our last update, but also at a number of interesting findings that have emerged from surveys commissioned during this period.

IBM

z/OS 2.1

z/OS 2.1 has been out for a while now, so what can we make of this release? Like previous releases, IBM has focused on a number of key areas for improvement:

- Processing speed: high performance FICON, SMC-R protocol for processor to processor communication, and improved DFSMShsm migrated data set retrieval are just a few of the items in z/OS 2.1 that provide this benefit.
- Managing increased size of data: file system size increases, and new data compression algorithms have been implemented.
- Smarter tools such as z/OSMF, which in this release supports the creation of complex IT workflows, full capacity provisioning management functions, and greater tracking and management of SMP/E controlled software.

z/OS Explorer (aka IBM Explorer for z/OS)

Although it has been available for some time, it is worthwhile touching on the z/OS Explorer product as it fulfils IBM's promises of creating more sophisticated tools that are easy to use by those new to the mainframe environment. The z/OS Explorer is an eclipse-based platform that connects to your z/OS system and provides you with secure access to z/OS data sets, zFS files, and JES jobs and their associated output. The z/OS Explorer also comes embedded with several other useful products: CICS Explorer, IBM Rational Developer for System z, IBM Enterprise Explorer, and IBM Problem Determination Tools Studio. Best of all, z/OS Explorer is free for those with a valid z/OS license.

IBM Wave for z/VM 1.1
This is another GUI interface tool to hit the arena is IBM Wave for z/VM. For many of you that are running Linux on System z, IBM Wave for z/VM simplifies monitoring and management of the z/VM and Linux environments through an intuitive interface. The time taken to perform tasks, such as provisioning new Linux servers, can be completed in a fraction of the time, while high level performance monitoring quickly identifies any issues that needs to be addressed. IBM Wave for z/VM is available for z10 servers and above, and is sure to be of interest to those organizations running an IBM Enterprise Linux Server.

CA Technologies

CA VM Manager for Linux on System z
CA VM Manager for Linux on System z is a new version of this comprehensive Linux-on-System z solution. It takes advantage of the latest z/VM 6.3 enhancements by providing a GUI interface that allows you to easily deploy and manage mainframe Linux images, automate routine z/VM and Linux tasks, monitor system performance and perform backup and restore tasks associated with the z/VM and Linux on System z data. A similar offering to IBM Wave...?what is this ellipsis for? must obviously be an area of concern for organizations that are increasing their investment in this infrastructure.

CA LISA 7.1
CA LISA 7.1 is a new version of CA LISA for use by DEVOPS professionals which was released late last year. It contains over 40 new features used to optimize the application development process, including the capability of testing new applications in simulated environments. New and enhanced mainframe features include support for IBM's Customer Information Control System (CICS) transaction gateway (CTG); the ability to virtualize more types of mainframe implementations as well as to trace and virtualize transactions through mainframe CICS.

BMC

BMC Cost Analyzer for zEnterprise
Which data center isn't looking to save money? Well, BMC's Cost Analyzer for zEnterprise aims to do this through the use of predictive analytics. In short, this product identifies system peaks and then analyzes the data to recommend cost reduction strategies in relation to the organization's mainframe licensing charge (MLC). BMC estimate that a typical customer consuming 5,000 MIPS at an annual cost of $3.6 million, could save $720,000 or more, so this is well worth investigating!

MainView SRM V7.8
This new version contains a number of features used to assist storage administrators with analyzing and fine-tuning data set usage. A comprehensive analysis of an application's use/non-use of data sets, including historical trends, is available, which is useful for determining better data structures. Data set backup statistics can now be analyzed, identifying information such as when the last backup was taken, and the number of backups retained. This information can highlight potential recovery-related issues, and whether data sets are being backed up too often.

Compuware

Strobe
Compuware has made several enhancements to Strobe, which is an

application performance measurement and analysis solution. Like a number of other vendor products being released these days, Strobe has been revamped providing a new and intuitive GUI interface designed specifically for new mainframe users. This release has improved DB2 reporting capabilities, analyzing data to highlight potential issues, and recommending any action that should be taken.

Surveys

A few interesting surveys have surfaced since our last newsletter.

