Dear Reader: Thank you for downloading this free book from Brian W. Kelly's finished book catalog. I finished the book titled **The All Everything Machine** at https://letsgopublish.com/technology/alleverythingmach.pdf in Jultu 2016. An IBM classic book that was refreshed in 2016.

Most of my books had previously been published on Amazon.

Click below if you would like to donate to help the free book cause: https://www.letsgopublish.com/books/donate.pdf**.**

Enjoy!

The All-Everything Machine

The Story of IBM's Finest Computer System

BRIAN W. KELLY

The All-Everything Machine
The Story of IBM's Finest Computer System

BRIAN W. KELLY

LETS GO PUBLISH

Published by: LETS GO PUBLISH!
Joseph McDonald, Publisher
P.O Box 425
Scranton, PA 18503
jmac@letsgopublish.com
www.letsgopublish.com

Library of Congress Copyright Information Pending

Book Cover Design by Michele Thomas

ISBN Information: The International Standard Book Number (ISBN) is a unique machine-readable identification number, which marks any book unmistakably. The ISBN is the clear standard in the book industry. 159

countries and territories are officially ISBN members. The Official ISBN For this book is:

09745852-3-8

The price for this work is :
$34.95 USD

10 9 8 7 6 5 4 3 2 1

Release Date: April, 2005

Dedication

To Gerry Rodski,

A friend of the IBM i5

And

One of my best friends

Acknowledgments

I would like to thank many people for helping me in this effort.

I would first like to thank my family, starting with my lovely and dear wife, Patricia. In all my books I tell the world that Pat is my source. She is the person who keeps me alive and sane and well in more ways than can be mentioned in a simple acknowledgment. She is the glue that holds our family together. Besides that, she makes a pretty picture to gaze upon. And I do take the opportunity. Her daddy, Smokey Piotroski, called her Packy as a nickname. I love that name and the person who wears it and I use it to address my little Packy on many occasions. She is as sweet as it gets. Thank you Packy for all you do to keep me well.

I would also like to thank my twenty-year-old daughter, Katie, who is my little baby doll. Kate helps me in any way she can when I am in my writing ways. But more than that, her sweet voice and her accomplished guitar playing often put a smile on my face as my fingers pound the keys. Though Katie hasn't had it easy lately, she is on her way. I thank my Katie for being Daddy's Little Girl. I love you very much. Thanks also go out to my twenty-three-year-old son, Michael, who is in his fifth year of pursuit for a bachelor's degree in accounting. Mike moved out this semester and we wish him well in his independence. Of course I miss him but he's around often enough to put a dent in the "fridge." I also thank my twenty-four-year-old son, Brian, who continues to knock 'em dead in his third year of law school. Brian is always available to help me with his strong English language skills whenever possible. Brian is in the top 10% of his class at a wonderful school in Philadelphia. I learned first hand just a month ago how much he is loved by all the folks in his school, from his professors to his peers to the cafeteria crew. It was a very heartening experience. Mom and I are very proud of all of our children and we thank them for their efforts on our behalf.

Accomplishments often materialize because of a strong friendship infrastructure. I am pleased to have a number of great friends. Among them is my longtime best friend Dennis Grimes, who is always there to help, though he may think everything I write sounds the same. Barbara Grimes, Patricia Grimes Yencha, Elizabeth (Wizzler), Mary the PhD., Denyse back from the U.K., Grandma Viola, and Grandma Gert also pitch in whenever the opportunity arises. Dennis is always there to review a chapter or a marketing theme or whatever I ask. I really appreciate all you do for me "D." Thank you

The busiest guy on all of my book projects, besides myself, is always Joe McDonald. Joe is the businessman in our publishing venture, and in that, he's all business. Joe is the former Publisher of the Scranton Tribune/Scrantonian Newspaper. So he's got the right background to make sure everything is A-OK! I promised Joe that my next book was going to be non-technical as we moved the publishing business to Children's books and third party authors. However, because this book had to come out now, I put him off one more time. Thanks Joe for letting me get away with it… again. The Adventures of Eddy (The Dog) written by Joe's Grandson will soon be on the bookshelves of America. My thanks also go to Peg McDonald for making sure that Joe is always ready for action.

Of course, the long list of helping hands includes Gerry Rodsi to whom this book is dedicated, Jeanne and Farmer Joe Elinsky, John and Carol Anstett, Carolyn and Joe Langan, Karen Komorek, Bonnie and George Mohanco, Becker and Robin Mohanco, Lilya, Josh, and Alaina Like Mohanco, Bob and Nettie Lussi, and, of course, Frannie and Mike Kurilla, Jerry and Hedy Cybulski, Linda DeBoo and Bob Buynak, and Joe, the Chief LaSarge.

And don't let me forget Cathy and Marty Piotroski, Dr. Susan Piotroski and Dr. Mitch Bornstein, Matt and Allie, Dr. Stan Piotroski, Justin and Katie, Carol, Merek, MacKenzie, Myranda, Erin, Ralph, Lynn, and Scott Piotroski, Pierre Le Kep, Nancy and Jim Flannery, Mary and Bill Daniels, Diane and Joe Kelly, Ed and Eudart Kelly, Bill Rolland the Notre Dame Ace, Bill Kustas, Bill Kush, Steve and Shelly Bartolomei.

Special acknowledgments to Steven Dressler and Howard Klein, the top management team at Klein Wholesale Distributors in Wilkes-Barre, P A, who use iSeries technology to the fullest. Their vision, foresight, and execution have brought Klein to the enviable position of being the fifth largest candy and tobacco wholesaler in the United States.

Various members of the Klein Development staff offered information over the time in which this book was written. In alphabetical order, by first name, the Klein team includes: Barb Chaderton, Bill 'Curly' Kepics, Cindy Dorzinsky, Cindy Goodwin, Dennis Grimes, Eric Priest, Jeff Massaker, Jerry Reisch, Joe Byorick, Joe Rydzewski, John Robbins, Paula Terpak, Rod Smith, and Rosalind Robertson.

I would also like to thank Nancy Lavan, our angel at Offset Paperback, our printer. She continually encourages us in our writing and publishing efforts. Special thanks go to Michele Thomas, who takes ideas and makes wonderful images from them, such as this wonderful cover..

To sum up my acknowledgments, as I do in every book that I have written, I am compelled to offer that I am truly convinced that "the only thing you can do alone in life is fail." Thanks to my family, good friends, and a helping team, I was not alone.

Table of Contents

Preface:

It is my pleasure to write about IBM's finest computer. Without even reading the first chapter of the book, you have already discovered that the all-everything machine is an IBM product. You may or may not know that from time immemorial, IBM has made the finest computers known to mankind. Once IBM passed Univac in the early 1950's it was clear sailing for the company from that point on. There are few who would argue about IBM's quality or IBM's service in the computer field and it says something for the all-everything machine that it sits on the very top of IBM's achievement list.

I will save most of the goodies about the subject of this book until the book proper, but you know already that the machine of which I write has struck me as so elegant and so powerful that I was compelled to label it the all-everything machine. It just happens to be available from IBM. If your curiosity abounds out of control to know why anybody would select an IBM computer as an example of an all-everything machine, I urge you to feel free to digress from this Preface and go directly to Chapter 1 and you will learn enough to know why this unique machine helps businesses to be successful.

I surely hope that you like this book. I have been in the computer industry as an IBM insider for 23 years and following my career with IBM, I have been mostly independent, providing consulting services for clients in many industries. I confess to be an addict of the type of no-sweat computing that IBM brought forth in 1969 with the System/3. I have remained an addict while IBM introduced its "Future System" in 1978, the great grandfather of the all-everything machine.

The "Future System" was laden with such advanced computer science concepts that this machine is still futuristic in its capabilities. It even tricked IBM. It was so complex in its internals in order to achieve ease of use externally that the company could not make it all work in time. This forced Big Blue to postpone its initial delivery date so that the labs could have the time to make the system work. IBM wanted the system to be known for it facility, not for how many reboots an average technician could perform in an average work day. In other words, unlike other OS vendors, IBM decide to make the hardware and the operating system work together before it made the new system available for all of us to use. When the "Future System" was made available, thanks to a yeoman IBM effort, it worked like clockwork.

Few would expect that any system originally built in the late 1970s would have advanced integrated design characteristics better than all of today's computers. Yet the 21st century all-everything machine, based on the 1978 "Future System" tops the charts in terms of innate computer capabilities. If IBM made more hoopla about its major achievements in technology in the fashion of Microsoft, we'd surely all know about the all-everything machine by now. But IBM is substantially more humble than Microsoft and the company reserves its messages about its business computers for the business marketing channel, not the consumer channel.

I wrote this book so that everybody, from consumers to business people, can know about the all-everything machine. Far from "legacy" as it is referred to by the unknowing, the all-everything machine is exactly that. It is all-everything. Moreover, it is not a one size fits all take-it-or-leave-it proposition. There are sizes from very, very small to humongous behemoth. It is so granular that it can do computing jobs for very small businesses and very large businesses and those in-between. Regardless of its size, its ease-of use personality and advanced software capabilities stay the same.

The story of the all-everything machine is worth telling and it is worth hearing. If you are a business person thinking about getting your first computer to run your business, or you have a PC server or multiple servers, or even if you have a full IT department, this story is worth your time. For the technical at heart, there is enough information about this server that by reading this book you will have a much better appreciation for how the all-everything machine gets its work done, and I would expect that you will be duly impressed. In a nutshell, this book is your best bet to understanding what the all-everything machine is all about.

This book is very easy to read. Each chapter is written as a self contained essay that gives historical background and/or technology information about the all-everything machine. By looking at the table of contents, you can pick the essay that you want to read first and then go right to it and it should make sense. Of course, you might want to read Chapter 1 first to get a feeling of the machine, its relevance, and its value to business. Either way, I predict that you will enjoy this book. Thanks for taking it home with you.

Brian W. Kelly

Wilkes-Barre PA

Chapter 1

What Is The All-Everything Machine?

He Thought Chicago Was a Treat!

In the 1966 song by Norman Greenbaum titled: "The Eggplant that Ate Chicago," An Eggplant comes in from Outer Space and lands in Chicago. That's Chicago as in Illinois. Fortuitously for him, as the song goes, the Eggplant thought Chicago was sweet, "it was just like sugar." This song comes to mind as I think about how to introduce the all-everything machine.

If an alien race came to earth and evaluated our state in computer evolution and picked a winner, it would be the IBM eServer i5, a.k.a. the all-everything machine. The hardware would win, the operating system would win and if somebody could convince the aliens to pay for what they take, the i5 would also win on cost-effectiveness since they would get

a lot for the money. In fact, if our friend the Eggplant were part of the alien expedition, it's a sure thing that he would find the IBM eServer i5 to be really sweet.

Note; For you Eggplant lovers, I have included the words to this 1966 hit at the back of the chapter.

Hardware

If any machine comes closest to pure business value it is the all-everything machine. Yes, it exists on planet earth and its great grandparents have been here since 1978. In fact, with regard to hardware scalability, reliability, availability, security, ease of use, flexibility, self management, self diagnosis, and much, much more, the i5 would be a winner in every category. For one thing, it is the only server/operating system combination today that supports applications and data with 128 bit addressing.

Software

Carrying the facilities even further, with regard to software scalability, reliability, availability, security, ease of use, self management, self

optimization, and self diagnosis, again the i5 would be the winner in all categories. Throw in an integrated relational database, integrated transaction processing, built-in productivity tools, development tools, middleware, and even more. The eServer i5 has it all. In fact, it is the only operating system that supports applications with 128 bit addressing running on 64-bit hardware. As a point of note, its predecessors, the AS/400 and System/38 have been doing that for almost thirty years, and for some of those years, they were using 48-bit hardware running at an abstracted 128-bit software level.

The all-everything machine has no real software limits as to the number of jobs, threads, transactions or data active in the system. Even an Eggplant could tell that there isn't another platform on the face of the earth that comes close.

Unmatched Elegance

The Secret is now out. The all-everything machine is an IBM box called the eServer i5. Of course, I do not expect anybody to take my word for it, so I have sixteen more chapters in which to tell you about the past, the present and the future of this remarkable machine.

For a commercial server to be the one and only all-everything machine, it would have to have an internal elegance unmatched by any other server, and better than that, it would have to be miles and years ahead of anything else that has ever been built. If distance were a real factor in computing, the i5 would be register at many times the distance from the sun and back. It would be way ahead of its competition since the i5's address space is so humungous. Time is a real factor and as you will see,

the i5 is just about thirty years ahead of all systems today and, as hard as it may seem to believe, the competition is not catching up.

There is a saying in i5 user circles that only IBM could have afforded to build a system with internal integration of hardware and software that is so rich in the most advanced computer science concepts of today. That explains why no other computer / server vendor has ever nor can ever take on the task of building such an advanced server from scratch. They simply cannot afford it. In Chapter 8, you will learn in detail about the architectural underpinnings of the i5. It's a good read for the neophyte and the expert alike. In Chapter 8, I introduce the seven fundamental computer science attributes upon which the i5 is based. Nobody has ever come close to building an i5 and the reason is simple. They cannot afford the unique combination of hardware and software that is at the heart of the machine.

Computer Science Research Project

Some might argue that the closest thing to an i5 is an experimental "machine" developed at University of Pennsylvania and now in the laboratory of John Hopkins University. It is called EROS, which stands for Extremely Reliable Operating System. You can learn more about the capabilities of EROS at the following URL:

http://www.eros-os.org/

However, EROS, for all its goodness is not a machine. It is just software. EROS is more or less an experimental operating system used for pure research into several of the most advanced computer concepts that have ever been brought forth by the computer science community, namely object orientation, single level store, and capabilities. It runs on standard fare x86 boxes from 486 up. Because it is just an experimental OS, it does not have its own hardware base and thus it is not and cannot be an integrated machine. Additionally, it does not have a technology independent machine interface, integrated transaction processing, or an integrated relational database. Compared to the i5 family of machines available since 1978 in one form or another, EROS is a partial implementation and it is the closest thing out there. All of these notions are fully explained in Chapter 8.

Though special indeed and the basis for EROS sponsor Jonathan Shapiro's doctoral thesis, it is nowhere close to making it to commercial prime time. Your neighbors won't be getting one in the near future or the distant future. However, and I repeat, it is the only operating system other than the IBM i5, even in experimental stages that attempts to use the most advanced computer science concepts as its basis. Windows and Unix don't even bother. They'd have to be rewritten to participate in the advanced concepts game.

Research vs. Reality

So, the closest thing to an i5 is a project in a lab that just needs a bunch more billion in research dollars to become a real commercial operating system. But nobody is lining up with those billions. Meanwhile this humble machine called the i5, built in an IBM lab located in Rochester

Minnesota, intended for use in small to medium sized business, has all seven of the most advanced computer science attributes ever conceived.

While the theorists were theorizing, IBM actually built a machine thirty years ago that did all of the things that were in their theories plus lots more. It wasn't cheap and IBM probably would not have spent the money if it knew how expensive it would be to build, before the company had committed. Nonetheless IBM did spend the money and did build the machine and it has been used successfully by businesses and organizations across the world in one form or another since 1978.

The i5 is the fourth generation of this technology and it is without doubt the finest computer science machine that has ever been built. So, my point in bringing a brief discussion of EROS into the foreground is that EROS is the pinnacle of today's computer science research and it is not as far along as the IBM System/38 of just under thirty years ago. This operating system, however, is the best imitation of several of the advanced concepts that have been running every day in i5 systems for just under thirty years. And this imitation supports only three of the seven most advanced principles implemented in the i5. The other four will never be implemented in a usable fashion as long as EROS is dependent on using Intel or other processors than its own.

The IBM i5 is not an operating system. It is a hardware server that today runs the i5/OS operating system. Together, the integration of hardware and software on the i5 server lead IBM and the rest of the world, including EROS, in advanced computing. In this book, you will learn where this system came from, why it is so unique, and why so many of the features that have been standard fare on this all-everything machine for just under thirty years continue to be so difficult for IBM's competitors to duplicate that for all intents and purposes, they have given up.

To many businesses, It really does not matter that there is a machine out there of which they know very little if anything that is the best at everything that it does. There may be some technicians or "computer people" or some smart computer users in their organizations to whom the all-everything machine may be somewhat intriguing but most business people are not interested in the special features of any server, even those that make the i5 so special. However, to the extent that having those features adds real business value, and not having them subtracts from business value, there are plenty of reasons for business managers and entrepreneurs to want to know more about the all-everything machine.

Who Needs the All-Everything Machine?

In the next three sections, we first examine the notion of the value of a computer and how to realize its value. Then, we will whet your appetite with some of the business reasons from which a company would benefit if it were to use an i5. Following the business value factors, for the technical at heart, we will examine a large, yet not exhaustive list of the technical capabilities that are to be found only within this unique combination of hardware / software -- the all-everything machine.

Note: I would like to acknowledge the works of Paul A. Strassman, former VP of the Information Products Group at Xerox Corporation, whose works I read prior to writing this section of this book. Paul A. Strassman has written a number of IT Management books and has expressed the same concerns regarding quantifying IT value that many of us in the IT business have felt for some time. He is a refreshing author and his many books, including Information Payoff, McMillan, 1985, have helped convince me and others that we have been right all along.

Generic Value of Computers

There is no question that in the 21st century, operations in most large corporations would rapidly grind to a halt if their computers ceased to function. The sustenance of all advanced information-oriented societies now rests on the proper functioning of small and large processors that control everything from electric power, telecommunications, financial services, and energy-supply enterprises. We are quite vulnerable to a deliberate attack on the very software that operates our information infrastructure in the form of information warfare or hackers with a mission.

Large and small servers and desktop PCs are only tools. Though blessings do come forth from these tools, they are not unqualified. I have seen seemingly identical machines with identical software performing admirably in one company, yet when deployed in another organization with say, inferior management or not as well-trained personnel, they actually make things worse. Certainly computers enhance sound business practices, but they also intensify inefficiencies whenever the user community is disorganized and unresponsive to customers' needs.

Ironically, the best computer technologies will always add unnecessary costs to a poorly managed firm. Of course, the problem is not the inherent capabilities of the technologies, which are in a word, overwhelming, but with management's inability to use the tools effectively. For instance, there have been various studies by think tanks that contend that as much as seventy percent of IT projects have not delivered their expected benefits. A major cause of the failings is documented that the organization has been unsuccessful in integrating the results of their efforts into day to day work processes. In reviewing these findings a number of top corporate executives share the same opinion.

CEOs and COOs and even CIOs complain that there is most often no correlation between IT expenditures and corporate profits. Yet, sometimes in some companies there is. How can this be? Though "all men are created equal," the human condition permits and delivers broad variances in our performance in given areas. If the machines are the same and the software is the same, then the problem is with the beings with the feet and hands and arms and legs, starting with management and working down.

Douglas McGregor's Theory X and Theory Y models of management style suggest that there are some managers who trust and give general direction and there are others who have none to little trust and they give micro-directions. Business productivity has roots in well organized, well motivated, and knowledgeable people who understand what to do with all of the information that shows up on their computer screens. This would be a Theory Y type management scenario. Such excellence does not prevail so frequently in Theory X businesses and that may explain why in a number of companies, there is no correlation between IT expenditures and profit. In those companies, it is unrealistic to expect that computerization could ever change that.

Prove the Relationship

In Theory Y organizations, business executives as well as computer experts typically recognize that the fortunes of the enterprise originate with the people who administer, coordinate, and manage employees, suppliers, and customers. Let's say that on the average, the cost of computerization equals less than one-fiftieth of revenues (<2%). Therefore, it does not make sense for top management to demand that the IT Manager prove how computer budgets relate directly to profits. The best that the implementation of a fully functional computer system can provide is to make the knowledge workers more effective, and sometimes more efficient – whether there is a correlation to the bottom line or not.

The experts suggest that this relationship between corporate profitability and computer spending has been like this for quite some time. It is not a recent phenomenon. From this, it is easy to conclude that it is unlikely that any direct relationship between computerization and profitability will magically appear in the future. Computers are only tools for change, hopefully for the better. However, observation shows that identically trained people in different organizations can come to opposite conclusions from an examination of data obtained by identical means. What matters then is not the provision of information on a computer screen. Good software can do that. What matters are the knowledgeable actions they take with the information they are given.

