Week-1 Reading - Introduction to IBM i

Contents

By now, you've seen the wonderful green screen and are probably shaking your head as to why you have to learn something that looks so old. Up until now, you've been an IT consumer. Websites and games attract you to them by being fun, colorful and have lots of tricks. IBM i can do all of that too, but you have to learn to walk before you run. We're studying the business side of computer science this semester. People who use this type of software are being paid to use it. There's no need to spend the money making the screens 'attractive'. Anyone who does data entry for a living will tell you that a mouse is annoying. Leaving your hands costs from the keyboard slows you down and time costs money.

The screens that we develop are not meant to be used by the public, but by people employed by a company to get a job done. If you're interested, DOM545 is a really great course that shows you how easy it is to create glitzy applications for IBM i!

History Lesson ^

Back in the early 1960's computers were making their way into the business world as an effective tool to make accounting more efficient. Today, you still see that the Director of IT reports to the Chief Financial Officer of a company, although more and more companies are adding Chief Information Officer to the board of directors. The 1960's calculator took up rooms and needed extensive air conditioning systems to run. Programmers were hard to find and very expensive to hire. IBM decided to develop a 'smaller' in size and cheaper computer. The operating system and languages used were to be easy enough for accountants to understand, making the machine much more attractive to use. Dr. Frank Soltis's undergraduate work in architecture of technology-independent machine interface (TIMI) and single-level store led to the creation of the System 36 and later the System 38.

System 36 ^

Companies still use the System 36. The machine uses a simple operating system language, OCL and all commands can be found on interactive menus. The machine was designed to be a 'batch' oriented data collector, meaning that data was entered in large quantities, manipulated and regurgitated in easy to understand formats. The operating system is OCL which is a simplified version of JCL that mainframes run. RPG and the subsequent RPGII are the languages of choice for data manipulation on the machine. COBOL also runs on this machine which allows for interactive screen development. A Flat file database is used making finding data a little slow, but the machine is so easy to use, that it more than makes up for it. Now accountants had a tool that they could easily program.

System 38 ^

Dr. Soltis laughs about this machine. This was an experiment gone good! He never meant for it to be a commercially viable machine. Still, today, Dr. Soltis marvels at how quickly this machine took off and wonders why companies still use it! My aunt and uncle have the same feelings about their oldest son, Luke. Luke now wanders the world as a respected DJ and Music Producer and Uncle Lynn and Aunt Irene, still scratch their heads and wonder how that happened!

The System 38 took all of the best features of the system 36, added a relational database, DB2 and more power! The operating system language was called CL (Control Language). All commands could be found on easier to use menus, but now the commands were made up of English words! Data could easily be retrieves using DB2 which has indexing and the System 38 became an interactive machine.

AS/400 ^

Rather than fix, repair and upgrade, in the 1980's, IBM took the two technologies, System 36 and System 38 and merged them together into the AS/400. This new computer could still be run in System 36 or System 38 modes and in the new AS/400 mode. Because of Dr. Soltis's believe in 'technology-independent machine interface (TIMI)' companies could convert to this new computer over a weekend and Monday morning, when the business started up again, the screens were the same - business as usual. This philosophy allowed companies to migrate to new technologies, easing the cost of training. Wouldn't it be nice if Microsoft kept to the same philosophy? My family is still getting used to Vista and now we have Windows 7? Yikes! My daughters will transition well. To them, computer operating systems are bathrooms, they can quickly figure out what they need to do and leave a mess behind! The love of my life, Trapper Tom, now that's a different story! Sounds like an interesting blog – Trapper Tom Takes on Technology (Cindy's headache gets worse)!

iSeries ^

1990's brought the arrival the internet. **B**usiness customers changed quickly. Picking up the phone was a waste of time. Customers demanded the ability to place and follow up on orders and pay bills online. Businesses that didn't react fast enough, quickly lost market share. Now a company's website was as great a marketing tool as their sales force.

To meet the demand, the AS/400 grew into the iSeries. From the standpoint of the internal operators, it was the same machine, except it could do more! The i stood for integration. Tools were developed that could quickly turn old RPG programs into web enabled applications. Processers were replaced with the new RISC based architecture chips so now the machines could process data recursively and Java became a supported language. The iSeries became a computer that run the business's enterprise software that evolved from the 1960's and websites.