BMC's 8th annual worldwide survey of mainframe users
This survey was completed by 1184 technical professionals and executives, spread globally. It begins by comparing mainframe perceptions and workload processing requirements of organizations from 2012 to 2013. The report doesn't show any alarming trends with statistics almost identical to the previous year's survey. Some key points from the survey:

- 93% see the mainframe as part of their long-term solution, with 70% indicating that it will play a key role in Big Data plans.
- 50% percent agreed that mainframe will continue to attract new workloads.
- 76% of large shops expect MIPS capacity to grow as they modernize and add applications to address business needs.
- No large shops - and only 7% of all respondents - have plans to eliminate their mainframe environment.

As you would expect, keeping IT costs down remains a key concern with 85% of respondents identifying this as their top priority area. A full version of the survey can be downloaded using the link below.

http://go.bmc.com/forms/MCO_DMSDB2_MainframeSurveyResults_8thAnnual_BMCcom?emailsource=press

Compuware survey
A recent Compuware sponsored study of 350 CIOs from enterprise organizations, looked at the impact of new technologies and trends on the mainframe application environment. The results indicate that there is increasing complexity that the majority of CIOs are concerned with, especially with the explosion of mobile and social data and technologies. Some interesting points from this survey:

- 91% of CIOs say high customer expectations are increasing the pressure on the mainframe to perform.
- 87% of CIOs believe complexity is creating new risks in relation to application performance.
- 70% of CIOs say mobility has increased MIPS consumption by over a quarter since their mainframes began interacting with mobile applications.
- 63% of companies are unaware of application problems until calls start coming into the help desk.

2014 Arcati Mainframe Yearbook
The Arcati mainframe yearbook is eagerly awaited by mainframe professionals each year and contains an annual user survey, a directory of vendors and consultants, a media guide and a selection of white-papers covering mainframe trends and directions. This year's survey was completed by 100 individuals, and as you would expect, a large percentage of respondents indicated some growth in workloads and this was especially the case with those larger organizations surveyed.

The results of survey questions relating to hot topics such as big data usage, and BYOD adoption can be found along with organizations' planned mainframe purchasing and expenditure in the yearbook. It can be downloaded using the link below.

http://www.arcati.com/newyearbook14/

Tech-head Knowledge Test

With every release of this newsletter a mini online test will be provided of a product that you are likely to be using in the workplace. You may want to challenge your work colleagues or just want to confirm your own knowledge!

The test for this newsletter focuses on z/OS 2.1 JCL coding and consists of 15 questions. Click the link below to start.

Coding your JCL

Learning Spotlight – Data Center Storage

One of IBM's mandates is to update its set of mainframe tools so that they automate as many tasks as possible, are visually appealing to the new group of mainframers joining its ranks, and are flexible enough so that organizations can add to it themselves. z/OSMF is one such product whose functionality continues to expand with each z/OS release.

The module that we have provided for you here is Introducing the z/OS Management Facility which describes how z/OSMF is used, and provides an overview of its ISPF interface, Incident log and the new Workflows feature.

We hope you enjoy it.

Takuzu Solution

Here are the answers from the previous newsletter.

0	1	0	0	1	1	0	1	1	0
1	1	0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0	1	0
1	0	1	1	0	0	1	0	0	1
1	1	0	0	1	0	0	1	1	0
0	1	1	0	0	1	1	0	0	1
0	0	1	1	0	1	0	1	0	1
1	0	0	1	1	0	0	1	1	0
1	1	0	0	1	0	1	0	1	0
0	0	1	1	0	1	1	0	0	1

technical: Eight Ways to Improve Your Assembler Programs

By David Stephens, Lead Systems Programmer at Longpela Expertise

A couple of years ago I had to work on an existing High Level Assembler (HLASM) program that wasn't performing very well. By not performing well, I mean it ran like a total dog. But as I started to look at the code to get a feel of what it was doing, my heart fell. No comments, no subroutines, no error handling, and no real structure. The code was a complete mess. This meant that a one week job was now going to take at least three times as long.