There is no question that all computer systems if deployed properly have a great potential to provide information. However, because of the human condition, managers may very well misuse that potential. Thus, one might conclude that the effective and profitable use of information technology does not begin with a better understanding of hardware or software; it comes from knowledge workers having a better understanding of their respective organizations, its goals, and strategies.

As a concluding thought on the business value of IT, it is still propitious to align IT with the business. It does not matter what technology is in play. Once aligned, the measurements are not so simple. You can forget about productivity, improved customer satisfaction and quality as IT measurements. The way to measure IT's alignment with business goals is to gauge IT's impact on the one metric that matters most to CEOs and shareholders: net cash flow. The bottom line is that alignment comes down to accounting. It's that simple.

The Feature du Jour Approach to Computer Selection

A risk in the deployment of IT is the notion of the system or feature du jour. There are systems out there, and you probably know of them that change their features every couple years and then by pulling support or by psychologically swaying the masses that their old wares are inadequate for today, they get to sell the same thing, new and improved, again and most often for more money. Most businesses are not in the computer business and do not want to be in the computer business so they rely on a team of inside and outside consultants who have been certified to protect the business opportunities of the computer company. This certified team is not certified to find the best solution for the company however. They know one thing and the one thing they know is what the company ultimately buys.

What often happens in these ad hoc scenarios is that companies end up with a proliferation (mish-mash) of incompatible systems that rapidly grow obsolete as the business or organization changes. Strassmann calls these the 'build and junk' solutions. In these situations, there is often not any room for new thought because the pattern of computing, successful or not, dependable or not, has been in place for some time and the voices supporting that equipment, the change brokers in the organization, actually do not want to change themselves. Thus a truly innovative and affordable solution – software and hardware -- would be left on the table because it would not be compatible with the current mindset of the firm's IT advisors. In many ways, that is why the i5 all-everything machine is not so well known in many small to medium sized businesses. It does not matter how good it is, nobody wants to hear about it - even the very IT advisors on whom the organization depends.

In addition to the mindset that espouses the short term "build and junk" solutions that continually patch one deficiency and create another, there is a similar mindset with software function that has been delivering its

payday for years without issues. Because anything that has been running on a computer for more than five years can be disparaged as "legacy" by the young Turks who often provide the prevailing thought in small to medium sized businesses, companies often find themselves pushed to eliminate the old and move on with the new, just because it is "new.".

More often than not, new means Microsoft and anything else is old, though Windows roots are well over 20 years old. Despite the pressure to replace, there is hard evidence that older applications and platforms still work fine and it is not hard to find them providing value every day in most organizations. However, if you will pardon me, it is not politically correct. Coincidentally, software built for the great grandfather of the IBM i5 just under thirty years ago still runs on today's all-everything machine. And, believe it or not, it is difficult to convince some people that this is an advantage, not a disadvantage, no matter who is doing the talking.

Business Value

The chanting by industry analysts for years not to worry because computers deliver competitive gains, speed up business transactions, increase customer satisfaction, deliver superior quality, and lead to improved profitability have become generally accepted wisdom. But sometimes, if the applications are not hosted on the right servers, regardless of the quantity or quality of the chanting, the benefits are never realized. Actually gaining the benefits from your server is not a given.

The question as to whether IT provides business value as noted in the prior discussion has spawned much activity in management circles recently. The question is not how much return on investment for projects, and especially information technology projects, is provided by IT but

whether there is any ROI at all for IT efforts. Many of the managers and academicians and analysts who have offered their thoughts on the subject seem to have concluded that as necessary as it is, IT implementations often do not add to business value in any meaningful way.

Strassmann affirms this thinking. The theory goes that as IT analysts and technicians streamline a given area of the firm using technology, a significant portion of the ROI, if the project is successful, comes about because the productivity cogs of the former system have been eliminated, and these could theoretically have been eliminated without the use of technology.

I do not share this doom and gloom view of the inherent value of IT. However, there are very many poor implementations in businesses for many different reasons. In my own backyard, I have observed companies and organizations in which managers could not accept that a desktop PC was not intended to be the IT panacea server for any firm. At about $1500 per box, that is a pretty nice notion to think that such a small investment is going to bring home huge technology benefits. But it never happens, regardless of the number of 'servers' the company buys.

I tell my clients and my students and you will hear it in this book, "The system makes a difference." And I also tell them "Not all computer systems are created equal." You'd think that they would already know that but the fact is some just don't. Today very few people in my industry even use the word "system" to refer to the computers that are used to run the business from the back rooms. They are all *servers*. A system however is much more than a server. A system in its most simple definition is a group of interrelated parts working together as a whole. An IBM i5 is a system and a very capable server. Any other server, especially a Windows server is merely a component in a system. However, such a server, or even a desktop client unit masquerading as a server, is often sold as a do-it-all server. My experience is that all is well in this

environment until you want to do something else – then you need a second one. Then, a third… Soon, you've bought the farm.

The Other Side of the Mountain

So, we might conclude from this reading that there is business value to be gained from IT investments. However, without a hefty fee, even Lloyds of London will not assure that any value will be realized. Nor can they!

In this section, we have learned that the management of the organization and their expressed desire to succeed in IT projects has a major bearing on IT success. We have also learned about Strassmann's notion of 'build and junk' solutions. Additionally, we learned that there are times when the IT professionals in an organization have more important things on their agenda than the welfare of the firm. Unfortunately, they may not even know it. A simple self-test for these IT folks would be if every decision they make favors their personal certifications.

Please know that I am not trying to cast aspersions on the character of IT personnel. However, I am a believer in the philosophy as George Patton once said that "when everyone is thinking the same thing, somebody is not thinking!" I submit that many of the Windows certified experts, systems programmers in my personal vernacular, remind me of the little boy who never saw the other side of the mountain. Because he liked the side of the mountain he was on, he concluded (imagined) that the other side of the mountain was ugly and barren and not worthy of even visiting. Yet, he had never seen that side of the mountain.

I run my business on two desktop PCs. One is backup for the other. I would love to have a business able to afford an IBM i5 because I know the i5 and I know how much better life could be for me. But, I live on the leeward side of the IT mountain, thankful for every macro Excel provides. I too use the Microsoft style of computing for the simple things necessary to run my business. And, of course I have helped my clients install Windows servers both inside the i5 complex and externally. Though I know where I want to live, I do feel I know both sides of the mountain.

There are many of my peers who stay on the Microsoft side of the mountain or the Unix side of the mountain. They know nothing about the other side of the mountain where my IBM i5 clients live. They have concluded however, just as the boy on the "good side" of the mountain, that there is no reason to even know what is on the other side. Because they have already thought it through and because neither IBM nor the Windows dominated press gives them a reason to look any further than Redmond Washington for the good of their organization, they choose not to look at the IBM i5 as a possible solution to so many ills that their company may face. Thus, in most Windows shops, the demand for funding is for more servers, faster servers, and more people to support the servers. Obviously, for them, just like the boy on the good side of the mountain, there is no other way.

However stacked the deck may be in favor of Microsoft and Intel in most IT shops today, I would not be telling the full truth if I ignored the fact that this results from there being no compelling reason to look at an IBM i5 as a real business solution for small businesses. Most businesses who should be driving their IT shop with an i5 have never heard of the IBM i5 or even its predecessor, the AS/400. The "uninformed" Microsoft certified IT staff sees no value in messing up the mix by looking at non Microsoft servers even if there may be the possibility for management to better realize the rewards of their investments.

Quite frankly, I can't blame the Windows certified professionals out here. They really don't know that there is better water to carry than Windows if that is their only game. Again, that's why I wrote this book. I expect and to a lesser degree hope that the Microsoft side will want to know about the all-everything machine so that they can advise their management that there is more out there than what they are accustomed to… and it may be lots better.

The system actually does make a difference.

ERP Provides Business Value

The business value factors and the technical factors that we are about to discuss and which are highlighted in this book differentiate an i5 from all other machines. It is no wonder that the IBM i5 is the predominant system used for ERP. It is the best system and ERP is the defining business application for most companies. It is the all-everything application and it is not too coincidental that the most implementations and the most successful ERP implementations run on the IBM eServer i5 family of computers.

Note: What is ERP?-- Enterprise Resource Planning is software that provides a business management system as a solution that integrates all facets of the business, including planning, manufacturing, sales, and marketing. As the ERP methodology has become more popular, software applications have emerged to help business managers implement ERP in business activities such as inventory control, order tracking, customer service, finance and human resources. IBM's i5 systems are the industry leaders in providing ERP solutions to small to medium sized businesses.

ERP is now being hailed as a foundation for the integration of organization-wide information systems. ERP systems link together entire organization's operations such as accounting, finance, human resources, manufacturing and distribution, and more. Moreover, they also connect the organization to its customers and suppliers through the different stages of the product or the process life cycle.

ERP systems come with many modules. However, the most significant modules, where the majority of business value is achieved are as follows:

1. Inventory Management
2. Order Entry
3. Pricing Flexibility
4. Purchasing
5. Production management

Besides all the benefits of the individual modules, and despite how a given company does business, the overall benefit attributed to an ERP package is the connectivity of information. In other words, the modules, when deployed are integrated such that the output of one module - order entry for example, feeds many others, such as billing, inventory control, accounts receivable, and sales applications. There are no rough edges. Each module knows how to "talk" to each other module, and the modules understand each other. That's integration and there is a whole lot of business value to that notion alone.

The Benefits of ERP and the i5

In addressing the notion of the business value of a computer system (server), it makes sense to see what software that server is running. Since most companies who automate do so to help their business run more smoothly, the typical business applications such as order entry, billing, account receivable, etc. are first to be implemented. This is the way it is regardless of whether the applications are part of a big ERP system or not. Therefore, we can say that the business value of any computer system is the value provided by its applications, such as ERP. So, rather than begin a discussion about system oriented features that provide business value, we can simply use the benefits of ERP systems as our guide to business value. After all, it is the combination of the ERP system and the i5 that bring those business benefits home.

Four generic objectives that companies have, who implement ERP, are as follows:

1. To improve responsibilities in relation to customers
2. To strength supply chain partnerships
3. To enhance organizational flexibility
4. To improve decision-making capabilities

From these objectives, companies have more specific motivations. Though these motivations do not equate to hard dollars, the most

common generic reasons for which businesses implement ERP are as follows:

1. Need for common platform, (such as an i5) with the intent to replace innumerable smaller systems (such as Windows servers).
2. Process improvement expected from the implementation
3. Data visibility that could be used to improve operating decisions
4. Operational cost reductions
5. Increased customer responsiveness in operations
6. Improved strategic decision making

Moving down the chain of rationale, for implementation, there are five major and specific reasons why companies undertake ERP projects.

1. Integrate financial information
2. Integrate customer order information
3. Standardize and speed up manufacturing processes
4. Reduce inventory
5. Standardize HR information

Knowledge of the generic benefits to be gained by companies that have already implemented ERP systems is often the main reason that drives

other companies to an ERP implementation. These benefits include the following:

1. Improved Work Process
2. Better customer satisfaction
3. Better customer service
4. Fewer complaints
5. Better quality (less rework)
6. Increased access to data for business decision making
7. Increased control of work processes by staff
8. More timely information
9. Greater accuracy of information with detailed content.
10. Improved cost control
11. Improved customer response time
12. Efficient cash collection
13. Quicker response to market conditions
14. Improved competitive advantage
15. Improved supply-demand link
16. Integration with eBusiness

When a company completes an ERP implementation with an i5, after the startup issues are resolved, the benefits begin to accrue. Benefits are in many different areas since ERP is so far reaching as an integrated application set. There are way too many applications and their associated benefits to list in this book. However, the major benefits that add to the business value in the operations and financial areas are as follows:

Operational

1. Reduction in inventories
2. More inventory Turns
3. Lower carrying costs
4. Reduction in total logistics cost
5. Fewer stockouts
6. More efficient picking
7. Reduction in manufacturing cost
8. Reduction in outside warehousing
9. Reduction in procurement cost
10. Increased production capacity
11. Improved order cycle time/ accuracy/cost.

Financial

1. Increased shareholder value
2. Reduced assets deployed
3. Increase return on equity
4. Improved cash flow

Now, we are talking. Business managers understand those things that add value by increasing profits, whether they manifest themselves as opportunities to gain more business or they manifest themselves in lower cost through operational and financial efficiencies. When these benefits are quantified, they become a real value that is added to the firm. But, with an IBM i5 all-everything machine, that's just the beginning

The ERP application benefits can be accrued on any computer system but because of the large system function and ease of use characteristics of an i5, any software project is more likely to be successful and it is more likely to cost less than on any other platform. It's also a fact. Surveys show that IBM i5 ERP implementations are completed significantly sooner than those on other systems.

Whether the application is ERP or CRM (Customer Relationship Management) or SCM (Supply Chain Management) or eBusiness, or simply Human Resources or Payroll, the i5 itself adds additional value to the business. This value does not come from the software. It comes from running the software on an IBM i5. Besides the list I am about to show, one of the most well-known aspects of the i5 is that its development tools help programmers and implementers get new work up and running quickly and they help the team maintain existing work in a productive fashion. My experience is that even with a packaged ERP solution, one of the biggest software libraries on a well-used business system is the "change library." There will always be changes and lots of changes over time. With an i5, it is a documented fact that you can develop applications or change applications five to ten times faster than on any other system.

If applications can be completed quicker, then their benefits are obviously accrued faster, and the firm benefits from the better method sooner, not later. Moreover, because it is finished sooner, it costs less to build. So, benefits more quickly roll in and costs are reduced when an all-everything machine is in play.

I5 or a Server Farm – You Make the Call

Another of the biggest values that an i5 adds to the business is that it can run the whole business on just one machine, thereby saving both hardware and implementation dollars as well as the support personnel that would have been required for the server farm. The next biggest value that an i5 supplies is that it just does not go down. And, downtime can be an extremely costly factor to a business depending on technology to survive.

Downtime is one of the main costs that should be taken into consideration during a server and software evaluation. An average ERP implementation for example, on a non i5 server would experience 2.8 hours of unscheduled downtime per week and according to a recent survey of 250 Fortune 1000 companies, industry analysts have reported that the average per minute cost of downtime for an enterprise application is as high as $13000. Considering that an i5 has a yearly average downtime of just over five hours, that's a lot of money to risk by not using an i5.

Dennis Grimes, CIO of Klein Wholesale Distributors, the fifth largest candy and tobacco wholesaler in the US explains it this way:

"There is a tremendous time savings because the system does not go down and force us to scramble to get our orders out and our work done. There is virtually no system down time, no restarts, and no calls at night or weekends. Applications just run and run and run. Forget its there! No time spent on getting things to run right. The machine is self optimizing.

We have Windows servers also and the i5 has them beat by far on economies of scale: It can run many things without choking. I only need to manage one system. It is even easy for me to add capacity on demand.

We have our box on the Internet. Nothing is impregnable but this baby is tough to crack. I know of no other system that can't be hacked. Security is just part of the whole package. You just get it. The i5 doesn't have the open doors like other systems."

Being able to develop and maintain applications in short time frames and run multiple workloads on multiple operating systems on the same machine with just one processor (or 64 if you need them) along with always being available for action, are major business values for an i5. But, there are a ton more.

The following is a comprehensive but not exhaustive list of the added value that a company gets from running its ERP, CRM, HR, or any other application on an IBM eServer i5:

Factors that Add Business Value with an i5

1. I5 is designed for small to medium businesses, not as a toy for the desktop.
2. Business value of working with IBM as a trusted partner
3. Provides unsurpassed competitive edge
4. Best tangible ROI
5. Quickest investment recovery
6. Return on investment (ROI), typically in less than one year
7. Elimination of multiple, underutilized servers
8. Highest level of integration
9. Outstanding performance
10. Best Security – no hackers, no viruses
11. Runs core business applications and eBusiness on same machine

12. Deploy new applications quickly
13. Fastest ERP implementation
14. Highest customer satisfaction
15. Highest ERP customer satisfaction
16. Intuitive management tools
17. Fastest speed to market
18. Greatest business agility
19. Reduced complexity
20. Enables change quickly
21. Highest IT staff productivity
22. Reduced requirements for technical staff.
23. Reduction of technical and administrative costs.
24. Free, integrated relational database
25. Free, integrated transaction processor
26. Free packaged Web servlet server for eBusiness
27. Simplified IT infrastructure
28. Best usability characteristics (ease of use)
29. Highest user productivity and effectiveness.
30. Easiest, least costly implementation
31. Lowest cost of ownership
32. Non-disruptive business growth
33. Virtually unlimited growth
34. Seamless, streamlined upgrades
35. Long lasting software solutions
36. No need to buy new packaged software version with upgrades
37. Lower implementation time and costs.
38. Most dependable, flexible, affordable
39. Zero downtime (99 44/100% uptime)
40. Fewest unplanned outages
41. Simplified maintenance.
42. Best service team in the world (IBM)

Technology Value

To the technical team, the above list would appear to be fluff kind of things with little substance. Yet, there is a story behind each and every one of the business value factors that are shown in the above list. It is tough, however, to digest that whole list, and it is tougher to believe that there are actually many more items that can be added to the list. Yet, there are.

The above business value factors are achievable, however, because of what IBM built into the i5. There may be a commercial machine out there that has implemented one or several of the below features of the system, but no other system has more than a few. The technical factors that bring the business value factors to the forefront are listed below. Please note that this is not a complete list but it is pretty large in its incompleteness.

iSeries Technical Factors

1. Implemented IBM's FS (Future System) technology
2. Most advanced computer science technology facility in the Industry
3. Ninth generation of 64-bit RISC computing
4. Advanced autonomic computing
5. 30 year old software runs without recompilation
6. Manages up to thousands of disk drives as one
7. No need for C,D,E,F drives
8. DB file placement auto-optimized for performance
9. Allocates file space as needed on multiple drives
10. No need to move or split files on different drives
11. Provides internal SAN for multiple OS environments
12. High Level Machine (hardware abstraction)
13. Technology Independent Machine Interface (TIMI)

14. Self-generating self-adapting object code
15. No recompiles on migrations from S/38, AS/400, iSeries, i5
16. Object based
17. Single level store
18. Capability based addressing
19. Integrated DB2 Universal Database
20. Pre-integrated database, middleware, and operating system
21. Automated database reorganization;
22. Integrated transaction processing
23. Tuxedo and CICS not needed
24. Runs many applications at one time
25. eBusiness and ERP on same server
26. Outstanding performance
27. Integrated performance collection
28. Integrated Apache HTTP in OS package
29. Standard WebSphere in OS package
30. Integrated dynamic workload management (self tuning)
31. Workload integrity
32. Integrated resource management
33. Integrated backup
34. Enable continuous operations with "hot site" failover
35. Runs up to four different operating systems concurrently
36. I5/OS, Unix, Linux. Windows on one i5
37. Integrated resource virtualization
38. Integrated security facilities
39. Virtual high-band integrated network
40. Share single physical storage pool
41. Multiple subsystems
42. Resource balancing (automatic and manual)
43. Continuous 24 X 7 operations – no disk defrags needed
44. Share resources and maximize central processing unit (CPU) utilization
45. Uses IBM Virtualization Engine
46. Increases server utilization rates
47. Logical partitioning (Up to 10 partitions per processor)
48. Built to be able to consolidate heterogeneous workloads
49. Advanced server consolidation
50. No assembler language needed
51. Programming independence from machine implementation and configuration details

52. High levels of integrity and authorization capability with minimal overhead
53. Efficient support in the machine for commonly used operations in control programming, compilers, and utilities
54. Self-generating, self-adapting object code based on technology independence
55. Efficient support in the machine for key system functional objectives, such as data base and dynamic multiprogramming
56. Underlying technology change does not translate into the need to recompile applications or disruption to the business.
57. Five to ten times programming advantage
58. Compilers are database and transaction processing aware (not an after thought)
59. Enhanced IT productivity
60. And more!