System i and Power Systems ^

AIX and LINUX grew from an IBM 'pet projects' into demanded operating systems during the 1990's and the 2000's. With the introduction of RISC technology to the AS/400, the internals of the AIX and AS/400 became remarkably similar. From a business perspective, computers needed to be available 24 hours a day, 7 days a week. From an environmental perspective, computers needed to do more with less power and produce less heat! IBM met the demand with it's new Power Systems Chip and merged iSeries with AIX into the new Power Systems computer. Now, the same computer can run the old 1960s' COBOL programs, store emails, serve web pages and run 3 dimensional data warehousing! Welcome to the new world of Business Computing!

IBC233 – System i Business Computing ^

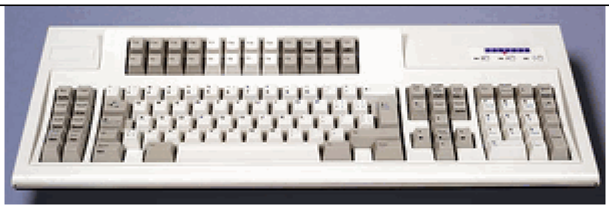
There's no way that we can teach you everything that the Power System computer can do because that would require us to teach you everything in the CPA program in one semester. IBC233 is all about the developing computer systems that a company would use internally. This is a busy course. By the end of it, you will be able to develop simple applications involving interactive data entry and batch reporting!

Be prepared for a busy semester – and don't miss a class or a lab!

Function Keys ^

A long time ago – before Bill Gates was born, function keys were called Command Keys and there used to be 24!

Recently, I was at the North Toronto Public Auction, near the 400 FleaMarket in Innisfil, Ontario. They used an old workstation with 24 Function keys! We took a picture of it, but I can't find it! Please picture, a regular keyboard with an extra set of function keys at the top!

I found this picture at: http://pckeyboards.stores.yahoo.net/122keyterkey.html	

PC's came along and didn't need 24 keys, so they took out the upper row. Who knew that PC's would become a viable business computer? Now, F12 can be F24 if you press the shift key at the same time!

System **A**pplication **A**rchitecture (SAA) are a set of standards that IBM developed in the 1980's to give all applications a similar look and feel. That's why, F1 is usually help and F3 is usually exit!

Control Language (CL) Programming ^

CL is the command language that we use to communicate with the computer. Commands are made up of English words – a verb followed by a noun. For example CRTLIB is the create library command. All commands can be found using a menu. You will memorize commands as you'll use them repeatedly, but don't worry about using the syntax as you can allows press F4 and follow the prompts.

The most frequently used command in this course are:

- WRKOBJPDM – Work With Objects using Program Development Manager. This command is similar to the LINUX ls and the DOS dir commands. It shows you everything that's in a library but does so much more!
- WRKSPLF – Work with Spooled Files. This command shows you everything that you have waiting for print and allows you to control what's printed, saved or deleted.
- DSPMSG – Display Message. This command allows you to manage your messages!

CL command can be grouped together in programs. The language is very powerful, so to make the programs run faster, they are compiled, but CL is not a high level language.

Most command or scripting languages are interpreted. A good example of an interpreted command language is LINUX Shell. An interpreted language is called a low level language because the computer that runs it must first interpret the code, then translate it into machine language. The program is never optimized so that it runs faster.

C is a high level language. The program is written in a language that programmers understand. The code is then compiled. The compiler first translates the code to machine language and then optimizes the code so that the program will fun faster.

The CL compiler is a one pass compiler. It only translates, it doesn't optimize, so CL is not a high level language. CL can do many things that high level languages such as C, RPG and Cobol can do. We choose the high level language over CL whenever possible because the resulting object will be faster.

CL programs usually start with the command PGM. PGM is only required when you are passing parameters to the program, but it's a good habit to get into! After that, variables are declared, followed by the program logic. CL programs end with ENDPGM.

Objects ^

Objects are entities on the server that take up space, have a description and are not considered to be temporary. A program is an object, but a message isn't as the message is meant to be read and then destroyed.

System Values ^

System Values are a collection of settings that customize a power systems server to a company needs. System Date is a good example of a system value as it's the same for everyone using the server.