It's easy to write bad assembler code. In many cases, a module that starts life as a 'quick and dirty' fix is modified and changed, to the point that it soon gets out of control. Or a programmer hasn't coded for a few years and needed to create something in a hurry. But let's take a step back and have a think about this. When we write assembler code, we really want code that:

1. Works.
2. Is efficient.
3. Is well documented - so you (or the person following you) know what it does, why it does it, and how it's used.
4. Is easy to diagnose – someone may have problems with it later, and that someone may be you.
5. Is easy to modify – there's a good chance it will need to be modified at some time.

The good news is that writing quality assembler code doesn't have to take any more work. In fact, the chances are that it will save you work, both now when you do the coding, and later when you need to go back and work on it some more.

So, here are my top eight easy ways to quickly improve the quality of your assembler code.

Step 1: Become a Writer

In assembler comments are free. They have no effect on the running of the module, so it makes sense to use them to make your program easier to understand. You can't do this with the odd comment or two. You need to become not just a writer of code, but also a non-fiction writer. This way you can explain clearly and in detail exactly what is going on.

To give you an idea of the number of comments needed, I usually have more lines of comments than lines of code.

I have a standard way of putting my comments together. I separate my comments into four different types:

a. Headings

These are blocks of comments that give an overall explanation of a module, subroutine, or section. My headings look like:

```
* ================================
* Heading text
* ================================
```

At the very beginning of a program, I will have a heading with a lot of information, including:

- Program name.
- What it does, and why.
- What parameters are passed in.
- What information is returned.
- Register usage (eg base register, workarea register).

- Return codes.
- A clear explanation of what goes on inside.
- Any external references (external programs called, subsystem functions used etc.).
- Binding information (31 bit or 64 bit, APF authorised or not, re-entrant or not).
- Change history – who did anything to it, what they did, and when they did.

I'll have similar information for subroutines.

b. Subheadings

These divide the code into easy to analyse sections. They look like:

```
* ------------------------------------------------------------
* Comments
* ------------------------------------------------------------
```

I'll usually have one of these for every 20 - 30 lines or so of assembler code.

c. Full Line Comments

A single line explaining something. They look like:

```
--- Comment  -----------------
```

The dashes make it stand out from the code.

d. Inline Comments

These explain a line of code. They are on the same line as the instruction, and start in column 40 (or as close as I can make it).For example:

```
        LA    R3,1(R3)                  Move to next character
```

You'll notice that my comment above doesn't say what the line does, but why it does it. So it doesn't say "increment R3" (you can see that from the code). And I like to have all of my inline comments start at the same column to make my code look more professional.

2. Don't Write in Assembler

During my first year at University, we had a lecturer that obsessed about structured code. By structured code, he meant:

- Code that starts at the top, and works down. No jumping around the code.
- Code that is broken into small subroutines – making it easier to read.

In our first year, we wrote everything in Pascal (this shows how old I am). In our second

we wrote first in Pascal, and then converted the program to assembler (it was PDP-11 assembler at the time). Doing this resulted in an assembler program that was (more or less) structured, and also much easier to read and understand. Twenty-five years later, I still use this technique for my larger assembler programs. I'll even write the high level version of the code in comments to the right.

For example

```
        LA    R5,STRING
        LA    R1,10
LOOP1   DS    0H          For i = 1 to 10 until string [i] = '0' do
        CLI   0(R5),C'0'
        BE    ENDLP1
        LA    R5,1(R5)
        BCT   R1,LOOP1    Next i
ENDLP1  DS    0H
```

3. Check

This sounds silly, but check for errors. Always. Any time you call a program or service check to see if it worked and handle things if it didn't. Always. I've lost count how many time I've seen programs that just call a module and assume it works.

Similarly if a value is passed in from another program or a user, check it. Always. Have error handling routines that handle invalid or missing values.

I even go one step further. I like to always have an ESTAE routine to trap abends, particularly if I'm hopping through control blocks. What this routine does depends on the routine, but usually it will output a summary dump (you can specify this in the SETRP macro in your recovery routine), recover with a message (saying that an abend has been detected), and exit with a return code indicating that an error has occurred. And yes, I always test my ESTAE routines.

Because this single point will make such a huge improvement to your code, I'll say it again. Check everything. Always.