From my IBM experience, I am convinced that I would be able to deliver a several hour presentation about the i5 with just these topics. However, I would admit that more than likely it would just scratch the surface of the topics in the above technology list.

If you spent the time to burrow through this list, and you are a technical person, you are probably impressed with the i5 as a technical unit. There really is lots more to tell you though, and throughout the book, you will be exposed to more of the technical magic surrounding the i5. Because I have written this book so that a business person or a technician can read it, however, the level of detail in this book does not approach what you would find in a technical manual or a technical book.

Moving On

So, hang on, the plot has been revealed but the best is yet to come. Stay tune for a number of chapters that bring forth even more exciting goodies about the all-everything machine.

If I am a bit too superlative in my remarks for your taste, permit me to apologize in advance. I believe in what I say but I do not expect the reader to share all of my opinions. So, I hope you hang in there with me. Whether you are a business person, an i5 person, a Window person, a Unix person, or a mainframe person, there is lots in this book for you. No, you're not going to learn which bit to turn on in the PSW to make the system purr like a kitten, but you are going to learn about the computer science attributes that make the IBM i5 more of a machine than you have ever seen in the computer industry. And. If you can get through that, you'll learn how a system using those advanced attributes makes life better for the IT staff as well as the whole organization.

"The Eggplant That Ate Chicago"
by Norman Greenbaum (Dr. West's Medicine Show & Junk Band)

You'd better watch out for the eggplant that ate Chicago,
For he may eat your city soon.
You'd better watch out for the eggplant that ate Chicago,
If he's still hungry, the whole country's doomed.

He came from outer space, lookin' for somethin' to eat.
He landed in Chicago. He thought Chicago was a treat.
(It was sweet, it was just like suger)

You'd better watch out for the eggplant that ate Chicago,
For he may eat your city soon (wacka-do, wacka-do, wacka-do)
You'd better watch out for the eggplant that ate Chicago,
If he's still hungry, the whole country's doomed.

kazoo solo

He came from outer space, lookin' for somethin' to eat.
He landed in Chicago. He thought Chicago was a treat.
(It was sweet, it was just like suger)

You'd better watch out for the eggplant that ate Chicago,
For he may eat your city soon (wacka-do, wacka-do, wacka-do)
You'd better watch out for the eggplant that ate Chicago,
If he's still hungry, the whole country's doomed ("it's in trouble!")
If he's still hungry, the whole country's doomed

Chapter 2

Where Did the i5 Come From?

No Secrets Please

There is no better kept secret in the computer industry than the new i5 from IBM. Another secret of which most modern computerists are unaware is that IBM makes the finest, most architecturally elegant, most usable, most productive, and most affordable computer system of all time. That system is the iSeries i5, the all-everything machine, and though its birth was on May 4, 2004, its advanced underpinnings go back well over thirty years. That's an awful long time for any company to keep such a secret but my speculation is that today's IBM is getting ready to change all that and begin to focus on claiming the proceeds from the many years of advanced development that culminated in its new i5.

That's what this book is all about

Not only has IBM kept the secret but with the all-everything machine, it has kept the lead. That is noteworthy but not quite as noteworthy as the fact that the machine architecture that was conceived and delivered just

less than thirty years ago is still the best that anybody has ever built. Using 30-year old "nobody else can afford to build one" architecture, IBM continues its technology lead by far compared with all the other machines of today, including the mainframe.

One would have to conclude that IBM is about 30 years ahead of its competition and that's before you factor in that during the thirty years since the all-everything machine's conception, IBM has not stood still. Each and every year, more and more capability and facility has been built into the all-everything machine. Now, I am not suggesting that the i5 all-everything machine is 60 years ahead of the competition but that is where the math logically takes you.

If I had never worked with other computers, mainframes, 1130's, System/360 model 20s, Unix, PCs, etc. I probably would not have appreciated what a solid system the eServer i5 family has been right from the start. The Rochester Minnesota - built small business computer line from which the i5 was spawned was unusually easy to work with. In every other early computer platform, there were cryptic codes to decipher and continual puzzles to solve just to get the machine turned on. Programming was/is even worse.

Of them all, at least before I worked with Unix, I felt that the mainframe was the most cryptic of the cryptic. Technicians carried special green cards with codes and translations galore in order to program properly on a mainframe. At the time I learned it, I was convinced that the mainframe had been slapped together by bit head engineers who expected just bit head engineers to work with it. Real people need not apply. Even today I have great respect for the technical acumen of the professionals who know the mainframe.

When IBM introduced the first ancestor of today's eServer i5 as the System/3 in 1969, it was remarkable. It was as if IBM had sent all the geeks home that day. There were no strange codes that were indecipherable. No IBM green "HEX" card was needed. Programming the System/3 was almost as easy as speaking in English. Maybe not that easy; but It was easy. IBM had succeeded in using high tech engineers to build a system for regular people. I don't know how they did it, but they did.

It was just a start, but it was a good start. From that moment on, the IBM Rochester style of computing became contagious. Rochester wares were the most popular computers in small businesses for decades. Each and every Rochester computer was built on the principle of large system function with small business system ease of use. Each model was substantially better than the preceding machine and IBM business customers just gobbled them up and their businesses grew unimpeded by technology and reboots.

Today, the i5 is positioned to be sold in small businesses to medium sized businesses to the largest businesses in the world. As a family of computers, with various sized models and various costs, it handles workloads from the size of just bigger than mom and pop organizations to the Fortune 500. IBM has recently labeled its i5 a "mainframe for the masses" because it gets as big as a mainframe but it can be used effectively by a small business.

This book walks you through the story of the i5 from the very beginning until today. In addition to telling a powerful, compelling story, the book describes in layman's terms the technology and computer architecture innovations that are part of every i5. When you finish this book, you will understand why IBM is proud to have built the finest computer system in the world, and you may just find a place for a particular size one of these rascals in your own business.

For the most part, this book reads as a series of twenty essays. Each of twenty chapters is built as a short story unto itself, with the sum of the chapters telling the story of the all-everything machine. For the most part, you can pick up any chapter and read it without having to read a prior chapter. However, you may want to read these early chapters first to get a perspective on what the i5 computer is all about and its relevance in IBM history.

This book presents the IBM all-everything machine, its underlying superiority, its rapid customer acceptance, the IBM development history, and the IBM all-everything machine's probable future starting with the i5. This is not meant to be a technical book at a detailed level. It is written for those who have some or little technical background, who may know lots or nothing about an eServer i5 machine or its predecessors. However, there are a few chapters in which I do get just a little bit technical, hoping that I can show the reader in reasonably simple terms how the i5 is a special machine with a long and successful tradition.

When you finish reading this book, regardless of your technical competency, you will have a good idea of a number of unique computer science architectural attributes from which any computer system, from any vendor, can benefit. You will also understand how those attributes can help any company, such as yours, preserve its software investment and permit the upgrading of hardware and software without forcing a rewrite or a re-build, or a re-purchase. You will learn that not only has no other computer company, of software or hardware heritage, ever created a machine with all of these advanced architectural attributes, no computer company has yet to be able to adopt even one of these powerful notions into their computer servers of today.

This book is written then to teach you what is unique about an i5, and why the parts that are unique, are also good, not bad; and why you should

demand these facilities in any machine you choose to use. As noted above, I believe that the computer system (server) actually does make a difference in the overall value of IT to your business, and there is no system that has ever been made that delivers value better than the eServer i5. In this book, you will learn why!

Chapter 2 Appendix:

Twenty Questions

There Could Be a Lot More

When I was first trying to create a compelling Chapter 1 to help the reader gain interest in this book right from the beginning, I started to ask myself a number of questions. These are the questions I would ask somebody who was suffering from any of a number of IT maladies prevalent in non IBM i5 IT shops. The maladies include "no perceived

business value disease," "system down disease," "where's my information disease," and of course the killer, "Microsoft myopia staff disease."

These questions are not subtle, and for the most part, they are answerable by a simple yes or simple no. In each case, however, a situation is portrayed that (a) you either do not have with an IBM i5 IT environment or (b) you can have only with an IBM i5 IT environment. The list of questions is not exhaustive but there are enough to give keep you busy in a very productive exercise if you have the time.

So, without further ado, here are the twenty questions plus a few more:

Business Value Questions:

1. Are you suffering from more customer complaints because your customer, product, inventory, and shipping information are not available to your customers when they want it and the way they want it?

2. Are you losing customers because your systems are not available or are not accommodating when your customers need information or responses?

3. Would you like to be able to reduce the breadth of knowledge that you need or would need to keep your IT infrastructure up and running?

4. Would you like never to hear (again for some) those words, "the server can't do any more. We need another server, and another…"?

5. Would you prefer to get your IT work done without a major hardware and human resources investment in a server farm?

6. Would you like to be able to contain and manage the cost and the increasing complexity of your IT deployments rather than be forced to add the next server, and the next, and the next?

7. Would you like to be able to reduce your required IT people skill level and cost and not require so many high priced IT staffers just to have your server(s) operational and ready for work?

8. Would you feel better about your IT investment if you did not need a plethora of skills just to keep your server(s) up and running?

9. Would you like your IT staff or existing person in the organization (depending on your business size) to be able to perform their IT related jobs with more flexibly and with less essential knowledge pigeonholed in individual staffers?

10. Would you like to be able to reduce (perhaps to one) the number of boxes and operating systems, and database packages and achieve the requisite savings in IT personnel costs?

11. Would you like your business database to be there when you need it for every transaction and every query?

12. Would you like to be able to have a comprehensive, information-laden database without the requirement for a high-priced database administrator?

13. If you already have a computer server that has not quite measured up, would you like to get it right this time, rather than hearing a bunch of sales pitches each ending with, "Of course it will do that" when, in fact, it cannot?

14. Would you like to get out of an environment where you need a new server and a backup server for every new application or new function that you need to run your business?

15. Would you like to not have to pay for the associated increase in server support people, to take care of your growing number of servers?

16. Wouldn't it be nice if there were one server that without breaking the bank, was able to absorb all of the work from all of the other servers and grow with you from just a few to several hundred to several hundred thousand users – without having to scrap the machine, add servers, or start over?

17. Would you like to have an all-in-one all-everything machine solution designed to address the business, technical, and financial pressures faced

by all small to medium sized businesses, rather than an IT environment that creates more pressure than it relieves?

18. Would you like to have a server with security and management capabilities that is a direct descendant of mainframe offerings with a long history in the marketplace?

19. Would you like a server that was designed and built with the facility and the agility to provide your firm a means to secure revenue opportunities that might otherwise be unavailable or technically problematic in a world with small Windows, lots of hackers, and limited support people ?

20. Would you like a server that is not subject to intruders, hackers, spyware or the infamous virus du jour?

21. Would you like to have a server platform in which your software does not have to re-written or re-purchased every five years because the new server or the new operating system line can't run it, or can't run it at full speed?

21. Would you like to hear "yes" when you ask your IT staff if your server has the ability to handle high workloads and data processing chores that offer your company (and other small and mid-sized firms) the technology needed to seamlessly work with robust enterprise computing environments at a fraction of the cost, even though your business is not gigantic and your pocketbook has limits.

22. Would you like to have an IT environment that lets you live comfortably like the big guys live without having to pay big guy prices?

23. Would you like to have a server built by a company that knows that smaller and mid-sized companies have concerns and needs that are unlike their larger cousins, because they live with constrictions and limitations on the small servers that are not usually found running larger enterprises?

24. Would you like a server that can provide you large enterprise function with small system ease of use and small system cost?

25. Is it upsetting to you that the business-critical nature of technology for the SMB market mirrors the IT reliance of larger enterprises, yet so far your IT tools have fallen far short of doing the job and providing business value?

26. Does it bother you that SMB companies such as yours must deal with similar issues of IT complexity, yet are challenged to find a way of achieving success with the economies of scale issues in the small multi-server IT environments?

27. Have you been forced to say no to important IT projects that can grow your business because at a hypothetical $70,000 annual cost for a single IT staff member, it has become clear that IT growth, despite its potential long-term competitive advantage, is simply beyond the reach of your company as well as many other small and mid-sized firms?

28. Would you like to have a server about which IBM, the leader in server technology says: "IBM eServer iSeries is a premier business server designed to help you to improve productivity while reducing costs and complexity?"

29. Would you like a server that can achieve significant cost savings for your organization either by never needing a server farm or by consolidating the industry-standard Intel servers running Microsoft Windows and / or Linux onto one server?

30. Would it not be great if the data center architecture enabled a consolidation server, such as an all-everything machine (IBM i5) to run multiple operating systems in series, i.e., first as a Windows server, then as a Linux server, etc.? How about all at the same time?

31. Do you find it a challenge for integrating business functions in the typical server environment that requires the execution of applications running different operating systems in parallel on many different servers?

32. Can you see how it would save lots of additional systems and thus lots of money to run all integrated business functions on one integrated system such as on an all-everything machine (IBM i5) that permits many operating systems to run on just one machine.

33. Would you like to have a system that can run Linux, Windows, Unix, and OS/400 under one set of covers with support for NetServer using virtual Ethernet and Microsoft Peer Networking as well as Samba enabling cross talk between operating systems under the same set of hardware covers?

34. Would you like your organization to benefit from unprecedented levels of reliability, scalability, and a high level of system integration?

35. Would you like additional savings to come from reducing system administration head count and avoiding the operational costs associated with server downtime?

Technical Questions:

36. Would you be able to achieve additional productivity with a system that provides its own virtualized storage area network, supports multiple file systems and multiple operating systems over the same disk storage?

37. Would you like a server that is programmable in both computer science languages, C, C++, Java, as well as business languages such as COBOL and RPG IV?

38. Would you like to work in a transaction processing environment that enables interactive and Web programs to be developed in 1/5 to 1/10 of the time of conventional systems?

39. Do you want to say no to disk fragmentation and reorgs?

40. Do you want to say no to ever running out of space on one disk while the system has many empty disks?

41. Do you want to say no to rewriting applications and splitting disk files because you, not the system, must manage disk space utilization?

42. Do you want to be able to migrate your software applications when necessary to the next generation of computing without having to scrap them, rewrite them or even recompile them?

43. Do you want to say no to placing files on specific disks and specific locations for performance reasons?

44. Do you want to spend time typing data definitions into your programs when i5 programming languages can bring in the data descriptions from the database automatically?

45. Do you want a server that provides everything that you can run on a PC without having to worry about having to do the CTRL-ALT-DELETE dance or deal with virus attacks?

46. Would you like a machine with a documented average up-time of 99.98%?

47. Would you like to have a machine that can easily convert from the older technology, such as 48-bit CISC hardware to newer technology such as 64-bit RISC without having to re-compile your programs?

48. Would you like to be able to perform Concurrent Maintenance on your system without having to bring it down?

49. Would you like to be able to backup your system while it is active? In other words, would you like to be able to preserve data and programs without having to perform a shut down of your server to do your backup?

50. Would you like to bring data down naturally from the server to MS Excel and other applications from one or more DB2 Universal databases using ODBC, SQL or OS/400s built in query and SQL?

51. Would you like to have up to 60 Windows NT4.0/2000/2003 servers, controlled and administered by one server rather than a farm of independently supported Wintel boxes?

52. Would you like to be able to carve out up to ten partitions (each treating the server as one whole machine) on a one processor server?

53. Without purchasing expensive virtualization software, would you like to run with virtualization always on, providing the highest possible utilization of your computer resources?

54. Would you like to be able to tune and auto tune the operating system in ways that are impossible with Windows and Unix boxes?

55. Would you like programmers to be able to develop new applications or change existing applications 5X to 10 X faster on your server?

And the Answer Is

Of course, the answer is that most business managers want a computer that provides productivity and efficiency and results without pain. Quite frankly, technical people aren't really interested in hurting themselves to get a computer job done either. Getting major business value from your production IT server should be easy and it is easy with the all-everything machine, the IBM i5.

Chapter 3

Voices of Users, Analysts, and Industry Experts:

Users Know Best!

There is nobody who knows the value of an i5 better than somebody who uses it day in and day out. So, rather than continue with twenty questions or get into the technical detail of the machine, I thought it would be a good idea to round up some of the good thoughts of i5 users, analysts, and industry experts from across the country. This assemblage of spokespersons for the IBM i5 does so of their own free will because they have a story to tell that they believe it is worth hearing.

I asked each to provide me with one to two pages. As you will see, some comments are shorter than a page and a few are a bit longer than two pages.

Most of the analysts, consultants, industry experts, and even IBMers have a background in working with i5 customers and thus their point of view represents observations of i5 family customers in action over the years. So as not to leave the reader with just the voices of the pundits, however, I went half way across the country to get a perspective from a bone fide user who happens to have experience with two different i5 family machines in his home town.

The writings of the individuals in this fine group are immediately below. The format of the rest of this chapter then will be to highlight the name of the person, followed by their story in their words. At the end of the stories, there is a short biography of each of the writers. I hope you enjoy

their musings and I hope that it gives you a real perspective on where this i5 machine came from, and what a fine machine it continues to be.

Jim Sloan, Jim Sloan, Inc.

JS: "I knew the System/38 when it was just a piece of paper. It was amazing in its conception but it seemed terribly slow in developing. Major IBM development managers fended off the IBM Company just to be able to produce the product. That was a terrific political success though from first hand knowledge, I know it was very difficult to pull off.

I must say that with all that we put in the machine, it was incredible that it worked as well as it did. The fact that something so large with so many different players (hardware, software, support etc) can come together is a tribute to good management and lots of effort.

The biggest problem that the system had from the get-go was that it was underpowered hardware-wise. Making up for the lack of hardware power on the early System/38 was a major accomplishment. Of course with the AS/400 and now the i5, all the power issues have been fixed.

I am in the development area and so I don't have customer testimonials or customer war stories to share but this was and is a terrific system. Ironically, after spending so much to make the system work, IBM tried to kill it. And I don't mean just once or twice. In the end, each time, customers saved the product. They would come back to IBM and just not let the company discontinue a system that was so vital to their business."

BK: "How do you see the product now and into the future?"

JS: "It is one heck of a good product. It is a terrific product with terrific acceptance but for some reason IBM just does not market it aggressively. I do not know why they don't market it. They just don't. I would hope that changes and IBM highlights the system once again."

BK: "Though I don't share this opinion, there are some folks in the industry who say that Windows has taken over and even if IBM chose to go after small businesses as it once did with System/32, System/34, and System/36, it is probably too late for the i5 product line to make an impact."

JS: "It is never too late if IBM chose to market the machine as it should. It would be successful indeed.

The box has been good to me in many ways and I sure have had a good time working with it. I have been working on this for over thirty years and it has been wonderful to me. Considering that I worked with it from when it was just a piece of paper, that's a long time."

Skip Marchesani, Custom Systems Corp

"Sure, I can tell you the most outstanding attribute of the iSeries and AS/400. It has rock solid reliability and availability, unsurpassed in the industry, and there are systems out there that have run non-stop 24 by 7, 366 days a year, for years at a time. More and more shops are noticing that when their other servers are misbehaving and failing, the iSeries or AS/400 [i5] continues doing its thing every day, day in and day out."

" Years ago when the AS/400 first came out, a large national insurance company installed about 10,000 of the smallest models in their remote sales offices all across the US. About every three years, a systems technician would visit each remote office to check on and do maintenance on the AS/400. On one particular visit to one of the remote offices the systems technician asked to see the 'office computer'. The office manager showed him a PC sitting on a desk. The tech explained that the PC was just a workstation and he needed to see the system (AS/400) they were connected to. The office manager just shrugged and pointed to the other two PCs on desktops in the office. Finally, the system technician traced the twinax cable connection (wires from PC to AS/400) to a point where it went thru a wall in the back of the office. He asked what was behind the wall and got more shrugs. The entire office staff had turned over in the last year. He went next door and asked to see the wall next to this remote office but there was nothing to see. He went back into the remote office and knocked a hole in the wall with a hammer, and saw the AS/400 humming away in what the building superintendent said had formerly been a closet."