Libraries and Library Lists ^

Libraries are objects that store other objects. Libraries are different from directories and folders that you're used to as they cannot be sub-divided into other libraries. The directory tree system does not exist in ibm I, so all libraries are equally important on this server with exception of the library, QSYS. QSYS contains all of the objects that are necessary to running the server. QSYS stores Library Descriptions which are objects that store a library's address.

Library lists help us manage the over 5000 libraries that exist on Zeus. Library lists are subset lists of the most important to us, libraries. A library list is similar in concept to a pc's search path, but it's built when you sign on and deleted when you signoff. IBM i uses a combination of system values and your user profile to customize a library list specifically for you. The first part of the library list consists of the libraries that the operating system requires. QSYS is included in this list. The second part changes, depending on the commands that you execute. The third part is the default library that's specified in your user profile and the fourth part are libraries that we have in common such as QGPL (Q General Purpose Library) and QTEMP. Try not to use QTEMP as it's a temporary library that's created when you sign on and deleted when you sign off.

User Profiles ^

Everyone needs a user profile, in order to sign on to a power systems server. A user profile stores settings specific to an individual, so information such as passwords are stored in user profiles. A user profile is an object. Some settings in a user profile can override system values. QLANGID is the system value that controls what language the operating system presents to the operator. Our ZEUS is set up to run in United States English, but the following languages are available:

AFR - Afrikaans	EST - Estonian	NOR - Norwegian - Bokmal
SQI - Albanian	FAR - Farsi	NON - Norwegian - Nynorsk
ARA - Arabic	FIN - Finnish	PLK - Polish
NLB - Belgian Dutch	FRA - French	PTG - Portuguese
FRB - Belgian French	DEU - German	RMS - Rhaeto-Romanic
ENB - Belgium English	ELL - Greek	ROM - Romanian
PTB - Brazilian Portuguese	HEB - Hebrew	RUS - Russian
BGR - Bulgarian	HUN - Hungarian	SRB - Serbian
BEL - Byelorussian	ISL - Icelandic	SRB - Serbian Cyrillic
FRC - Canadian French	GAE - Irish Gaelic	SRL - Serbian Latin
CAT - Catalan	ITA - Italian	CHS - Simplified Chinese
HRV - Croatian	JPN - Japanese Katakana	SKY - Slovakian
CSY - Czech	KOR - Korean	SLO - Slovenian
DAN - Danish	LAO - Lao	ESP - Spanish
NLD - Dutch	LVA - Latvian	SVE - Swedish
ENA - English Australian	LTU - Lithuanian	FRS - Swiss French
ENP - English Upper Case	MKD - Macedonian	DES - Swiss German

ITS - Swiss Italian	TRK - Turkish	URD - Urdu
THA - Thai	ENG - UK English	ENU - US English
CHT - Traditional Chinese	UKR - Ukrainian	VIE - Vietnamese

The language identifier setting is also available as part of a user profile, so Cindy could set her user profile, so that she sees French versions of screens, while everyone else sees English!

A general rule of thumb: if the information is specific to you, then it belongs in your user profile. If the information is common to all of us, then it belongs in a system value.

A power system server cannot run without User Profiles, so User Profiles are stored in the library, QSYS.

Device Descriptions ^

The last piece of the puzzle is device descriptions. We need hardware (PCs and printers) to communicate with the server. Each workstation at Seneca has it's own device description on the server. Temporary device descriptions are created when we sign on to the server outside of Seneca. Since a power system server cannot run without device descriptions, these objects are stored in QSYS.

Files ^

Files store data – that's it!

DB2 is the database built into this operating system.

You can't 'run a file' on this system, but you can read it. In IBM I, we execute programs.

A file is similar to a textbook.

The layout of a Data Physical File is up to you. We try to define our data layouts using Third Normal Form (3NF) design techniques. You can use either SQL or DDS (Data Description Specifications Language) to define Data Physical Files. Either method accomplishes the same goal.

Source Physical are DB2 files with a layout specific designed for storing code. These files usually have more than one member. A member is similar to a chapter in your text book. Each member has the same layout, but is used for a different purpose. When writing CL programs, we would create a source physical file called QCLLESRC. Each member in QCLLESRC would be a different program.

Author: Cindy Laurin