4. Write Re-Entrant Code

I automatically write all my assembler code as re-entrant. This has a couple of advantages:

- The size of my programs is smaller (it doesn't include my working storage)
- I don't have to think whether my module needs to be re-entrant (or refreshable), or not.
- Event if my routine does not need to be re-entrant now, there will be no problems (or work) if it needs to be in the future.
- If my routine is later used in a way I didn't think of, there will be no re-entrancy problems.

- In today's mainframe processors, re-entrant code can run faster.

I find writing re-entrant code is no more work than non-re-entrant code. You can find out more about re-entrant codes from the article Understanding Re-Entrant Programming.

5. Use Subroutines

Most assembler programs I see are just one program – no matter how many lines of code. I've seen assembler routines that need three or more base registers. A far better idea is to break up large programs into smaller subroutines. This gives you a lot of advantages, including:

- Less base registers.
- Easier to keep track of register usage.
- Easier to understand your program.
- If code is used in more than one place in your program, you only code once.
- Easier to steal code. If you have another program that needs similar code, it's much easier to use a subroutine, rather than try to use part of a program.
- Easier to solve problems. Let's say you have an abend in your program. If the abend occurs in your subroutine, you have only 20 or 30 lines of assembler to debug. Without subroutines, you debug the entire module.

I have a standard way of separating code into subroutines. I call a subroutine by:

```
            (processing)
         LA   R1,PARMS           R1 -> Parms
         LA   R15,=A(SUB1)       R15 -> Subroutine SUB1
         BASR R14,R15            Jump to subroutine
            (processing)
MAINEND DS   0H
         PR                   End of main program
         …
         DROP ,
```

The subroutine code looks like:

```
* ============================
* SUB1
*      Perform some processing
*
*      Input:  R1 - Parms
*              R13 - Workarea
*
*      Output: R1 - Updated parms
*
*      Register Usage:
*             R10 – Input parms
*             R12 – Subroutine addressability
*             R13 - Workarea
*
```

```
* ============================
SUB1     DS  0H
         BAKR R14,0          Save callers' environment
         LR   R12,R15        R12 = Base Register
         LR   R10,R1         R10 -> Input parms
         USING SUB1,R15      Addressability
         USING WORK,R13      Address workarea
            (processing)
SUB1END DS   0H
         XR   R15,R15        Zero return code
         LR   R1,R10         R1 -> Updated parms
         PR                  Return to caller
* ------------------------
* Constants
* ------------------------
#SUB1NAME  DC C'SUB1'
         LTORG ,
         DROP ,
```

You'll see that the subroutine is completely separate from the main program. It has separate addressability, and the main program passes parameters via R1. In this example I use BAKR and PR, so no savearea needed. The subroutine and main routine share a workarea (rather than call Getmain/STORAGE for every subroutine), but I try to separate the areas used by the main program and each subroutine in the workarea. I also like to use labels that make it obvious whether it's for the main or subroutine (for example, all the labels in my subroutine above start with SUB1). Finally, I also like to have independent ESTAEs for my subroutines.

6. Think About Labels

I like a standard naming convention for my labels, so I quickly know what they are, and how they can be used. This reduces the chances of errors creeping into my code. For example

All my equates start with a @, so I can't mistake an equate for a data constant.
All my data constants start with a #, so I can't use a constant as a variable or equate.
All my subroutine labels start with four characters that identify the subroutine. This way I can't accidentally use a subroutine variable or label in a different subroutine.

7. Consider Standard Macros

Many assembler programmers use a standard set of macros to make their programs easier to understand. Find out more from the following articles:

- Structured Programming in Assembler – Charles Davis, 1999
- Structured Assembler Language Programming Using HLASM - Ed Jaffe, 2008

IBMs HLASM Toolkit product includes a set of macros for structured programming. You can find some freeware versions from file 438 on the CBTTAPE website.

8. Test Everything

Everyone knows that you need to test your code. However it's easy to just test the basic program operation. Don't fall into the trap - test EVERYTHING. Here's an idea of what I test:

- If the program accepts parameters, I test if no parameters are passed.
- If the parameters need to be in a certain format, I test if the parameters are not in that format, or are invalid.
- I test all ESTAE routines by temporarily inserting an ABEND macro in the program.
- If I call another routine, I test my error handling. Ideally, I temporarily insert some code that will cause the other routine to fail.
- If I call a service, I test my error handling. For example, if I call the STORAGE macro, I may temporarily insert an instruction just before this macro that will cause the STORAGE macro to fail.
- I will test my re-entrant program to make sure it really is re-entrant (see the Re-Entrant Programming for Beginners article for more information).