"I once had a conversation with a database manager for a large government facility on the west coast. This person who was in charge of all kinds of servers - Ingress, SQL Server, Sybase, Oracle, etc – and

had a total of five people on his staff counting himself. I asked him how much time the AS/400 database took to maintain. He said no more than 1/10 of one person. I asked him how much attention the other machines required. He started to ramble - or so I thought. He likened the AS/400 to a daycare child who comes in each day and you tell him what to do and he goes and does it and you don't see him again until the end of the day when he gets picked up. He likened the other kluge of databases to the hyperactive kids who get dropped off without their medicine. They are in your face, literally from the moment they arrive until they are dragged out at the end of the day."

"One time I was teaching a DB2/400 class and a student asked why Oracle DBAs make so much money. Before I could answer, another student volunteered that Oracle is such an inferior product that it takes a full time, highly skilled, highly paid DBA to keep it running."

"Oracle is good example of an inferior product with outstanding marketing. It's absolutely amazing that companies like Microsoft and Oracle can develop products that have very serious shortcomings, but their marketing is so outstanding that in spite of themselves they create a very loyal following. DB2/400 (aka DB2 UDB) is a functionally rich, standards compliant, object-based relational data base product. And, it just doesn't break! But, IBM's marketing is such that the industry is not aware of it."

Al Barsa Jr., Barsa Consulting Group

Note: Though this is the text of the audio taken from IBM Legends of iSeries # 0213, Al Barsa lived through this ordeal and was the "IBM Rep" in the story. The first paragraph here is the story as transcribed from the IBM videotape. This is followed by Al's personal comments about the incident.

"Like I was saying this IBM Rep shows up at this NY labor union to do a checkup on their IBM server. The thing is nobody knows where the server is! I mean nobody has ever backed it up. No one even knows if it is in the same building so they start tracking this cable. They go up one corridor, down another corridor, and they go round a lot of corners. I mean it's wrapped around… It goes up one floor. It even goes through this ventilation duct. Finally, the cable leads to a storage closet two floors away. The door is locked. Secretary says nobody has been in there in over six years. Somebody figures out that the super two buildings down might have a key. They get inside. It's like a blast furnace. It's so hot… the tech guy gets a nose bleed. Evidently, a power outage two years ago knocked out the AC. But the server [AS/400] in that closet rebooted, and got back to organizing and running that union [with no manual intervention and without anybody knowing]. Are you following this? Six years, no attention, no maintenance, and a 140 degrees virtual oven…"

Al Barsa has a few casual comments about the video:

"Look at this video. It's a fairly true story about me! In late 1999, I was doing last minute Y2K stuff at some of my accounts in NYC. While I was engaged in this process, one of my clients took me up on the offer and wanted to make sure their system had been prepared for the millennium.

So I showed up in my Brooks Brothers suit, no hat. The missing system in the story is absolutely true, and the super was from that building, not one or two buildings over, and he had a key ring that must have been 18" in diameter!

He found the key to the closet in no more than 30 seconds (much to my dismay).

The system was a B30 [old AS/400 model] that had gone through a blackout two years earlier, and rebooted because the system value QPWRRSTIPL had been changed to '1', but the air conditioning never recovered.

The story about me getting a nose-bleed is absolutely true."

Bob Warford, Labette Community College

Electrical Failure

"One night, the city of Parsons, KS, lost all its electrical power and when the batteries on our UPS got low, the IBM AS/400 shut down as it was supposed to.

When we came in to work the next morning, we found the IBM AS/400 was not running. This caused a lot of excitement. We could see that the lights on the control panel were on, but we could not figure out why the IBM AS/400 wasn't responding to the system console or workstations.

After a lot of looking and research, we finally gave up on trying to find the problem ourselves and called IBM Tech. support. The first question they asked us was "Did you press the white start button?"

Talk about feeling dumb. No, we hadn't pushed the white start button and yes, the IBM AS/400 came right up when we did.

To be fair to my staff and me though, because we are not dumb, we had never shut down the IBM AS/400 without instructing it to do an automatic restart and IPL so no one on my staff had ever seen the IBM AS/400 down. Not since the day it was installed.

To be truthful, no one on my staff had ever even started the IBM AS/400. The technician who installed it turned it on during installation and there was never a need to turn it off and there was never a time when it had had a problem that would take it down.

I think it is pretty impressive that a computer could run two years and never be down."

Six Days Down in Twenty-Five Years

"We (Labette Community College) bought our IBM System/34 in 1980 only because the IBM sales representative signed his name to provide several reports that our president at the time wanted. Although we were only buying a small IBM System/34 and a $500 student management system that didn't have the required reports in it, IBM fulfilled the sales representative's commitment and developed the reports for us. We had an IBM SE on campus most of the first year.

I believe the sales representative decided to work for someone else shortly after the sale was completed. I never heard what happened to his supervisor who also signed off on the reports.

When the college finally got a grant in 1986 that provided the funds to replace the IBM System/34 we migrated to an IBM System/36 because the IBM System/34 had only had one day in six years that it had been down. In addition, all our data and software migrated to the IBM System/36 without having to make any changes. It took one night to do the total migration.

In 1998, when the college got another grant, the decision was made to switch to an IBM AS/400 for the same reasons we had switched to the IBM System/36. In the twelve years we used the IBM System/36, we had only had three days we could not run and all of them had been in the last year and were problems relating to the diskette magazine drive. Actually we were able to run, we just couldn't backup.

All of our existing software also ported without any major problems. The only problems dealt with the IBM AS/400's library lists and duplicate program and menu names in the production libraries. The IBM AS/400 migration started at 4:30 P.M. on Friday and for all practical purposes was completed by 1:30 P.M. on Sunday. This included unpacking the computer and configuring all the workstations and printers.

The reliability and compatibility of the IBM System/34, IBM System/36, and IBM AS/400 has just been phenomenal.

To be honest, we did have a problem with the IBM AS/400 this fall. Something went wrong in the power supply and when the system did its scheduled shutdown and restart; it could not come back up. The technician who repaired it said he had never seen that problem before. That resulted in the AS/400 being down for two days while we waited for parts.

I would say that six days down in twenty-five years is pretty good. Although we did have other service calls in that time, there was none that prevented us from completing our work.

As far as software compatibility goes, what other computer system can say what I can about the IBM system. I still have a few of the original programs from the original student registration system that was purchased in 1980 running untouched. Although the IBM migration utilities recompiled the load members and we are still running some things under 36 emulation, we have not had a need to change the source code.

As you can tell, I like IBM AS/400s.

Someone really needs to help IBM do a better sales job on the IBM AS/400 [i5 marketing] because the IBM AS/400 is really a wonderful machine."

Doug Hart, Whitenack Consulting

"The System/38 was developed from the IBM "Future Systems" project. This heritage continues today with advanced OS features that continue to place this system at the front line of business systems.

For me being "old school", I still find the strength of the AS/400 line being the backbone of a companies computing platform. The integrated database, security, communications facilities give the system a consistent standard in which all the operating components work flawlessly. A business' primary applications (Accounting, HR, etc.) today must be available full time. The AS/400 with its 99.999% up-time rating gets the job done.

Today the i5 line using the Power 5 processors has outstanding performance. The systems are truly scalable from quite small to the most powerful of platforms. With advanced functionality such as Logical Partitioning (LPAR) the sophistication and state of the art capabilities of the line continue to lead the industry.

IBM's group in Rochester Minnesota that develops the system understands both their customer's needs and the future directions for computing. As I follow the evolution of the line I'm continually impressed with capabilities of the product."

Ken Anderson, Quadrant Software

"The IBM i5: The greatest business machine money can buy.

I first met Brian while attending the NY IBM users group. My company had been invited to present to the group on the benefits of Electronic Document Distribution in an iseries/i5 Enterprise. The interesting thing was that I showed up 2 hours early, before anyone had a chance to get there. I cordially asked the front desk where the user group meeting would be held. After she led me to the room, the only thing there was a copy of a new book Brian had written about IBM's relationship with the iSeries. (before the i5 was announced). (I'd recommend the book to any i5 shop I might add). I read about 50 pages and realized that we were on the same page. I think most i5 shops I meet think I'm too young to know anything about the i5, but once they hear me speak, they are amazed that the black box has made friends with some (not enough) in my generation as well. So, when Brian approached me to add a comment to this book, I jumped at the chance.

I don't want to tell any war stories. It's not that I don't enjoy hearing about the i5 that was sheet rocked into a wall and continued to run for 5 years, or Dr. Frank [Soltis] describing what a great customer Microsoft was on the platform and how they replaced a couple black boxes with lots of NT servers. I do. I love them. Rather, I want to describe how one mid twenties guy was converted and what I think needs to happen to convert EVERYONE else.

Because you see, it's not the decades experienced IT Director or CIO that is going to ensure that this box continues to run SMB's all over the world.

It's not the diehard programmer who came up on the 34, 36, and recollects using punch cards and tubes back in the day. Those people already love the i5. It's converting the people, like me, who learned right out of the gate you press start to turn off your PC.

First off I have to admit I was VERY skeptical the first time I saw the mean green screen. I remember thinking it looked like the computer James Bond used to look up spy information. You remember, back when it was still amazing that "M" had installed that phone in a car. I thought as most right out of college people entering the working world do. After all we are conditioned to believe that Windows is *the only* OS out there. Every program you are taught or use is NT based. People still use these? I thought.

But over the years I have had the opportunity to meet hundreds of i5 customers and talk to literally thousands of them over the phone. In every possible industry from manufacturing, healthcare, distribution, insurance, food, city government and even police stations, customers were using the platform for every conceivable computing purpose. And they were using it with half the staff and twice the reliability of anything else out there.

I guess I will tell one story. And this is just one of many I have that all begin and end the same. I have a friend/customer named Rick. Rick came aboard as IT Director at a division of an i5 shop where each division has the autonomy to choose what they want to run for applications etc. The business had significantly changed since the decision was made to bring in the 400 initially, and Rick was brought in as part of a new ERP project.

Rick hated the 400 right at the beginning because of all the reasons most folks in his position do. It seems expensive to buy apps, maintenance fees seem high, it seems old, etc. It really boils down to a simply learning something new. But Rick has something that I think is a prerequisite for anyone that does well in IT, an open mind. If one is intent on getting rid of the i5 for something else, they will and have. It's much more difficult if you approach it with an open mind. So he decided to allow i5-based apps, in with all the others.

He did painstaking tests on all of them. I remember him measuring how fast the order entry folks could enter an order in each candidate's

application (a nightmare for sales people like me I might add). An each time I spoke with him, he was little less harsh on the black box. Until, finally, a year later, they had made a decision. He chose an application I know only lives on the i5 family, and I flat out asked him how he arrived at that since it was no secret that he had no the great affinity for the i5 family.

Rick said, Ken, "I tried every possible justification, every ROI calculation, but they all came up the same. The total cost of ownership with this thing is simply lower than anything else I could get my hands on. I can run everything on the same machine. I can do multi-company, different languages, I can even partition. It runs email too." Rick wasn't ready to admit he'd been converted, but I knew, that was his way of saying he'd been wrong at the beginning.

And it's hundred of these types of stories that got me where I am today. And if I hadn't seen it with my own eyes or heard with my own ears I would have put up the wall and gone on thinking there is only one choice out there. And so would Rick.

But because of my job, I have the opportunity to see so many different kinds of businesses and how they operate. It is much more difficult for someone straight out of college to do the same thing. So how do we convince them? IBM can't do it. I think they need to learn it themselves. By exposing WHY you love the platform and really show them what this thing can do, it will happen on its own. Rick had to learn it on his own and so did I. The die-hards that simply crammed it down my throat could have never convinced me of anything other than, "they like it because it's all they know". It was only after real life examples and real ROI that I came to realize, if you are running a business vs. downloading .mp3 files, and surfing web pages, the i5 is simply the best business machine money can buy.

So I challenge any IT executive out there, to show the accounting folks how you arrived at that native iSeries payroll solution. Or the downtime figures of someone who chose the other. Or to bring the Jr. Programmer into the ERP selection process and show them how many less i5 servers you need to run 5 companies vs. how many you need with the "other" choices out there. I'm not in my mid-twenties anymore, but, I like to think I'm carrying the torch a little further than those before me can. And

if I can, I hope I'll be converting a few in the generation to take my place, along the way.

One last thing. While sitting at home the other night watching the latest primetime show, LOST, I almost fell out of my chair. It cut to commercial and one of IBM's latest campaigns came on. Although I love the new ad campaign it always annoys me that the i5 is never part of the puzzle. It's hard enough for i5 shops to get the budget to buy the new i5 they want or my products from Quadrant (hopefully you all will), without IBM highlighting every other server but no i5 on TV. There's Linux and global services, but, never anything on the do-everything machine. Then it happened. The best consolidation platform in the world was the message. It wasn't like the Sox winning the World Series or anything (I'm obviously from Boston). We have a long way to go for that. But, it was a little like the late inning rally when Boston was down to the last out and losing to the Yankees in the ninth inning of the 2004 ALCS. A little glimmer of hope. Maybe they are finally getting it, I thought. Getting what thousands of SMB's all over the world already know. The IBM i5 is simply the greatest business machine money can buy."

Dave Books, Former IBM Systems Engineer

"One of IBM's best kept secrets is the incredible reliability of the AS/400. I did some work with the Rollins Company here in Atlanta. They are the parent company of Orkin Pest Control, among others. Orkin has a small AS/400 in each of its four hundred plus branch offices. They're controlled from Atlanta. Critical information is downloaded to a large AS/400 here each night. Thus there's no need to back up the individual AS/400's at the branch level.

Last year I was talking to one of the support reps on the Orkin help desk. He told me about a call he got from an Orkin branch manager. As the manager described the problem he was having, the support rep became more and more convinced it was an AS/400 hardware problem. The support rep called IBM hardware support and they dispatched a CE to fix the problem.

When the CE arrived at the branch office, there was no one there who knew where the AS/400 was. It had been rocking along doing its job with no attention from anyone for so long; no one was currently working in the branch office who had been there when it was first installed. The IBM CE and the branch manager literally had to go around the office opening doors until they finally found the broken AS/400. Fortunately it was fixed quickly and was back in normal operation. The support rep who relayed this story to me thought it was an incredible testimony to the day-in, day-out dependability of the AS/400 family."

Bob Cancilla, Ignite/400

"There may be some concern and question about the future of the machine within IBM, but not about the machine. As you well know, the machine and its software gets better and better exponentially. Talk about the world's best kept secret!

IBM recently bragged about the big deal they did with eBay selling them AIX or Linux (non i5) based machines with WebSphere. It was a very huge sale. I think that IBM did the customer a giant disservice by not selling them on i5 based technology. The i5 could have reduced the staff and administrative nightmares that eBay must suffer from by an astronomical number! I would bet you could probably run the entire eBay network on three of the big i5's with total replication and redundancy creating an environment that would never fail. Furthermore the total environment could be managed by a handful of people.

But, IBM's Software Group sells WebSphere Server (WAS) by processor so they sold a lot of copies of WAS, ND, and machines and other supporting software and hardware and the CIO of eBay seems to love having a huge body count to administer his kingdom. Too bad the CEO wasn't aware of iSeries [i5]; she might have had a different opinion. "

Sr. Marketing Manager at IBM Software Group

A friend of mine (your author BK) who spent many years in iSeries activities had this to say in an email note to me just recently:

"A while back.... I was creating my own list of why I love iSeries [i5]… even though I have been away from that division for 4 years.

I looked up the OS vulnerabilities and OS/400 had only one recorded vulnerability (and it wasn't even on the OS/400 partition) vs. hundreds and hundreds on other operating systems. Check it out at: www.securityfocus.com

The IBM i5 platform uses I/O Processors (IOP) in conjunction with I/O Adapters (IOA). The i5 offloads this work to the IOP's freeing up the CPU(s) to run many more applications. This is not how other servers operate.

Automatic Load Balancing - iSeries creates one large disk pool that automatically balances content across all disk heads for maximum performance. This means you never need to know where your data is, therefore negating the need for a $100K+/yr Data Base Administrator. I remember one of my old roles at a previous company was to keep track of where and how objects were stored for my department. And it was a big job. It is hard for non-i5 folks to realize that is not necessary. It took me 6 months after my arrival at IBM to understand why that role is not needed with iSeries. BIG money saver for iSeries owners.

Again around Data - the IBM i5 allows you to create data spaces that can be dynamically added to MS Windows or Linux partitions. Because these spaces are dynamic and not fixed (like adding a 120GB Hard drive to a PC for extra space), disk space is maximized and not wasted. Then you get the benefits of having this data under multiple disk heads. All this equates to lower TOTAL cost of ownership.

For writing Java Applications, the i5/OS Java Virtual Machine is embedded in the Machine Interface (closer to the hardware). In addition to this the JVM utilizes better garbage collection (which cleans up memory or unused objects no longer running). Instead of shutting down all threads (like other operating systems) on a Server to run garbage collection, iSeries shuts down one thread at a time picking up a double digit performance boost.

And what about TIMI? The Technology Independent Machine Interface (some call it "Firmware"), which allows a company to change the hardware without affecting the software and change the software without concern for the hardware. This is unheard of on most operating systems.

Also - I remember Over 65,000 virus threats to other operating systems - none to IBM i5 operating system or data. i5 DB2 data is particularly difficult to penetrate. Check Symantec's Site - it is likely there are still zero threats to iSeries or i5 data...even today.

These are a very small number of the long list that makes iSeries so different... but each makes iSeries less expensive to own. So I suggest businesses look at the longer term cost of a server or operating system... instead of just the acquisition cost. Acquisition of other servers may be inexpensive...but they often bite you over the long haul. It CAN be much more (or much less) to OWN a system ...versus what looks like the low cost of ACQUIRING a system. "caveat emptor."

Paul Harkins, Harkins Audit Software, Inc

The Best Corporate Computer there ever was

The IBM System/38 and its follow-on computers, the AS/400, and the iSeries are the very best combination of brilliantly conceived and revolutionary computer hardware and software that I have experienced in my 43 years in corporate programming.

In fact, the introduction of the System/38 in 1980 prompted me to abruptly leave the IBM Data Processing Division (DPD) where I was a systems engineer supporting the System/370 mainframe computers, and switch to the competing General Systems Division (GSD) which developed and announced this fantastic computer.

I was about to accept a great three-year assignment with IBM World Trade Corporation in the IBM Process Industry Center in Düsseldorf Germany to develop an IBM apparel product for the unannounced IBM 4300 (code named E series) replacement computer for the System/370

when I was stunned by the elegance and power and the simplicity of the IBM System/38 announcement

The reason for my giving up skiing in Switzerland and living abroad at an IBM headquarters location for IBM was selfish. The development of the IBM ERP apparel system on the System/38 would clearly be many times more productive, and be much simpler and more satisfying, and produce a better product in less time then developing with the aging and difficult software available on the System/370 or its follow-on IBM 4300. I actually told my furious Germany born wife Gisela and our children that they would be skiing in Switzerland while I was trying to finish my ERP product by the required announcement date.

With the System/38, IBM Rochester had made what was difficult very easy and transparent to programmers. For instance, in the System/370 doing online screens required working with the IBM online product CICS. CICS required very small program modules called Transaction Processing Programs (TPPs) which were a maximum of four thousands bytes each and complex Assembler or COBOL processing of these online processing programs.

The System/38 totally simplified both batch and interactive programming by **integrating and simplifying** the online screen processing in a conversational programming approach within the OS/400 operating system. This allowed System/38 application programs to be programmed in a natural way in a powerful but easy Report Program Generator (RPG) programming language as the programmer implemented the application and in "pleasingly plump" robust application programs that were very easily maintainable.

IBM and particularly Dr. Frank Soltis and the Rochester programming team got it incredibly right with the System/38 by doing managing all the difficult system hardware and system software things and shielding the corporate programmer from that difficulty while allowing corporate programmers to focus on the creative part of programming corporate business applications. The result was perhaps a ten times increase in corporate programmer productivity with the System/38 and RPG over the System/370.

IBM has multiplied the power of the original System/38 hardware by many thousands of times with the new iSeries I5 processors, and is poised

to multiply the I5 processing power another *billion* times over the working career of a programmer.

Today, the iSeries also enjoys the unprecedented capability to completely audit the execution of every source statement and the variable data in real-time as programs execute. This allows programmers and auditors to see everything executing inside the computer and to audit or log everything for later review. This auditing capability uniquely satisfies the Sarbannes-Oxley legislation requirement of "auditing at every level", and provides a quantum jump in program quality and programmer productivity.

Bob Morici, Former IBM Field Technician (SE), iSeries Brand Representative

The Casino System

The casino industry was not automated in the late 1970s. Legal gaming was limited to Las Vegas. The Las Vegas casinos were largely family owned, with the exception of Howard Hugh's corporation (I can't remember their name, but they owned the Sands, The Dunes and 3 other famous properties). There were some systems running payroll and other back office functions, but the general consensus was that you could not automate the gaming functions. It was a service industry and good service required a high touch environment.

Casino gaming was legalized in Atlantic City in 1977. The State of NJ was determined to keep organized crime out of Atlantic City. As a result, there were many more regulations in Atlantic City than there ever were in Las Vegas, plus the market was quite different. While Las Vegas had vacationers and high rollers, Atlantic City had over 20 million people within a 2 hour drive. This resulted in lots of day trippers, some of them were regular Atlantic City visitors. For example, one large AC casino brought in over 150 busses per day.

The intense regulations along with the millions of fairly small, but regular day trippers required a level of automation far in excess of what existed in Las Vegas. The IBM sales team in 1978 located a Hotel system from a

hotel in Atlanta, Ga. It ran on a System/3 under CCP. This system was brought into Atlantic City to run the hotel side of the business. The local sales team brought a banking terminal, the 3610, into the casino industry and programmed it to be a point of sale device attached to the hotel application. This was the only terminal that the System/3 supported. The last part to be automated was the casino application.

I had been hired into IBM in April 1979. I had been a programmer at several large IBM customer sites. As a result, I was asked to write the first automated casino system on the System/3. I worked closely with Larry Cole, VP of IT at the Sands Hotel & Casino. Larry had worked with the accountants at the Sands to spec out a casino system. We completed the system and went live in August 1980. The Sands also sold the system to the Claridge Hotel/Casino and they went live with the application in April 1981.

The System/3 was outdated and we all knew that it had to be replaced, but it was all we had at that time, plus having a working hotel system for the System/3 was a big plus. And we were GSD, so we had to sell what was on the truck. The System/34 was available but did not have enough power for these applications.

In early 1982, we started our rewrite to the System/38. Four members of the team wrote the hotel system, which later was called HRGAS (Hotel Reservation Guest Accounting System), another member of our team wrote a point of sale system, based on 3483 cash registers attached to the Series/1. I began work on the casino system. I did receive some assistance from one of the hotel system programmers. No one really does anything by themselves, and we were a tight knit group.

I worked with Larry Cole of the Sands again. We developed the system, but the Sands was in the process of being sold, so we took the application live at Bally's Hotel/Casino, which was across the street from the Sands. The IBM account team was instrumental in working out all of these joint efforts.

We took the application live in early 1984. We actually completed the application in 1983, but at that time the Casino Control Commission was fighting with the property owners for 'unfettered access' to this new system. The industry does not like regulators wandering around their

systems. This issue went all the way to the NJ Supreme Court who finally ruled in favor of the Casino owners, and we were able to go live.

Our Branch Manager, Harry Griffiths, had wanted to create a Hospitality Competency Center in the branch office. Las Vegas was changing. Large companies were building casinos, the old time family owners were moving out. Howard Hughes died. Harry knew that our systems would fit in the new Las Vegas and this would allow us to poach in their territory. So we tried to purchase these applications from the Sands (casino) and Harrah's (hotel). We wrote the applications under contract with these customers, so they owned the rights. IBM management did not have the foresight that Harry had; they soundly rejected this idea. IBM did not want to get involved with the casino industry.

Larry Cole at the Sands did not want to be a software vendor, but he owned a valuable asset. He made an agreement with Russ Keil of the Claridge. Russ left the Claridge and formed Logical Solutions Inc. LSI added marketing modules and started the re-write of the point of sale system. Since the POS system was not System/38 based, but Series/1 based, it had to be re-written every few years as technology changed.

Today, the casino system is owned by one of the Casinos. Russ has retired, Larry Cole died in Oct 2002. It has a market share in excess of 70% world wide. The Hotel system has a similar market share, but the POS never really achieved the market success of either Hotel or Casino. The People Republic of China authorized 3 casinos on Macau, which had been returned to China from Portugal. All 3 of them run this casino system.

In 2001, my second daughter, Krista, went to work for Bally's in AC. She used the system that I developed the year she was born. I received quite a bit of free advice as to how I should have done certain interfaces. Krista has since left Bally's and returned to school.

If our AC customers had gone with the darling of the industry, they would have written this on Wang, then rewritten it on a DEC Vax, then Unix (several iterations), perhaps VSE, and someone would have given Windows a try (a couple of iterations there too). As it was, they have never re-written a line of code because of the changes from the System/38 (at release 4.1) through the AS/400, through the iSeries and the i5.

This next piece of the story involves two casinos that were part of industry consolidations and neither exists today so I will just call them Casino 1 and Casino 2.

I was still working in AC in 1991. At that time, one of the Casinos where I did some work on behalf of IBM owned 4 other large casinos. With the Casino Software I wrote, they managed it all with only 9 professional systems folks. It was a major operation. There were also secretaries and operators who are not included in this count.

Then, this Casino (a. k. a. Casino 1) bought another Casino (a. k. a. Casino 2) that was actually bigger than them. Casino 2 had been running on IBM mainframes because the IT management there did not want to use the System/3, when they opened in 1979, and instead chose the IBM 4341 mainframe.

I went over to Casino 2 at the time of the acquisition and was really impressed with the large number of people walking around, the massive size of the IT staff and the huge computer room with lots of blinking lights. Soon, I realized that they were not doing anything more than our AS/400-iSeries-i5 customers, and they weren't doing it as well or Casino 2 would have bought Casino 1 and not the other way around.

As a system that really affects the bottom line, the Casino 1, Casino 2 story as much as any demonstrates the value proposition of today's i5 and that goes way back to the IBM System/38. With the i5 as I have found in most instances, less is more (staff, downtime, errors) and you get much more for much less, and that costs a lot less than more. As you might expect, the rigors and exactness of the Casino industry could accept nothing less.

Biographies:

Jim Sloan is a retired IBMer (1991) who is now President of Jim Sloan, Inc. Jim was the lead software planner on the System/38 Operating System project in IBM's Rochester Labs until he retired. From the beginning of AS/400 time through the early stages of development

through completion and to the ultimate success of the System/38, Jim Sloan saw major action with the historic AS/400 product line. He continued in this capacity through the development and the early releases of AS/400 and through his company, Jim Sloan Inc., Jim has worked with the AS/400, the iSeries and now the i5 products. While he was still working on System/38, Jim started what is known as the QUSRTOOL library and he wrote all of the "TAA" Tools in the library. Since 1991, Jim Sloan, Inc. has a license from IBM to include the TAA Tools in his TAA Productivity Tools product. Jim is the developer of this product.

My interview with Jim Sloan was the first time I had the opportunity to be one on one with him, but I had spoken with him as part of small groups at COMMON conferences over the years. He is quite a guy. He is one of my favorite technical speakers of all time. He knows APIs and CL programming like the back of his hand, and he has a masterful presentation technique. As an aside, Jim has spoken at every COMMON Conference since 1979. He is truly an AS/400 and System/38 folk hero. He is a legend for those of us that have been with the product since its early days. It is a pleasure to include Jim Sloan's comments about our favorite system:

Skip Marchesani retired from IBM after 25 years and is now a consultant with Custom Systems Corp, an IBM Business partner. Skip spent much of his IBM career working with the Rochester Development Lab on projects for S/38 and AS/400, and was involved with the development of the AS/400. He was part of the team that taught early AS/400 education to customers and IBM lab sites world wide. I met Skip in Philadelphia in 1980. He was my instructor for several weeks of internal IBM System/38 education when we were preparing to initially install System/ 38 boxes in the local offices. Those were the days.

Skip is recognized as an industry expert on DB2 UDB for iSeries and AS/400 and author of the book DB2/400: The New AS/400 Database. He specializes in providing customized education for any area of the iSeries and AS/400, does database design and design reviews, and general iSeries and AS/400 consulting for interested clients. He has been a speaker for user groups, technical conferences, and iSeries and AS/400 audiences around the world. Skip is an award winning COMMON speaker and has received their Distinguished Service Award.

Al Barsa, Jr. is President of Barsa Consulting Group, LLC and Barsa Systems Distribution, Inc, which specialize in the iSeries - AS/400. Al is the President of the Long Island Systems User Group and covers new hardware and software announcements for iSeries News. Al is very active in the COMMON organization as a frequent speaker at both US COMMON and COMMON Europe, as an Editor of the COMMON technical library and as a member of the Speaker Excellence Committee, and has addressed other user groups throughout the world.

In the past, Al has been voted COMMON's Best Speaker", won Gold, Silver and Bronze medals, and has received COMMON's highest honor, the COMMON Distinguished Service Award. For the year ending 2002 and in six prior years, Al was named on the 'AS/400 Insider Weekly's' "10 Biggest AS/400 Market Influencers" list, making him the only person in the world ever to be named seven times! Both Barsa Consulting Group and Barsa Systems Distribution are IBM Premier Business Partners. Barsa Consulting Group was the recipient of the IBM Business Partner Mark of Quality Award.

Bob Warford is the Director of Information Systems / Computer Services at Labette Community College in Parsons, Kansas

Doug Hart is a midrange systems consultant for Whitenack Consulting. located in Rochester and operating in Upstate New York. He has been in the I.T. industry for over 30 years, with much of it focusing on the AS/400 family of computer systems. Doug works with systems used in very small "Mom and Pop" companies to the largest Fortune 500 enterprises

Ken Anderson, Quadrant Software -- A frequent speaker at QUEST, multiple midrange ERP specific conferences and local user groups, Ken Anderson has spent the past six years of his tenure at Quadrant Software promoting the concept of Electronic Document Distribution (EDD) solutions to iSeries users and IT managers throughout the North America. He has helped over 400 companies including Sara Lee Foods, Phillip Morris, and Office Depot; recognize the value of automating document processes. As a speaker Ken combines the

business strategies behind EDD with case study examples for an informative and thought-provoking presentation.

Dave Books. For the last three years prior to retiring (for good), Dave was an AS/400 consultant for Venture System Source, an IBM Business Partner. For the three years prior to that he was an AS/400 services consultant to IBM. Prior to that he spent 30 years with IBM, mostly as a Systems Engineer. Dave ended his IBM career with the title AS/400 Consulting Services Specialist.

Bob Cancilla has spent 30 years managing large-scale systems development projects and technology for both large insurance companies and independent software development companies, and he has been involved with AS/400 Internet technology since its inception. He is managing director and founder of the 6,500-member computer user group Ignite/400.

Paul H. Harkins, President and Chief Technology Officer of Harkins Audit Software, Inc., is still an active corporate programmer.
Mr. Harkins has been working with IBM systems for more than 40 years, including 21 years at IBM, where as a senior systems engineer, he was involved in hundreds of customer accounts worldwide and where he created the original IBM Apparel Business System, the first on-line IBM software package ever designed for the apparel industry. Paul has published articles relating to programmer productivity in several information technology magazines, and is the author of the newly published book "How to Become a Highly Paid Corporate Programmer". He also pioneered a software auditing technique to increase programmer productivity, the Real-Time Program Audit (RTPA), an award-winning software utility. In August 2004 Paul was awarded U.S. Patent 6,775,827 B1, for his invention of the Real-Time Program Audit software auditing idea.

Mr. Harkins holds BS and MBA degrees from Drexel University, and is a graduate of the IBM Systems Research Institute (SRI). His email address is paulhark@aol.com.

Bob Morici is a former IBM Systems Engineer (SE) who now works for the IBM iSeries-i5 Brand. He focuses on IBM's largest iSeries customers world wide. Bob's IBM career spans 26 years. As an IBM SE in Atlantic City for 14 years, he took on the major role in developing the Casino System and he assisted in opening most of the Atlantic City casinos. When IBM changed its business model to resellers, Bob left Atlantic City and became a certified AS/400 sales specialist in the Philadelphia area. For a brief period he left IBM and became a business partner and recently rejoined IBM in his current position.

Chapter 4

Why Have We Not Heard about the All-Everything Machine?

IBM In 1975

In 1975 when I worked for a local IBM Branch Office in Scranton PA, IBM was the only game in town to supply software and hardware solutions for small businesses. Its systems, for the day and age, were remarkably easy to use. A small IBM lab in Rochester Minnesota had started IBM's dominance by creating a machine called the System/3 in 1969. By the time 1975 came around a smaller version of the machine had been introduced. (See Chapter 5 for full details of the System/3 origination and progression).

The new system was named the System/32 and it was unique in its small size for the time. We joke today that it was desk-sized, not desk-top as today's many PCs. This machine was actually bigger than most server racks are today in small businesses. But, it was desk sized. Well, in reality it was big desk sized.

No, it is not the all-everything machine that this book is all about but at the time, IBM's General Systems Division, in which I worked, treated it as an all-everything machine for very small businesses. I must admit that for its day, it was quite a unit. And in historical context, it is one of the ancestors of today's all-everything machine.

Application Software Challenge

At the time the System/32 was announced, IBM's General Systems Division had intentions of being the leader in small business systems. It was the best place to work in all of IBM.

To be a leader, GSD had to supply software solutions to its prospects. Though these packages sold well at the time, the only surviving software package of the many developed at that time by GSD for the System/32 is something called MAPICS (Manufacturing Production and Information Control System), a popular ERP package of today. This package evolved from a System/32 version called MMAS which stood for Manufacturing Management Accounting System.

In a way, I have already given you a big clue that the System/32 application software effort did not succeed in a historical sense. Over 50 different packages, as I recall, were introduced by GSD in the mid 1970's and MAPICS is the sole survivor. That is not to say that the packages were not good or that the customers were unhappy. The packages were good and the customers were mostly happy and the System/32 was an absolute raging success for IBM. Like all real computer systems that I have ever worked with at IBM, the System/32 did not break… ever. OK, it did! But when it did, even the IBM repairman was surprised.

These application software packages were sold to first time computer users who had to be taught the discipline of working with a system in which pencil erasers, creating new ledger cards, and rewriting accounting journals to cover mistakes were strictly verboten. IBM had a staff of trained computer experts at the time called Systems Engineers (SEs) who patiently held the hands of its System/32 customers, sometimes round the clock, until the kinks and sometimes the attitudes were ironed out.

In the mid 1970's I can attest that SEs were feeling the strain of working sixty to eighty hours per week to assure the success of branch sales office customers. Though we wanted the office to meet and to exceed quota for sure and we were all motivated to get the job done and leave the client a happy, repeat IBM customer, the work was long and sometimes even too long. Moreover, it seemed sometimes that inefficiencies in the software that caused the SEs and System/32 customers much angst and much overtime hours could not be fixed by the labs expeditiously enough to minimize the time to install a new system. Knowing the state of the software, you can imagine the reaction of IBM's own customer support

team when the company launched its best small business advertising campaign ever.

IBM's Best Advertising Campaign Ever

It is in the light of the days of 60 to 80 hour work weeks and no relief in sight that I reminisce about the best advertising campaign that I can recall for IBM's small business offerings (System 32 and its Industry Application Program software). It was clear to me when I saw the TV ad for the first time that the IBM Company intended to sell a zillion System/32s.

I admit that I had two emotions about it at the time. I was first tickled that IBM was aggressively marketing because I knew there would be good results and my job would be well secure. At the same time, I was concerned that there were not enough of us locally to make the installations occur if the customers who received IBM's mass marketing message and had their expectations all jacked up decided to buy at the same time.

I can remember the ad almost as if it were yesterday. In fact, I would suspect that most Systems Engineers working with IBM's System/32 clients still remember it well. The three biggest problems that we had with the television ad were as follows:

1. We felt that customer expectations for a smooth installation would be over-inflated.

2. Since initial expectations would more than likely be unmet, we felt that customer expectations for support above and beyond the call would also not be able to be met.

3. The ad would create a customer who would not trust IBM or its representatives (us) again.

Show Me The Ad!

I will be the first to tell you in retrospect that I feel that the ad was sheer genius. IBM has yet to advertise the i5 s it did the lowly System/32 back in the mid 1970's. That's why you may not have heard of it. The System/32 ad may not have been reality, but neither are most ads. Since that ad, and over the thirty years that have passed, the hyperbole used by IBM's competitors makes this little ditty from the mid seventies seem mild in comparison.

Picture the beauty of this scenario as the camera breaks away from your favorite program to an ad with no announcement. You see the loading dock of a company like yours with a truck backing in. You see the desk-sized System/32 being wheeled in to the factory / warehouse and then very quickly into the office. You see a short period in which the packing material is removed and the brave installer from IBM plugs the unit into the wall – all very quickly. An already trained System/32 operator from the company takes control of the machine and the camera moves to the printer already rapidly producing the company's aged receivables report – just in time. It was a miracle.

Being trained to be a nitpicker, I had all kinds of questions of how that could ever be. Where did the data come from and how could the customer person already know everything to do. Selfishly, I saw my workweek increasing even more than it already had – for no additional compensation. None of my peers were thrilled with the ad either. We hoped nobody actually believed it.

I am in my fifties today and not my twenties. I have learned plenty since the 1970's. In retrospect, it was a wonderful ad and no rational being expected that this machine would be able to do all that it did in those few seconds in that ad. No company at the time would already have the applications and up-to-the-minute data pre-loaded on the computer at the plant from which to produce the business reports that just kept coming off this obviously phenomenal new machine.

For anybody in IBM who was not working the 60 or 80 hours per week, it was viewed immediately as a great ad. IBM had done the right thing with the ad. It worked. The systems sold as fast as IBM could ship them. People in strange places were talking about an IBM business system. The ad had done the trick.

So SEs had to become better friends with their customers to get the extra time needed to make the increased installations a success. We did! Looking back, it was all good. And, IBM customers really appreciated the effort.

As you will learn in the next chapter, the all-everything machine came from the same roots as the System/32. It was intended for the same general audience though perhaps the specific takers would have somewhat larger wallets.

In 1978, IBM introduced the all-everything machine as an infant to the data processing industry. There was a big announcement meeting and a big press conference in October, 1978 when the first all-everything machine was unveiled. However, from 1975 to today, even on the eServer i5 announcement day, I cannot recall an IBM sponsored TV ad that was so realistic that IBM's own technicians did not know how to limit the customer perspective.

Who Has Heard of the All-Everything Machine?

This Chapter title is a question, "Why Have You Not Heard About the All-Everything Machine?" The business answer to that question is that IBM has been successful with small businesses systems for over thirty years from well before the time of the infamous System/32 ad. The IBM Company has not had to advertise in order to achieve its sales or revenue objectives. There are those of us out here though who have a feeling that we may have complained a little too loudly back in the mid 1970's when IBM broke its tradition and chose to tell the world about its marvelous "little" System/32. Hey IBM, It's OK to tell the world about the all-everything machine – even if sales are good.

At any rate, though IBM's systems are still very successful, and they still sell well, it is my opinion that more and more business people should know about today's all-everything machine as an alternative to the most hyped systems of today coming from Intel and Microsoft and from Unix vendors. None of these other systems come close to stacking up, feature by feature to a system family that has endured and has advanced and has

helped many businesses prosper and has now evolved into a system that is truly the all-everything machine.

So, you may be asking, what is this all-everything machine? You are about to learn its origins in Chapter 5 and more about it in each of the following twelve chapters. It's a big story. I can't tell you that the all-everything machine will be running your A/R reports right off the truck. However, I can tell you that it is a quantum leap in sophistication, elegance, and capabilities over the ever popular IBM System/32, and any other system that anybody can mention – even with more than one breath. You and I would both be tickled if we had one running our businesses.