Often I will write another program that I use for testing. This test program will:

- Call my program with every possible combination of input parameters, and will check the output of my program.
- This program will call my program a few thousand times. This checks that I've release any storage that I've obtained, and there aren't any other things I've forgotten.

Conclusion

That assembler program that ran like a dog really did turn out to be a nightmare. There were absolutely no comments, the code hopped around everywhere with branch statements, and it used undocumented ways of getting IMS information. I spent an entire week just trying to find out what the program did, and how it did it.

If the original programmer of that routine followed these eight hints, my job would have been far easier. And what's more, these hints are easy to implement in your programs. If you use them, you'll have fewer errors, and more robust code that is easier to understand and debug.

Source:
LongEx Mainframe Quarterly - August 2013, Retrieved from
http://www.longpelaexpertise.com.au/ezine/EightWaystoImproveAssembler.php

management: Software Products To Help Your Batch

By David Stephens, Lead Systems Programmer at Longpela Expertise

The chances are that you don't often think about your batch processing. Your schedules have been settled for years; occasionally altered, never rebuilt. Batch streams continue without much intervention, and in many cases few know why some jobs run. The chances are that you only think about batch when there's a problem: a job abends, a schedule over-runs into an online window, or output isn't received on time.

But it's no secret that batch is critical for most z/OS systems. From backups and log management to statements and output; from processing bulk data from other systems and organisations to data management and mining. Batch processing and schedules remain a quiet, but important part of your processing.

By now automated batch scheduling products such as Tivoli Workload Scheduler, CA-7, BMC Control-M and ASG Zeke have become the norm, taking a lot of the hard work out of batch scheduling and management. But what other software products can help with batch?

Control

One of the big features of z/OS is its sophisticated batch functions. Automated job scheduling software can automate the complicated schedules that can come from this sophistication, and it's easy for these to get out of hand. Sites with daily batch schedules comprising of hundreds of jobs are not uncommon. Most automated batch schedulers include features to view and manage these schedules. However some software vendors have decided that these aren't enough, and provided additional features including graphical representation of batch schedules. These include APS Advisor, ASG-APC Batchmap, ASG-Workload Planner, Software Engineering of America VisiTrac, HORIZONT TWS/Graph, APS Visualjob and APS TWSemon.

Other software performs batch schedule analysis – assisting in reducing batch windows from smarter scheduling. These include ASG Workload Analyzer and BMC Batch Impact Manager

Controlling batch is another issue. Do you allow SAS to run on all LPARs, or just a few? Who can submit jobs to each LPAR? MVS Solutions ThruPut Manager helps here.

Restart

In an ideal world, your batch always works. Unfortunately, we don't live there. So from time to time, batch jobs fail: a space abend, an application failure, a loop, or an operational issue. And when this happens the question is "where do we restart the job?". At the beginning, from the failed step, or not at all? If restarting from within the job, there's a lot of work deciding the appropriate step (some will rely on datasets or processing from previous steps), cleaning up after the failed step, and preparing for the run. Products such as CA Workload Automation Restart Option (CA-11), ASG-Zebb, and Control-M/Restart have been automating much of this manual work for years. Tivoli Workload Scheduler includes similar functionality in the base product.

Batch restart becomes even more

complicated with jobs that work with databases. For many years IMS Batch Backout Manager has allowed failing IMS batch jobs to backout batch database updates. Several products go one step better, allowing batch to restart at the last successful IMS checkpoint. These include BMC Application Restart Control for IMS, CA Mainframe Restart Manager for IMS and IBM IMS Program Restart Facility. DB2 doesn't miss out with BMC Application Restart Control for DB2 and Softbase Checkpointing Facility. Some products extend this idea to VSAM, including BMC Application Restart Control for VSAM. And Relational Architects' Smart/Restart covers multiple RRS activity including IMS, DB2 and Websphere MQ.