Chapter 5

History of Computers from IBM Rochester

The Rochester Mission

Once upon a time, in a small IBM laboratory in Rochester, Minnesota, there was a team with a big mission. Their job was to build a more modern set of unit record equipment. The Rochester team was blessed with the electrical and mechanical engineering know-how that could make the project a success, but they realized that because it was the 1960s, electromechanical machines would soon not be in demand. After all, the IBM System/360, the first solid sate chip-based computer already had been shipped; it was a huge success; and computers were really catching on in the marketplace.

> Note: Unit record equipment is a term used to describe the family of machines that would read or punch out IBM cards prior to the advent of bona fide computers. This gear was also called punched card processing equipment. Even after computers came to town these machines continued to provide accounting reports, sorted and/or merged card decks, duplicate card decks, and interpreted card decks, as well as calculating punched card decks for countless businesses. It was always impressive to see and hear all of this 80-column card gear in action. For a better understanding of unit-record gear and to see pictures of many of these behemoth machines, see the Chapter appendix on page 80.

The Rochester team was well aware that the mission to build real computers rested elsewhere in IBM, yet they earnestly believed that they

should use computer technology in addition to electromechanical circuitry, in the new machine set. Though there may always have been a desire in Rochester to produce the all-everything machine, they also knew that if they called this new machine a computer in its internal project stage, they would not gain IBM's approval to build.

However, Rochester was approved and it got the budget to build the next generation of "card processing machines." Officially, that's what they began to develop. Unofficially, however, the team knew they were designing and building a new computer system based on unit-record storage. The machine that flowed from this work would be called the System/3. It would change IBM forever, offering ease-of-use computing to small businesses for the very first time.

Once the IBM System/3 was introduced in the fall 1969, the Rochester team was no longer able to hide the fact that it had built a bona fide computer. The first System/3 Model 10 Card System would be recognized in the industry and in IBM as a computer system, albeit one with limited capabilities.

Lots of Time to Think

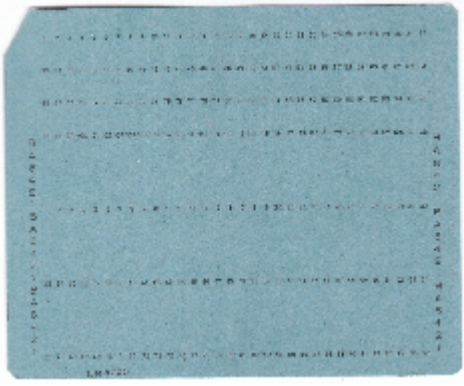
Some say Rochester Minnesota is a land where all there is to do is think. The opportunity to think in the cold while enjoying more than 250 days of sunshine each year made Rochester the perfect site for the conception of a new generation of computing. Though the System/3 was simple, it was very capable and innovative. A picture of the announced System/3 Model 10 card-only system is shown in Figure 5-1.

Figure 5-1 IBM System/3 Model 10 with MFCU (Right) and Printer (Left)

One-Third Size, 20% More Data

The first innovation at Rochester was the introduction of the 96-column card (see Figure 5-2). It was one-third the size of the 80 -column punched card forms, in which many people over the years had received their paychecks and income tax return checks. By using a smaller card, all of the card processing equipment would be smaller and therefore, less costly to build. The main input unit for this card on the System/3 was a device called the 5424 multifunction card unit (MFCU). It is located on the right side of the picture in Figure 5-1. This name is a derivative from IBM's System/360 Model 20, which had a similar, but much larger, multifunction card machine (MFCM) that processed 80-column cards.

Figure 5-2 No Holes, 96-Column, System/3 Punched Card



The 96-Column Card Processing Gear

In addition to the System/3 itself with its magical MFCU, Rochester actually did build a new set of unit-record equipment. Along with the MFCU, this gear could do all of the work for 96-column cards that IBM's 80-column workhorses had been doing for 80-column cards since the 1930's. See the Chapter appendix for a look at the unit-record workhorses from the past. The two other pieces of card gear built by Rochester at this time were the IBM 5496 data recorder and the IBM 5486 sorter.

By any other name, the 5496 data recorder would be an intelligent *keypunch* machine. It was the source of original entry. Its purpose was to permit an operator to create 96-column punched cards that represented either master records or transaction records for the business. Combinations of holes in the three-tiered card represented numbers and letters. Together, these were the data elements that provided input for the system.

Before being processed in the MFCU, the data often would be sorted using the IBM 5486 sorter. This was a two-tiered desktop device and was necessary in order to re-sequence cards for processing. IBM also provided a sort program for the System/3 to companies that believed that they could not afford a 5486. This permitted them to sort their cards using the two hoppers and four stackers of the MFCU attached to the System/3.

96-Column Card Processing Versatility

Unlike other unit record incarnations over the years, there was no separate collator unit available that could be used to merge two decks of sorted cards. There was no interpreter that could be used to print the meaning of the holes on the top of the cards. There was no reproducer that could be used to duplicate card decks. There was no big calculator that could be used for computations. And there was no 96-column accounting machine that could list the cards and provide printed invoices, orders, or management reports. (The Chapter Appendix shows these forerunner 80-column card machines.) The System/3 would provide all of these unit-record-like 96-column card functions using the MFCU and its system attached printer, the IBM 5203.

For example, the MFCU, instead of a collator, was used for merging card decks. Special card programs were provided that enabled two columns of cards to be merged into one. The 5496 data recorder was used as an interpreter to print on the cards punched by the MFCU. Another special card program permitted the System/3 MFCU to reproduce cards by reading one deck on the left side and punching out a duplicate deck on the other side of the MFCU. The central processing unit (CPU) of the System/3 provided calculations and report formatting. (The CPU frame is the highboy column in the middle of the picture in Figure 5-1.) Finally, the System/3 complex included a choice of printers. The 5203 Printer (shown on the left side of Figure 5-1) printed reports at several hundred lines per minute.

There was no disk on the original System/3 computer system. Cards were the only storage medium. The system came with just 8k of memory as standard. That's a mere 8,096 memory positions. The System/3 card system did have a mini no-name operating system. It was provided in a stack of cards less than an inch high. This deck of cards was called the System Initialization Program (SIP), and its job was to simply "boot" the system. After powering up the unit, an operator would place the SIP deck in MFCU1 (the first hopper of the MFCU) and press the Start button. When the SIP deck was read, the System/3 was ready for business.

Powerful Business Language for the New System/3

Another major innovation for IBM at the time was the perfection of the RPG (Report Program Generator) programming language in the form of RPGII. This language was originally built for very old IBM computers in the 1950s, such as the IBM 1401, but it had not yet been perfected and prior to its use on System/3 it had a questionable reputation. The RPG II for the System/3 was lots different. It had all the characteristics of a real programming language. It was rich in business function and thus made the System/3 a real business computer. The language was instrumental in making the System/3 an instant success in small businesses. It was simple. It was somewhat English-like, and, unlike COBOL, it was not verbose or intimidating for new programmers. Most of all, it was easy to learn.

Since there were not many for-hire programmers back then, the lucky folks tapped to learn RPG in the 1970s with System/3 were often young, bright, and trustworthy. They held other positions in their companies and seemed like the right candidates. Most of these programmers have grown up to become the gray-haired IBM i5 professionals who are now approaching retirement.

Disk Drives for System/3

In late 1969, IBM saw the need to make the System/3 a more capable computer by adding disk storage. As shown in Figure 5-3, in the area directly under the MFCU, Rochester provided space for four disk drives. These were known as 5444s, and they were stacked two in each of two drawers. In each drawer, one drive was fixed and the other drive permitted removable cartridges to be mounted / dismounted thereby providing additional removable storage. Each drive, fixed or removable, could hold 2.45 million characters of storage. That was it. But back then it was so much that for disk based System/3's, the second drawer was optional. The Photo in Figure 5-3 shows the optional second drawer open and a friendly IT person is inserting a removable disk cartridge.

Figure 5-3 System/3 5444 Disk -- Open Bottom Removable Drawer (R2)

New Disks Form Basis of New System/3

In 1970 IBM created a new model of the System/3 with a keyboard console and a dot matrix printer as part of the basic setup. The System/3 Model 6 also used the IBM 5444 Disk Drives. The keyboard was its only input device. No card reader would ever be attached to a Model 6 so disk was its only storage. Later as CCP (See page 76) became successful on the large System/3 models, IBM re-introduced the System/3 Model 6 with substantially more standard memory along with local terminal capability. The System/3 Model 4, announced in 1975 looked almost exactly the same as the Model 6. The one noticeable difference was that a model 4 had a 480 character CRT as its communications console. A picture of a System/3 Model 4 is shown in Figure 5-4.

Figure 5-4 IBM System/3 Model 4

More Storage, Please

As the demands for more storage on the System/3 model 10 increased, IBM attached its more capable mainframe heritage 2319 drives to the System/3, re-christening the units as the 5445 Disk System, whenever they were attached to a System/3. (See Figure 5-5) Each of these drives could hold 20.48 million characters of storage.

Figure 5-5 IBM 5445 Disk Drives

As the product matured, the 5445 drives were no longer adequate to satisfy the storage requirements of larger System/3 customers. In the mid 1970's IBM announced that four of the mainframe developed, innovative

3340 disk drives using the 70 MB IBM data module were able to attach to the System/3 model 15D, the largest System/3 ever built. Four 3340 drives are shown in Figure 5-6 along with an IBM 70MB data module sitting on top of the third drive.

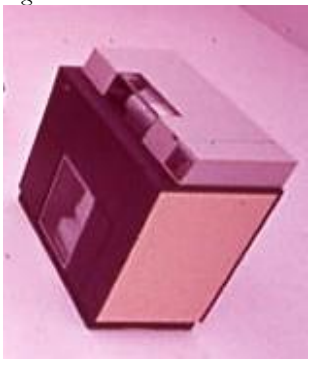
Figure 5-6 IBM 3340 Disk Subsystem with Data Module

For backup, all of the models of the System/3, except the Model 4 and Model 6, were able to attach the IBM 3410/3411 Tape subsystem as shown in Figure 5-7.

Figure 5-7 IBM 3410/3411 Tape Subsystem

Eventually, faster printers, such as the legendary IBM 1403 (1100 lines per minute), as shown in Figure 5-8 were added and the System/3 became a very popular small business computer. All models of the System/3 were very successful and profitable for IBM, and the machine was well-loved by its users.

Figure 5-8 IBM 1403 Printer



IBM rewarded the Rochester Lab for its accomplishments by permitting the Lab to continue making these computers. The biggest and most powerful System/3 was introduced in 1973. It was known as the Model 15D. Other System/3 models included Models 4, 6, 8, 10, and 12.

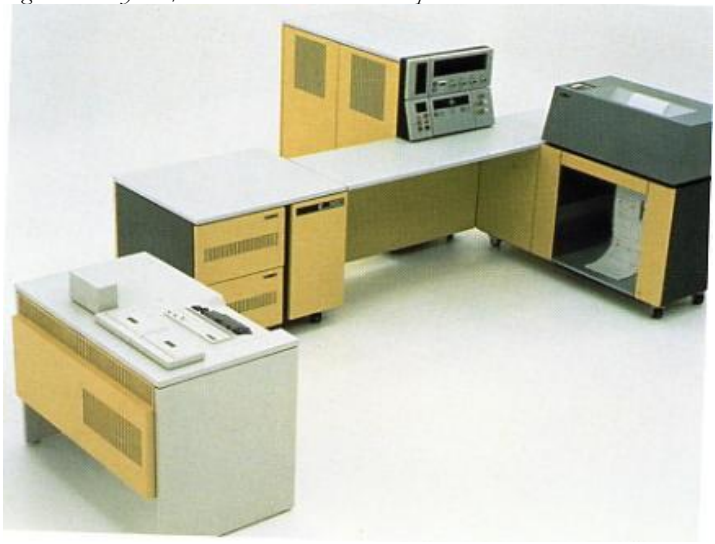
The System/3 model 15 is shown in Figure 5-9. The System/3 Model 12 is shown in Figure 5-10. The System/3 Models 4, 8, and 12, were introduced later than the Model 15 in the System/3 product life cycle.

Figure 5-9 IBM System/3 Model 15

Figure 5-10 IBM System/3 Model 12

During this period, IBM moved from card-oriented processing to floppy disks in eight-inch packages. The later System/3s were all "cardless." (See Figure 5-11 for a picture of a System/3 model 8 cardless computer with its direct attached IBM 3741 Data Station / diskette reader. Note also that the Model 12 in Figure 5-10 is also "cardless.") Therefore, the unit record façade for Rochester soon came to an end, yet the plant continued to make System/3 machines, which everybody referred to as "computers."

Figure 5-11 System/3 Model 8 "Cardless" Computer



Made for Humans, Not Machines

In addition to RPG, one of the factors that made the System/3 easy to use was its control language, known as the Operator Control Language (OCL). All computers preceding the System/3 required humans to learn cryptic languages, such as Autocoder, Symbolic Programming System (SPS), or Job Control Language (JCL), in order to communicate with the machine. Rochester intuitively knew the old way was not going to fly with a machine destined for small businesses and run by non-IT professionals. Programmers at the time who got their first look at OCL for the System/3, especially those who were mainframe-trained were amazed by its simplicity.

IBM made the System/3 control language easier for the programmer and user in the business environment, rather than for the software engineer in IBM who had to write the complicated routines that would scan the cards and interpret their meaning for the machine. Before the System/3 existed, the control language used on IBM's and others' machines was very cryptic and quite difficult for a normal human to read, and even

more difficult to write. A control language statement for a mainframe disk drive, for example, might look like the following:

```
//
Dlbl,,,3,,42,,sys011,,39,payroll,,,,99999,,,en
```

There was nothing easy about writing this type of mainframe statement. If you are an old mainframe person, you know that this is not exact but it is representative. Mainframe job control language (JCL) was quite difficult to master and it took a significant amount of time to get this stuff to work. For the non-veteran, it was almost impossible to know how many commas were needed in-between parameters. If you were off by one comma, the statement would mean something entirely different than what you intended, and the mainframe machine was very unforgiving and not very helpful. System/3 OCL was much different. It was English-like and keyword-driven. A sample statement might look as follows:

```
// File Name-Payroll,Unit-F1, etc.
```

The purpose of showing these statements, of course is not to teach about old computers, but to give a perspective as to how much simpler the new System/3 made computing at the time. Because the new OCL was keyword-oriented instead of positional, programmers no longer had to worry about how many blank commas to leave in between parameters. The "Unit=F1" part of the S/3 statement above was needed because the system back then had more than one disk drive. Just like a PC with multiple disk drives uses one-character symbols, the letters A through Z, to distinguish the drives, the System/3 used two-character symbols. Instead of A, B, C, or D drives; the System/3 drive names were F1, F2, R1, and R2. The F's were for the two fixed drives, and the R's were for the two removable drives. Today, other than diskette, CD, and DVD drives; disks are "fixed" in all computers and are non-removable, or fixed in place. The day of the removable hard disk passed when System/3 technology made its exit from the marketplace.

Terminals for System/3

During the mid-1970s, IBM developed a program on mainframes called the Customer Information Control System (CICS). This program ran in one part (or partition) of a mainframe and permitted many terminals to be used simultaneously with the machine. CICS was in a phrase, "difficult to use." The IBM 3270 terminal (Figure 5-12) was the terminal of choice at the time for CICS and other IBM terminal oriented operating systems.

Figure 5-12 IBM 3270 Terminal as Used on System/3

So that System/3s could also support terminals, after disk drives were introduced and accepted, Rochester built a program called the Communication Control Program (CCP) between 1971 and 1972. The System/3 model 10 was too small to support CCP so IBM built and introduced the System 3 Model 15. This box came with three partitions so that CCP would be able to have its own partition while the rest of the machine could do normal batch processing.

I can remember learning CCP and announcing it to the IBM office in Scranton, Pennsylvania. CCP was very similar in function to CICS. Along with the new capabilities, however, CCP added a degree of complexity to the System/3 environment for terminal processing, but it was nothing close to the degree of difficulty brought forth by CICS in the mainframe environment. Nonetheless, CCP was not for the casual System/3 programmer.

The IBM System/32 Is Introduced

With all of this innovation, the System/3 became a big hit in businesses all across the world, and Rochester became a big hit within IBM because it was making money for the corporation. In 1975, IBM Rochester was at it again. The Lab introduced a System/3-like machine that was desk-sized. Notice I did not say desktop. Desk-sized is about as small as it got back then. This unit had a keyboard and a small monitor, and had a printer attached to its back. It was an all-in-one computer called the System/32 (see Figure 5-13). In Chapter 4, as you may recall we discussed a TV commercial which IBM ran during the System/32 era.

Figure 5-13 System/32 – Circa 1975

The System/32 used the same notion of OCL, as did the System/3 disk systems—shown in Figure 5-3. However, since there was just one big disk drive on the left side of the unit, the OCL was even simpler than that of the System/3. There was no need for the R1, F1, R2, and F2 designations in OCL. The System/32, however, came with one major disadvantage. It had just one input keyboard attached to the top part of

its frame. Though key to diskette units, such as the IBM 3741, could be used to help with the keypunch load, and the System/32 did have a diskette reader that could read the standard fare 8" diskettes of the day, the one keyboard proved to be the major disadvantage of the box. As such, the System/32 lasted just two years before IBM improved the design.

In 1977, IBM announced the new and improved System/32. It was a boxy computer called the System/34 (see Figure 5-14). It used Operator Control Language, just as the System/3 and the System/32 before it. Therefore, the System/34 was also easy to work with, and OCL was a big reason. With the System/34, IBM shipped up to two disk drives. Unlike the System/3 whose OCL had to tell the system, which drive a file was on, as a predecessor to *Single Level Store (*Chapter 8*)*, IBM had improved its ability to treat all disk drives on a small business system as if they were part of one mass storage unit.

> Note: This was in 1977. Windows and Intel and Unix and Linux have still not achieved this major ease of use characteristic.

By using terminals instead of a built-in keyboard, the System/34 solved the "one keyboard" problem of the System/32. Up to sixteen separate terminals could be attached to just one Systrem/34 providing sixteen more online input devices to the system than the System/32. Thus, the big difference between the two systems was that the new System/34 was a multi-station, multi-user system. By introducing the notion of multi-user and multi-programming with the System/34, IBM enabled each user to have a piece of this one computer system as if it were his or her own machine.

Figure 5-14 IBM System/34 Multi-Station Computer

Though System/34 used terminals, it did not need the complexities of System/3 CCP or anything like IBM's CICS or BEA's Tuxedo. (See Chapter 9, Integrated Transaction Processing.) Terminal management was built-into the S/34's System Support Program (SSP) operating system. It was an industry first. The compilers were written to recognize a terminal as a real device thus making programming the S/34 for interactive work far easier than any computer vendor has yet to achieve.

Moreover, you could attach these semi-intelligent, high-speed terminals to the system over a local high speed wiring type called twinaxial cable, without the need for modems. IBM provided a link to RPG and COBOL so that programmers could directly control one or all terminals from one program rather than requiring a program for each terminal.

Note: Tuxedo is BEA's terminal monitor program, introduced in the 1980's with similar function and purpose to CICS.

The new terminal that IBM invented was also a major innovation for its day. It was big and square, and it was called the IBM 5250. See Figure 5-15. Each of these terminals, at the time, could be purchased for about $4,000. Though 5250s are no longer sold, the green-screen 5250 legacy continues today through PC products that emulate the 5250 terminal's data stream. The System/38 machine and the AS/400 historical line including the IBM i5, all of which are introduced in this chapter also use the 5250 display station protocol as their native terminal discipline.

Figure 5-15 IBM 5250 Type Terminal

The 5250 terminal had actually been built for the Rochester designed and developed System/38 computer system, which was to be the follow-on computer to the System/3 Model 15D and the entire System/3 line. The System/3 had used the IBM 3270-type terminal (Figure 5-12) that had been introduced for mainframes. The 3270 line continues to be popular on mainframes today and is an often-emulated terminal device.

In 1977, when the in-process System/38 was taking much longer to complete than originally anticipated, Rochester decided to announce the System/34 product line as an upgrade to the System/32 and as a stop-gap while the System/38 was being perfected. The 5250 terminals and printers that were designed for the System/38 were thus first used on the IBM System/34.

The First Version All-Everything Machine

In this book, you will learn lots about the System/38, the direct predecessor of the AS/400, and its origins and unique attributes. The System/38 was a well-designed system, using the best that IBM knew about computers. Rochester had never really built a sophisticated computer before. Therefore it was impossible for the engineers to know how difficult it would be to achieve the major technical advances brought forth with the System/38.

When IBM announced the System/38, in October 1978, Rochester knew that the machine was not working well enough for prime time. However, based on experience with other systems, the Lab felt that the machine would be ready in 1979, in time for the first customer shipment. System/3 Model 15D customers, as well as many others, enamored by the outstanding specifications of the System/38, signed up in droves on the day it was announced for an early shipment of this new box.

For System/38, there would be no early shipments. IBM seemed to take forever to give customers a ship date, and when they got one, it was two years out. There were problems with the box. There were so many new computer science attributes built into the System/38 that for a time it seemed almost improbable that the system would ever be completed, no matter how hard IBM tried. Yet, IBM did not compromise on the

underlying advanced architecture of the System/38. The company just dug in and made it work. It is no wonder why today that there is not any system in existence that has yet to catch up technology-wise to the System/38 machine that IBM announced way back in 1978.

Of course not being able to get a system out the door as promised created a big public relations problem for the IBM Company. It is ironic that Microsoft, a company competing for IBM's all-everything machine customers today has never seemed to have a problem announcing new worlds and delivering often less than a city block in need of immediate repair. At IBM, however, the inability to bring out a system on time was looked upon as shameful.

In 1979, to call off the dogs, Frank Cary, CEO and chairman of IBM at the time, appeared before IBM's customers and the world, and asked for forgiveness for delaying the System/38 for 11 additional months so that it would be ready for business use when it was first shipped.

The System/38 finally arrived in 1980 to a mostly welcoming customer set (see Figure 5-16). It was the best system that IBM had ever built. It uses the all-everything machine principles that are described fully in Chapter 8 Its underpinnings were so advanced that no machine, besides its direct descendents, the AS/400, iSeries, and i5, has ever reached the same level of hardware and software technology and integration.

Figure 5-16 IBM System/38, Announced in 1978

System/34 Was Available

Because of the delays, as well as the remarkable popularity of the System/34, total sales for the System/38 never surpassed 50,000 units. There are unofficial estimates that the total of System/38 shipments was even as low as 20,000 units. Yet the System/34, with its 5250 workstations, caught on like gangbusters and shipped well over 100,000 units.

In the early 1980s, the mainframe division of IBM became concerned that there were too many IBM systems aimed at the same customer. Mainframe executives were never particularly happy that Rochester built computers, and felt that job should be done in a mainframe plant, such as Endicott or Poughkeepsie.

To assess the feasibility of a product line consolidation and to get a jump start on the effort, IBM commissioned a big project and spent hundreds of millions of dollars trying to come up with a new system that, among other things, would do everything that the System/34 and the System/38 could. Before this new product design, which the mainframe chiefs in IBM anticipated would become its all-everything machine, had born any fruit, in 1983 the IBM Rochester Lab replaced its aging System/34 line with a snappy little box called the System/36. (See Figure 5-17.) By 1985,

the systems consolidation project, called Fort Knox, had failed. (See Chapter 14, "The Fort Knox Project."). Thus Fort Knox was cancelled and the future all-everything machine therefore continued to be based on the architecture of the System/38.

Looking back, even the mainframe component of IBM with access to all of the secrets of the System/38 advanced machine, could not launch an affordable product that would include all of its design points. There is no question that IBM was "mighty" during this time period. If the mighty IBM itself, with all its resources, through its most capable mainframe division could not re-build the System/38 as part of Fort Knox, it is no wonder that nobody else has yet to be able to do so.

Even after twenty plus years, other formidable 1980 era computer companies from DEC to Microsoft to Intel to HP to Sun have not been able to introduce a system as architecturally powerful as the System/38. Considering that the underpinnings of the System/38 are over thirty years old, IBM's competitors clearly had lots of time to catch up. The fact is: they can't. Even IBM couldn't do it again.

The fact is that if IBM had known when it launched the System/38 project in the early 1970's exactly how much effort and internal cost the System/38 was going to require, most analysts would bet that the machine, no matter how good, never would have seen the light of day. A naïve new IBM computer lab in Rochester Minnesota literally did not know it could not build a system as powerful as the System/38, and so they went ahead and ultimately did it. Without this naiveté and mother IBM's big pockets when Rochester failed in its prescribed time frame, the company would not be in the position that it is today of reaping the benefits of all this effort with its very own all-everything machine.

Figure 5-17 IBM System/36, Announced in 1983

Finally, the AS/400

After Fort Knox had failed, a project called Silverlake was initiated at Rochester in the mid 1980s to create one replacement box for both the System/38 and the System/36. After little more than two years, in June 1988, IBM announced the results of Silverlake as the Application System/400, or AS/400 (see Figure 5-18).

Figure 5-18 AS/400 Model B60 Circa 1988

In many ways the box was a repackaging of the System/38, but it also ran System/36 programs untouched. It also ran untouched System/38 applications in its own separate environment and it also ran mainframe programs using a facility called the Cross System Product (CSP).

Through a number of incarnations described fully in Chapter 11, in May, 2004, the AS/400 was reincarnated again as the eServer i5, the all everything machine, as shown in Figure 5-19.

Figure 5-19 The IBM eServer i5, the New All-Everything Machine.

Though I quickly brought you almost to the present with a picture of the new i5, it was way back in 1988 that the AS/400 became its grandfather and the basis of the all-everything machine. If you start adding them up, the machine in 1988 was equipped with the following four machine capabilities:

1. Native AS/400 Processing
2. System/36 Environment
3. System/38 Environment
4. Mainframe Environment with CSP

Note: The AS/400 is the immediate successor and a derivative of the revolutionary System/38 that was introduced by IBM in 1978. In October 2000, IBM renamed the AS/400 as the iSeries. In 2004, IBM renamed the iSeries as the eServer i5, a.k.a. IBM i5. In this book, for the most part, I use the term "i5" to mean the AS/400, iSeries and i5 other than for historical correctness. The System/38 is substantially less in power and substantially older than the AS/400, iSeries, and i5.

The 1988 AS/400 was a resounding success by all measurements but one. System/36 customers were not too happy about it. It was much different from the System/3, System.32, and System/34 heritage machines. It also appeared to be more complex because of its many new features. Moreover, the emulated System/36 environment did not initially perform as well as System/36 customers expected.

The System/36 crowd expressed their displeasure by keeping their old System/36 boxes as long as they could, and when they upgraded, they would buy either a second used System/36 (same size) or a bigger used System/36. It took a long time for IBM's System/36 customers to warm up to the AS/400. However, there was enough new AS/400 business at the time for IBM from the former minicomputer vendors, such as DEC and Data General. So, at the time, it was OK with IBM that the System/36 installed base stayed where they were, in their existing, "happy-with-their-old-system-state," for years after.

AS/400 Evolution

In 1995, IBM changed its AS/400 hardware to 64-bit RISC from 48-bit CISC, yet the company chose not to rename the system, as it had done when the AS/400 replaced the System/38. (See Chapter 11, "The Rise of the All-Everything RISC Machine" for a high level explanation of CISC and RISC architectures.) At the time, IBM made some additional changes to the box, and the new chips (PowerPC) permitted the former System/36 operating system called System Support Program (SSP) to run natively on a pre-release version of the new RISC AS/400, dubbed the AS/400 Advanced/36.

This machine performed exceptionally well, and IBM's System/36 customers rewarded IBM for giving them what they wanted by purchasing lots of these new boxes. Eventually, IBM was able to place the entire System/36 instruction set, as well as the AS/400 instruction set, on newer and better 64-bit PowerPC chips. After just a few years, IBM did so well that it was able to withdraw the Advanced System/36 from marketing. Today, the AS/400, the iSeries, and the i5 can all perform System/36, System/38 and AS/400 operations from instructions built within the same POWER architecture chip.

In 1997, IBM spent a little marketing money on the AS/400 image by adding an "e," for *e-business*, to the AS/400 name, thereby making it the AS/400e. IBM also renamed the AS/400 again in 2000, as the eServer iSeries 400. This name change affected only new shipments. AS/400s that were already installed in customer locations were not renamed. From this point forward, both the AS/400 name and the shortened name iSeries applied to the AS/400 server. Then, in May, 2004, IBM extended the line again with the introduction of the POWER5 processors and IBM renamed the machine as the IBM eServer i5. Now, versions of the box in use are known as the AS/400, the iSeries, and the i5.

Since 1995, with the introduction of the 64-bit RISC processors, IBM has boosted the power and the number of processors that are available on the AS/400-iSeries-i5 product line. In 2004, for example, with the POWER5 series of microprocessors, the company doubled the number of processors that could be packaged in one IBM i5 machine from 32 to 64 and increased the performance of each processor by well over 200%. In addition to changing the system name to the eServer i5, IBM also changed the name of the operating system from OS/400 to i5/OS.

The POWER5 brought with it the capability of having sixty-four phenomenally high-speed computers operating simultaneously in one i5 machine. That sounds a lot like a mainframe because it is. The eServer i5 is now recognized as a mainframe-class machine. Industry watchers are expecting similarity to be extended with the introduction of the POWER6 chips as the mainframe begins to use the same processor as the i5.

With all of the enhancements over its 17 years, the AS/400 is clearly the most architecturally elegant and capable machine in the industry. From the ground-up, it was built as an integrated machine. When you add this internal elegance to the powerful engines (64-way POWER5 and more POWER systems coming) that are now available with the IBM i5 technology, the box is clearly the best and most powerful computer of all time. With all this going for it, the IBM i5 is the machine that is recognized as giving the most value to businesses for the least cost.

Is It Really That Nice? Yes!

If the i5 were as easy to explain as it is to use, the public would already be aware of its nuances and ramifications. Knowing about the systems that

came before the i5 from the Rochester Lab, and recognizing that the IBM i5 is the follow-on to all those technologies, it is easy to surmise that with an IBM i5 at the heart of your computing infrastructure, life could not be much easier and more productive. It is a fact that an i5 adds more business value that goes right to the bottom line than any other server.

The IBM eServer Family

The AS/400-iSeries-i5 family is just seventeen years old, with roots about thirty years old. Like most teenagers, the machine depends on the good will of its parent, IBM, for its future well-being. Like any good parent, IBM has decided that its three other eServer children, the PC server, the Unix box, and the venerable mainframe, should be treated equally to the AS/400 historical line.

So, in 2000, IBM gave all of its servers the same surname, "eServer," and changed their unique names to line up better with their new surnames. The AS/400, for example, became the eServer iSeries. The mainframe became the eServer zSeries. The Unix box became the eServer pSeries, and the PC Server Line became the eServer xSeries. In 2004, as noted previously, the iSeries changed its name again to the eServer i5. Additionally, at about the same time, the Unix box changed its name from the eServer pSeries to the eServer p5.

Of all these eServer servers that IBM markets, the eServer i5 is by far the best!

IBM as a good parent cannot say that about just one of its four server children. However, with no blood relationship at all to the IBM servers of today, I can say it. In fact, that is the message of this book. It's why I wrote it. The IBM i5 is the best computer on the market. It is the best computer ever built. It is the all-everything machine.

Enhancements & AS/400 Marketability

As you can see in this little history of the IBM i5 product line, the company has enhanced the machine to make it a technology leader in many areas. However, until May 4, 2004, IBM had priced iSeries

hardware substantially higher than the same hardware in other systems. As an integrated system that ships with a complete operating system, integrated database, integrated transaction processing, etc., customers always saw great value in the machine so sales were not affected by what some thought was a higher price. Most of IBM's iSeries customers believe that the most advanced operating system in the world ships with the system, and so the extra value is worth the extra charge.

When IBM announced the new POWER5 based eServer i5, the company signaled that a big part of the hardware cost for acquiring a new iSeries family machine was being eliminated. With May 4th's tremendous jump in power and capability, coupled with a substantially lower price, the AS/400-iSeries, now in the form of the i5 model set is now an even more affordable machine for many small businesses. Considering that the box still ships with an integrated operating system (i5/OS), the hardware price is now well in line with the hardware on IBM's other servers. There sure is no reason for complaining, especially if you examine the cost of Windows server software and Microsoft SQL Server software. That makes today's i5 an even greater value than the popular AS/400 heritage systems of the past.

Chapter Appendix: IBM's Pre System/3 Unit Record Gear

This chapter appendix is structured to help those who are interested have a fuller understanding of the types of machines that were used in small businesses when the System/3 was being developed. In 1969 when I joined the IBM Company, IBM trained me to wire the panel boards that controlled the myriad of machines that are shown in this chapter appendix. Back then, IBM customers could get themselves a keypunch, a sorter, and an accounting machine for just over $500.00 per month. Before the System/3 was introduced, to get a jump on automated data processing, many chose to do exactly that.

Here are a number of the machines that were necessary in the pre-computer 80-column card processing world. These highlighted machines were collectively and methodically replaced by the System/3, and its IBM

5496 Data Recorder, and the IBM 5486 Sorter from 1969 through the seventies.

A Keypunch (Figure 5-20) is a machine that data entry personnel would use to type in data. For each record of data typed, the machine would "punch out" a card to represent the information record that had been keyed. This operation in computer shops in the 1960's was known as keypunching.

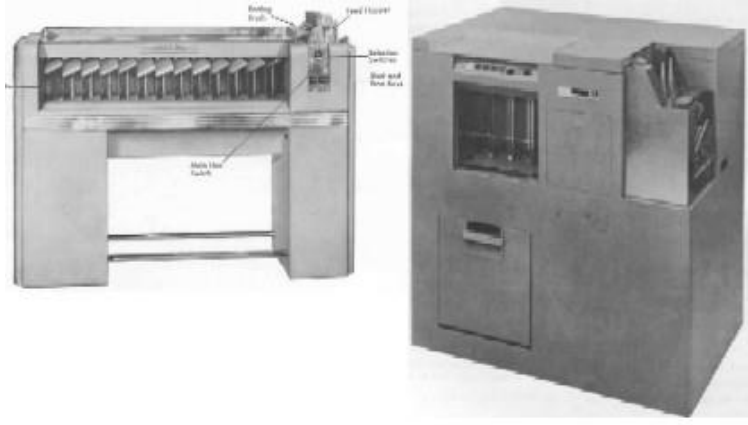
A Sorter (Figure 5-21 left) is a reasonably small electromechanical machine. It has one input hopper and eleven stackers. The Sorter has ten stackers for the digits 0 through 9 and an eleventh stacker for cards that could not be read (rejects.) A deck of cards would be placed in the sorter and the machine would read the value in the selected column and place the cards in the proper stacker after one pass. The operator would carefully place the results from all the stackers one on top of the other and the card deck would be in sequence (sorted) on that column. The deck would be placed into the hopper for as many additional passes as there were columns remaining to be sorted.

A Collator (Figure 5-21 right) is a large electromechanical machine that has two card hoppers and four stackers. Normally it would be used to merge cards together but it also could be used (with a different panel board) to match cards to assure for example that there were a transaction card for every master card. In this case the other two stackers would be used to select the unmatched masters and the unmatched transactions respectively.

A Reproducer (Figure 5-22 left) is a large electromechanical machine that would read in a deck of punched cards and punch out an identical deck.

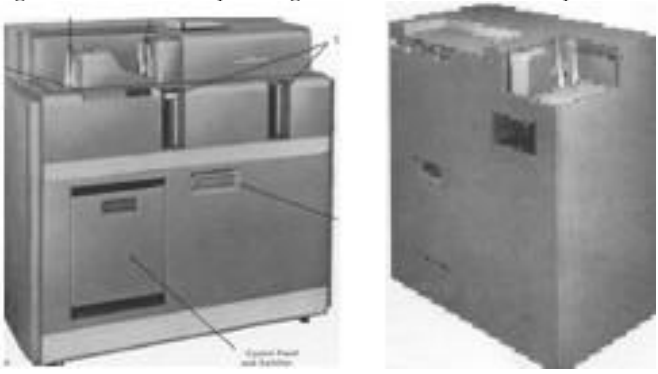
Figure 5-20 IBM 129 Keypunch Circa 1970

Figure 5-21 IBM 082 Sorter circa 1965and IBM 085 Collator



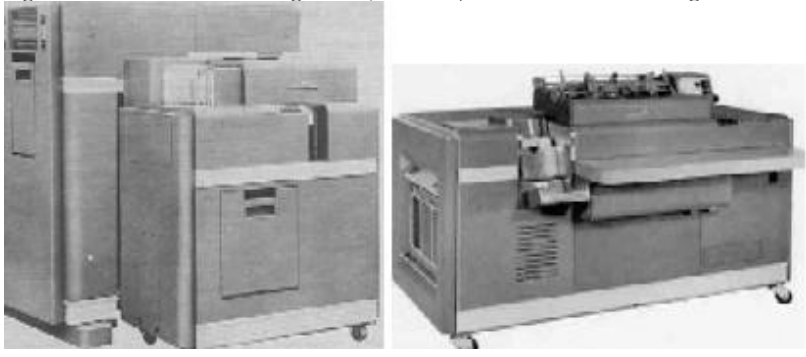
An Interpreter (Figure 5-22 right) is a large electromechanical machine that would read in the cards that had been reproduced without printing and it would read the holes and print their meaning in the form of letters and numbers on the top line of the card.

Figure 5-22 IBM 519 Reproducing Punch and the IBM 548 Interpreter



A Calculator (Figure 5-23 left) is another huge machine that is capable of performing computer-like mathematical functions using electromechanical circuitry. A panel would be wired, for example to read in values from distinct columns of an input card and the calculator would produce a result, such as a price extension and it would punch the result into another set of columns on the 80-column card.

Figure 5-23 IBM 604 Calculating Punch (Calculator) and IBM 407 Accounting Machine



An Accounting Machine (Figure 5-23 right circa 1969) is the unit that read in the cards after all the sorting and manipulating and it produced accounting reports. So, in many ways, it was a big card reader with a big printer. "Programmers" would wire a panel board to tell the machine what card columns from the input should be printed on what columns of a report.

Chapter 6

IBM's eServers & the IBM i5

The Best Computer Ever

The historical IBM i5 is the best and the most special computer ever built. That is why it is inconceivable that the company that owns the rights to the machine does not seem to try hard to earn even bigger revenues from it. For you music lovers out there, it may help to know that the i5 is to computers as what Bose is to great sound. Bring on the music.

As the direct descendent of the System/38, the i5 is even more functional and more powerful. The older System/38 line was not as well endowed performance-wise. In fact, because it was intended for smaller businesses, in its infancy it suffered from capacity constraints naturally imposed by IBM's mainframe division. Just as you would run your company, IBM management found no value in the idea that Rochester machines would compete with traditional mainframes.

So, IBM gave the engineers in Rochester specific constraints to assure that this all-everything machine that was being built would be well positioned to support small businesses, but not the big businesses to which IBM targeted its mainframe line of computers. The resulting system, known as the System/38 as introduced in Chapter 5, was thus underpowered for all of its inherent advanced capabilities. However, it was a heck of a machine for small businesses, most of whom had no idea the box was underpowered for its architecture.

As underpowered as it may have been, the System/38 was built with the same advanced architecture, and thus, by design, it is the same high tech machine as the AS/400 and now the i5. Therefore, one could argue that

the historical IBM i5 and the System/38 are singularly the finest computers that any company has ever made.

AS/400 Becomes eServer iSeries

In the fall of 2000, IBM changed the name of the AS/400 to the eServer iSeries 400. Many who earn their livelihoods from AS/400-related work have chosen to recognize the iSeries not as a different computer but as a branding change. For those working with AS/400-iSeries machines every day, IBM's renaming of the iSeries to the i5 was met with the same reaction. IBM's customers see the i5 machine as a logical extension of the finest computer system ever built, the System/38. When the Application System/400 (AS/400) was introduced in 1988, it was so different looking and had such better hardware specifications that System/38 and System/36 aficionados accepted the AS/400 name with no complaints.