Performance

When your overnight batch schedule starts impacting your online processing during the day, or reports and data transfers don't meet SLAs, batch performance becomes very important. We take a closer look at batch tuning in our article Why Your Batch is Running Slow. However there are some products that can speed up your batch.

Dataset buffering software such as CA Hiper-Buf, Serena StarTool IOO, Rocket Performance Essential, and Dino VELOCI-RAPTOR batch can dynamically improve your VSAM and QSAM buffering without application or JCL changes, potentially removing large chunks from your batch processing time. Other software such as Macro4 VSAMTune and Software Engineering of America SmartProduction can analyse VSAM performance and recommend changes. Not exactly a software product, but Critical Path Software provides services and software to analyse dataset performance for entire systems, recommending JCL and definition changes to speed them up.

DB2 performance can severely impact batch. Other than standard DB2 tuning tools, there are performance tools aimed directly at DB2 batch. These include SoftBase Batch Analyzer Facility. CA-IDMS users also have options such as ARCH Consulting BATSTAT and Hybrid Systems DBSTATS.

Sometimes batch jobs can wait for DFHSM dataset recalls. A batch job requiring tens of dataset recalls from tape can potentially wait for some time. Some software products can recall these datasets before a batch job executes – saving this time. Examples include DTS MON_PRECALL and OpenTech HSM/Advanced Recall.

Some software products combine some of the above features, and provide additional performance analysis and improvement functions. These include Software Engineering of America SmartProduction, BMC MainView Batch Optimizer, and Trident z/OSEM.

Reliability

We all hate production batch that stops from JCL errors. Or space errors. Or RACF security errors. Fortunately, JCL scan software can check the JCL syntax, confirm that datasets exist, and that all the necessary RACF or security rules are in place – all before the job is submitted. Many will slip seamlessly into your batch automation product, performing automatic JCL scans when schedules are prepared. Some can also be used to enforce JCL coding standards. Examples include ASG Job/Scan, ASG JCLPREP, CA JCL Check, and E-GEN JCLChecker.

Output produced by batch can be critical. Software to verify the validity of this output can be invaluable. Examples include BMC

Control/M Analyzer, ACR/Detail, ASG ViewDirect/ABS and Beta 91 Automated Balancing and Quality Manager.

Conclusion

Aside from the normal batch automation software, there are lots of options to improve batch schedule management, control, reliability and performance. The software here is just the start. Lookup Mainframe Software is a great place to start investigating the options.

Source:
LongEx Mainframe Quarterly - November 2013, Retrieved from http://www.longpelaexpertise.com.au/ezine/BatchProducts.php

technical: Why Your Batch is Running Slow

By David Stephens, Lead Systems Programmer at Longpela Expertise

Batch continues to be a critical part of most z/OS workloads. However most of our performance-related tuning is based on CPU consumption and online response times. It's only when a batch stream runs outside of its window that batch gets its share of performance work. This month, I'm giving my seven top reasons why I find batch jobs run slowly, and what you can do about it.

7. Waiting for Dataset Recalls

Often jobs, particularly those that run weekly, monthly or quarterly, will hang around waiting for dataset recalls from DFSMShsm or similar products. To my mind this is classic wasted time that can, and should be avoided. The obvious answer is to manually recall datasets an hour or two before the beginning of the batch run. If this is too hard, there are a few products that can help you out, automatically recalling migrated datasets before a schedule starts. These include OpenTech Systems' HSM/Advanced Recall and DTS Software's MON-PRECALL.

One pet-hate I have is when batch jobs recall datasets to delete them. Usually they use IEFBR14 to allocate the dataset with a disposition of (MOD,DELETE). The z/OS ALLOCxx parmlib option IEFBR14_DELMIGDS(NORECALL) will save this pain, and convert the HSM recall to an HDELETE. If this can't be done, using the IDCAMS DELETE command will do the same job.

6. Bad Choice of Utility

Far too many times I see sites copying between datasets using IDCAMS REPRO or IBMs standard IEBGENER. These utilities use default dataset buffering, which is rarely good. A far better option that guarantees optimal buffering is probably already installed in your site: the IEBGENER replacements DFSORT ICEGENER and SyncSort BETRGENER. Everyone knows that these can copy sequential (QSAM) files. Less know that they can also handle VSAM.

If installed on your disk subsystems, Snapshot or similar features are another great option: instantly copying datasets with minimal CPU overhead. Similarly, I think you would be crazy to use IDCAMS EXPORT/IMPORT when backing up or defragmenting VSAM datasets. DFSMSdss, CA Faver and Innovation FDR/ABR can do much better.

5. Waiting for CICS Files

A very standard scenario is an overnight batch run that updates CICS VSAM files after the day's trading. Unfortunately, many still manually close these files in CICS, and then start their batch run. If you absolutely need a clean point to do this, then OK. But the chances are that you don't. You have a couple of choices to improve here. You could use CICS EXCI to call a CICS program to access the CICS/VSAM file from batch, or you could use features like DFSMSdss BWO features to backup CICS/VSAM files with data integrity. VSAM RLS and DFSMStvs are other options. There are also products such as DSI Document Systems' BatchCICS-Connect and MacKinney Batch to CICS that can help.

4. Waiting for Datasets

Of course there will often be several jobs that need access to the same datasets. So accessing them with a DISP=OLD, and waiting for previous jobs seems obvious, and in the past mandatory. But there are smarter options.

If you need concurrent access to VSAM datasets, features like VSAM RLS can allow multiple jobs to access datasets at the same time. Hiperbatch may also help when multiple jobs need to read a dataset at the same time, and BatchPipes can have a writer and reader running concurrently. Software products such as BMC Mainview Batch Optimizer can also help here.

3. Waiting for Initiators

In the old days, we used initiators to limit the number of batch jobs executing to manage CPU usage. In many shops, this is still the case. However today we have a far better tool: WLM. In fact today, everyone should be using WLM managed initiators to manage the number of initiators based on workload, rather than setting hard limits.

2. Waiting for Tape Drives

I remember a site where we only had eight 3480 tape drives shared between two z/OS systems. If a tape drive was onine on System A and needed on System B, it would first need to be taken offline from System A, and then brought online to System B before it could be used.

Those days are history, and dynamic tape sharing is the norm. And obsolete. Today Virtual Tape Subsystems (VTS) can define hundreds of logical tape drives to a z/OS system. Even if you can't afford a VTS subsystem, you can achieve similar things with DFSMShsm. Today there is no reason for a job to wait for a tape drive.

1. Dataset Buffering

Whenever I'm tuning a batch job, the first thing I look for is dataset buffering. And the pickings are usually rich. It's as if IBM deliberately set default dataset buffering values for the worst performance. So if you're not setting BUFNO for QSAM, or BUFND/BUFNI for VSAM in your batch, you probably should be. Of course modifying DD statements for every batch job is a huge task. Using VSAM System Managed Buffering (SMB) is one excellent option. There are also many products out there that do it for you: from CA Hyper-Buf to Rocket's Performance Essential.

What Isn't In My List

In many ways, what isn't in my list is as interesting as what is. Dataset compression doesn't make it: it will increase your CPU, but probably not your batch response.

Similarly features like dataset striping are great for online performance, but I haven't yet seen them make a big difference to batch times. VSAM tuning and QSAM blocksize selection can make a big change to your run times, but most sites are on top of this. Batch scheduling is often a cause of delays, but automated scheduling packages and tools that most sites use will help overcome this. And there are a range of other things such as database and SQL tuning. However from my experience, my top seven above will give you the most batch elapsed time reductions for the least effort.

Source:
LongEx Mainframe Quarterly - November 2013, Retrieved from
http://www.longpelaexpertise.com.au/ezine/WhyYourBatchisSlow.php

Mainframe 50th Birthday Quiz Answers:

1. 7th April 1964

2. 5 billion dollars

3. DOS/360

4. Gene Amdahl

5.
   a) tape drive (1970s)
   b) disk drives (late 60's / early 70's)
   c) early printer model
   d) card reader/punch
   e) console (attached to the System/360)

6. 80s (instead of System/3??, it was Model 3081)

7.
   a) 1974
   b) 1983
   c) 1988
   d) 1995
   e) 2001

8. HASP (Houston Automatic Spooling Program, was the predecessor to JES2)
   ASP (Attached Support Processor, was the predecessor to JES3)

9. MVS