Regardless of what you call the all-everything machine, anyone who takes the time to look deeply into this server would see a machine that is the embodiment of all that IBM knows about computers, implemented with elegance unparalleled in the computing era.

Only IBM Could Create an eServer i5

Only a big company with such huge resources as IBM could have conceived, designed, and built such a superior machine. For this, I regularly thank the IBM Corporation. IBM spent billions of dollars to develop and billions to improve the advanced integration features of the i5 system. None of the company's current competitors are in a position to even consider making such a technology investment.

Ironically, the approach used internally for the Windows, Unix, and Linux operating systems and Intel hardware is similar to the systems that predated the System/38. In other words it is legacyware brought forth to the future. Yet, for all the truth is worth, the industry press continues to hail the Windows crowd for their technical accomplishments and calls their wares ***modern***. It is IBM; however who has the most modern architecture ever developed for any computer system in the frame of an

IBM i5. Yet, for all the truth is worth, the press has no problem calling the new IBM i5 a ***legacy system***.

IBM Has the Server Bases Covered

What a blessing the IBM Corporation has today! It has all the computing bases covered. When you consider that Microsoft has just a piece (though a reasonably large piece) of just one base, PC software, you can readily conclude that IBM has the armaments that should power it to victory in today's computer marketplace.

First Base – PC Servers

In the personal/micro/X86 space, IBM has *first base* well covered with its industry-heralded ThinkPad, its appealing and inexpensive ThinkCentre, and its NetVista line, (all offered by IBM partner Linoma). The company also has its high-function, high-speed Netfinity Servers (now called eServer xSeries.) The mainframe-enriched xSeries servers compete head on with all PC Network servers running Windows NT, Linux, Netware, and OS/2 LAN Server. Most of IBM's success in this space is shared with Microsoft and Intel, who provide the bulk of the software and processor hardware in this system area. However, today, there is no question that IBM has very formidable offerings in this area.

Second Base – The Unix Box

In the multi-user and workstation Unix spot, IBM is well positioned on *second base* with a rugged "taken no prisoners" submission. It has developed a mature offering with its RS/6000 hardware (now called eServer pSeries – and p5) and its highly stable Advanced Interactive Executive. Dubbed AIX by IBM, this is the company's Unix operating system offering. If you want to buy Unix from IBM, you would buy its AIX offering.

On December 10, 2004, IBM added a new facility to the p5 hardware line. The company announced that a CD was available for purchase for p5 models so that the i5/OS operating system could run in up to two processors of a large eServer p5. In other words, not only can an i5 look

exactly like a p5 and run all of its programs on all its processors, a p5 can do the same for two processors worth of i5 applications.

Efforts to move the Linux operating system to the pSeries have only strengthened the product line in the Unix marketing space. The eServer pSeries (p5) offers i5-level hardware facilities to system customers who prefer the personality and the unique applications of a Unix machine. And with i5/OS as a guest, the p5 can even accommodate some light i5 style computing. There's no question you can get to *second base* with today's very powerful eServer p5.

Third Base -- Mainframe

In the traditional mainframe system arena, IBM's leadership in commercial hardware technology is unquestioned. Mainframes are the types of computers that General Motors and JPMorgan Chase Bank, and Prudential Insurance and other large companies use as their main processors to run their businesses. IBM's System/390 product set (now called the zSeries) competes against relatively few. The players in the large mainframe and supercomputer marketplace include Fujitsu, Hitachi, Cray, and not many others. In this period of resurgence for the power of mainframe computing, IBM is doing very well for itself. For sure, you can get to *third base* with a zSeries. And, there's not much wrong with a triple!

Home Run – IBM i5

In the business solutions sweet spot, IBM has hit a *home run* with its eServer i5 product line as it stands on home base as the obvious winner. Its biggest problem is that since the work that an i5 does so nicely can also be performed on the other three bases, though with far greater difficulty, IBM has a real marketing concern in understanding where the i5 box actually belongs in its product mix. The company also has a concern in making its purpose for the i5 clear to its IBM computer prospect list. Unless you already knew, or you are reading this book, it would be hard to tell the circumstances in which the i5 would be the compelling server choice over IBM's other fine servers.

Regardless of where it is positioned however, IBM has invested tons of money into the IBM i5 box and has in fact created an all-everything computer that analysts predict will one day soon be able to run

applications from all of the popular operating systems, including Windows and the mainframe operating system flavors including z/OS.

Though Windows and z/OS are not yet on the list, both Linux and AIX (IBM's Unix) already run on the i5. Moreover, these operating systems actually run better on i5 hardware because of its use of small processors as disk controllers to augment disk processing. In case you did not catch that, let me repeat, not only does the all-everything machine run Unix and Linux, it runs them better than if they ran on IBM's Unix box, the eServer p5. Additionally, these operating systems can run in partitions on the i5 so that even a small i5 server can run all of these operating systems at the same time.

Other than some confusion in product positioning as IBM works out the details of releasing the full bodied all-everything machine, the company is well positioned with the i5 on home plate for the big score of the millennium.

Even More Environments

In Chapter 5, we discussed the base capabilities of the i5 and how these already positioned it to be the all-everything machine. You may also recall that the AS/400 and now the i5 were introduced with environments permitting each to run the following:

1. Native AS/400 and i5 Processing
2. System/36 Environment
3. System/38 Environment
4. Mainframe Environment with CSP

With the availability of running multiple operating systems as noted above and with about eight years of work perfecting a native Java Virtual Machine (JVM), the i5 all-everything machine can now do even more as shown in the following add-on list to the four items above:

5. Java Processing through an integrated Java Virtual Machine
6. Unix Processing through AIX
7. Linux Processing through standard distributions

Considering that there are only two operating systems / environments that the i5 does not support today on POWER processors, (1) Windows and (2) IBM mainframe OS flavors such as z/OS, from a hardware and operating system standpoint, the future all-everything machine is certainly well on its way to realizing its full future. In case you were wondering, no other machine in the industry, from IBM, from HP, from Sun, or from Dell can do the same thing.

If you can believe the industry prognosticators both of these missing capabilities will be added to the i5 hardware when IBM changes the microprocessor base from the POWER5 platform to the POWER6. At this time, of course we may also see a name change from i5 to i6 since the rationale behind the 5 is the POWER5 microprocessor which powers the unit.

It makes sense that IBM will stop making expensive CISC (complex instruction set computing) processors that are unique to the mainframe and begin to migrate mainframe OS ware to run on the POWER processor line. The fastest AS/400-iSeries today in the form of an eServer i5 model 595 with 64 integrated POWER5 processors rivals the mainframe for best commercial performance. With even greater CPU power available in the IBM POWER6 processor that is expected this year or next, it would be imprudent for IBM to continue investing billions in unnecessary technology. Those billions would clearly be better spent making the mainframe OS run seamlessly on the next generation POWER processor. That's what I see happening.

Then, there is Windows. In many ways, knowing the haphazard methods that Microsoft has historically deployed in OS construction over the years, as characterized in the book, Barbarians Led by Bill Gates, IBM is understandably skeptical about running an error-prone operating system on such a solid machine. The book, a joint effort by Microsoft insiders, Jennifer Edstrom (daughter of Gates long time PR chief, Pam Edstrom) and Marlin Geller, a 13-year veteran developer who worked on DOS, Windows, and the Pen operating system, is so revealing about Microsoft's lack of discipline in its OS development efforts that there is no longer a mystery for me as to why I must reboot my PC so frequently.

Though IBM is not looking for ways to have to bring down its steady as a rock all-everything machine, one can bet that the company is concerned and that surely is one of the impediments to pulling it off. Having said that, it is a fact that Windows NT is the grandfather of Windows XP and Version 4 of Windows NT once ran on POWER technology. In 1999, Microsoft decided that Windows NT would no longer be updated for processors other than Intel and DEC Alpha and it stopped development for the IBM PowerPC chip.

Knowing this history, it is clear that there are no technical reasons why the Windows Server operating systems could not be up-tuned to run again on POWER technology. In fact, many speculate that Microsoft already has XP running on POWER and is just waiting for its negotiations with IBM to complete. There would be no reason at all why Microsoft would not like to enjoy the benefit of the solid, reliable hardware base in the all-everything machine so that Windows would be able to scale substantially better than in the Intel line.

Of course, once the mainframe z/OS and Windows XP run on the i5 or i6, IBM can change its name to the all-everything machine for indeed it will have become exactly that. Add these two to the list above and you have a machine with two additional capabilities as the following:

8. Windows XP Native Processing
9. Mainframe z/OS Native Processing

Now, that's an all-everything machine from a hardware and OS perspective if I have ever seen one. At this point of the game, the all-everything machine would be able to run all applications from all operating systems. Moreover, since the applications would be from four different environments, it would be proper to conclude that the all-everything machine would be providing four times the business value of one machine. That sure is a lot of everything for one machine to handle by itself.

Chapter 7

Autonomic Computing from the Start

Automatic Transmissions 'R' Us

From the very beginning, the AS/400 was designed to be simpler than all other systems. To this day, no other platform has such a good balance between "easy-to-use" and "powerful." Unlike Mainframes, Windows, and Unix/Linux, the AS/400 in the form of an iSeries or an i5 today comes without a clutch. It's got a fully functional automatic transmission. In fact, when you drive an i5, you would find that for the most part, you are not needed; the system drives itself.

You can know enough to run an i5, when you know less than a few percentages of what there is to know. With the i5, for example, much of what you want to do is already set up with default values, and thus, you do not have to think out each piece of a command. You just run it. With a minimal amount of training, one person can in fact know enough to run an entire company using an i5. It's done all the time. That's why once people have worked with an AS/400 or i5, with OS/400 or i5/OS, they become spoiled and resent working again with other machines.

In basic no-frills form, the i5 is hard to beat for a new install of a reliable system at any new customer location. PCs are still for fluff things such as e-mail clients, drawings, and things requiring really cheap connectivity. You may not yet want to surf the net on an i5, but you surely would not want to trust a fully audited, transaction-controlled, mission-critical invoicing application running on behalf of 100 users if it were written in a PC-oriented kids language, and if it were running on a farm of Windows PC servers with 20 label printers in multiple plants. For this, you need a nice sized professional staff if the application is for a PC-based system. Why would anybody do this with a PC-based system? If the system were

an i5, just one person would be able to handle the mission, and the person would also be able to take lunch.

Part of how the i5 is able to get lots done in a reliable fashion is that it is much easier to use, and its rules are stricter than any other environment. Hackers don't like rules, so for the most part; they stay clear of the i5. On other platforms, for example, you can write a program that destroys the system itself. You can do it intentionally as a hacker, or you can do it by error, unintentionally, because you did something wrong. Most of us have seen the ease with which viruses can be created on Windows systems and how hackers break into Windows and Unix boxes all the time. The eServer i5 prevents this within its architecture. It prevents users from killing themselves. It is not unimportant that the techno-geeks don't like it as much as they like Unix or Windows. They get stopped at the door like a wolf and a brick house. They can't hack the i5 and bring it down successfully--and they really don't like that one bit!

Ease of Use for Technical Staff

AS/400-iSeries professionals love the ease with which they can manage the i5 system and its relational database facility. On mainframe computers and Unix boxes, and even Windows boxes, it is not quite so simple. For example, on all three of the non-i5 flavors, the database is not integrated. That means that you get to install it, apply the patches, and ensure that it is fully functional. For the record, Oracle database administrators, which are needed in heavy database environments, get paid a ton of money. Oh, they are worth it all right! Without them, your Oracle database would be crashing as often as a Windows/ME PC. See Skip Marchesani's comments on Oracle on Page 35.

With this environment, you get to make sure all the pieces work. You get to integrate it with everything else on your machine. Moreover, as noted above with Oracle and SQL Server, in order to have a database, you have to hire an expensive extra person to your staff. This new person is called a database administrator (DBA) and he comes with a price tag of more than $80,000 per year. Whatever business value a system with a database provides, the extra care and feeding and the extra staff quickly chip away at that value.

A DBA is not just needed on a mainframe. When A PC server is used for real business applications, a DBA is required on this inexpensive platform

as well. Moreover, on the PC platform, you always install servers in pairs, in case one goes down. So you get to do the DB installation work twice. If you know of any advanced PC shops with database products that do not have a DBA, you also know they're not doing too well. Though the i5 is a database machine, you need no DBA because the database is built into the machine. Many users and even programmers discover that they have been connected to a database long after their applications have been using it successfully with the i5 family for years.

The i5 Keeps on Ticking

Internet and i5 - oriented magazines have many wonderful stories about how the machines just go ahead and get their work done, regardless of the level of attention the servers get. The new i5 is very much like a good old Timex watch. Sometimes, however, i5 units keep on ticking long after they are forgotten. For example, this story relayed by Mark Villa of Charleston, South Carolina, is one that brings the ease of AS/400 operations picture well into focus.

> "There was an AS/400 in a plant that was doing its thing on a regular basis, and it was basically unnoticed out in the plant. Unknowingly, the company built a wall in the area during some construction, and someone went hunting for the AS/400 [i5 family] months later, and found it was enclosed."

That quickly gives us an idea of how much constant care an i5 database requires.

Runs Many Applications At Once

Unlike Windows Servers, i5 machines run many applications at the same time. Windows servers do not do well when used for more than one function. That's why a single-server PC grows into a small farm of PC servers almost overnight. Today's i5 can be a Web server, a Domino Notes server, a Java Virtual Machine, a Windows NT server, an OS/2 server, a firewall, an invoice machine, an accounts receivable machine, and

so on -- all on the same single-processor box, without even having to partition the unit. With partitioning, of course it can also be a Unix Tuxedo Application Server, or a Linux application server. More industry analysts are noticing this facility and giving the i5 very high marks in their total-cost-of-computing analyses. There is a high cost to run a server farm as each machine needs attention. Additionally, the more machines you have in the 'farm,' the more likely one of them is down right now.

The i5 can actually be a server farm under its one set of covers with just one server box. It can also provide the same facility for Windows servers as a storage area network (SAN). Because the i5 is so many machines in one, sometimes it gets no credit from the industry press for being any, when it is actually closer to all than none. From its inception, IBM highlighted the i5 family as its workhorse of midrange servers for business. IBM called the early AS/400, for example, its midrange business system. It still is IBM's finest business system in its newest form, the IBM i5. When all the pieces (mainframe and Windows processing) come together very soon on one i5, it will be the indisputable all-everything machine.

> Technical Note: A SAN is short for Storage Area Network. This is a modern notion involving the separation of the data storage elements from single computers and the centralization of that data on a central disk server, the role of which is storage management. A topology would show many servers all accessing data from the same set of disk drives managed by the Storage Server in the Storage Area Network. Because many Intel servers can be installed as blades in an i5, the i5 box itself serves as a SAN for Windows Servers at 10 to 15% of the cost of a typical SAN approach. Considering that reduced cost is one of the most typical and most quantifiable categories of business value, one can see the impact that an i5 SAN, instead of the "farm," can have on the bottom line.

Today the AS/400-iSeries, in the form of the i5 is still alive and kicking, with an installed base of more than 400,000 and perhaps as many as 750,000 systems in about 250,000 businesses around the world. Between 30,000 and 90,000 new systems are sold each year, according to industry analysts. The i5 continues to be successful because many of its customers buy a new one every four or five years, and because IBM continues to

enhance the product line to levels far exceeding all other machines on the market.

Old Reliable

The most cited reason behind the continuing popularity of the i5 heritage line is its reliability. The unprecedented ease of programming, ease of use and the low cost of management follow right behind. The i5 models continue to be out-of-the-box products with bundled applications, communications software, and an integrated database. No commercial system requires the small amount of care as an i5.

Ease of Migration

The system provides the ability to integrate new technologies with very little disruption to business operations. I5 heritage users have been benefiting for many years. For instance, Pagnotti Enterprises of Wilkes-Barre, Pennsylvania, a holding company for some mining and insurance businesses, replaced its old AS/400 CISC architecture system with a 64-bit RISC system in 1999. The company's RISC machine is now old enough that the company is looking again. Despite the magnitude of the 1999 shift, resulting in a major performance increase, no changes were required to the application code or logic, according to Betty Carpenter, IT Director for the company.

"The conversion to 64-bits was as simple as restoring the objects on the new system," said Carpenter, who has worked on AS/400s for more than a decade. That's why AS/400-iSeries customers do not want to switch.

In 1988, IBM launched the AS/400 to replace its aging System/38. Over the years, IBM has kept many of the original features but adapted the overall system to the technology changes needed for the times. Over these 27 years, counting the System/38 years, IBM also has succeeded in making the platform far more open than anyone ever would have expected. For instance, the i5 today offers native support for mail and messaging technologies, such as Lotus Domino and ERP, from companies such as SAP, PeopleSoft, and Baan.

The IBM i5 has become a mainframe in size at the large end, and a mainframe in capability on all models. Super mainframe capability can be seen in a concept called *logical partitioning* on the IBM i5. This feature was borrowed directly from the mainframe. Using this facility, an implementer can define one i5 as if it were many i5s, and each one can behave as a separate machine. Moreover, one i5 may be running i5/OS, Linux, or IBM's AIX at the same time. The future is wide open. In private meetings, IBM has announced that Bill Gates would like Windows to run on an i5, and IBM has not ruled it out.

How Popular Is the All-Everything Machine?

Besides my little cadre of i5 customers in Northeastern Pennsylvania, there are several hundred thousand others. Of course, I think they all should be my consulting customers, but I am happy with what I have got. A few national and world-class IBM i5 heritage customers, last time I checked, include the following:

Enterprise Rent A Car,
with over 40 AS/400s, 20 of which are dedicated to handling an application with 1.3 million transactions each hour.
Ball-Foster Glass Container Co.
in Muncie, Indiana.
J&L Fiber Service
in Waukesha, Wisconsin, a materials supplier for the paper industry.
Cornerstone Retail Solutions
in Austin, Texas.
Bergen Brunswig Corp., a pharmaceutical distributor in Orange, California.
Saab Cars USA,
Inc., in Norcross, Georgia (U.S. headquarters).
AppsMall
(AppsMall.com) in Rochester, Minnesota.
Klein Wholesale
in Pennsylvania, the fifth-largest candy and tobacco wholesaler in the United States.
Marywood University,

Liberal Arts higher education institution in Scranton, Pennsylvania.

Better than half of all i5 heritage machines are installed in countries outside the United States.

You'd have to pry an IBM i5 away from its users with the biggest crowbar ever invented in order to create some separation. Check out this comment from a leading IBM i5 news company, NewsWire/400, of Penton Media:

> "We've been running our Web site on Domino on the AS/400, and we're not even running on the latest and greatest platform. We're running on a [model] 50S. The beauty of it is, the thing never goes down. Our maintenance on it is almost nil. We don't do anything with it; it just runs."
>
> --Terry Bird, principal, Appsmall.com

It's not just the i5-biased media that pump the i5 line from time to time. In an *InfoWorld* article on July 31, 2000, just before the rebranding of the AS/400 to the iSeries, Maggie Biggs, writing for the "Enterprise Toolbox" section of *InfoWorld*'s e-magazine, noted that the industry perception of the i5 family seemed to be changing.

In her article, Biggs discussed the changing perceptions as the traditional i5 family morphs into what she calls a powerful, dynamic e-business server. The article was published a few years after IBM had stuck the little "e" on the back of the AS/400, making it the AS/400e. While writing the article, as a matter of course, Ms. Biggs felt compelled to suggest that IBM start marketing the box more aggressively.

> "Actually, the AS/400 has been e-business-ready for several years, but it's nice to see the marketing folks at IBM finally catching up with the platform's technological advances."

Biggs continues:

"Our experts from the Test Center and Info-World Review Board (made up of our free-lance writers) examined the newest release of the AS/400 and its operating system, OS/400 [now i5/OS]...

"After more than 10 years of advances and a metamorphosis into a beefy e-business server, the majority of people still view the AS/400 [i5] as a legacy platform. This is a shame because the AS/400 [i5] is a multifaceted server capable of fulfilling a myriad of business needs regardless of the size of the enterprise or the tasks that are thrown at it. And the AS/400 [i5] continues to be one of few platforms that can simultaneously support legacy, client/server, and Web-based computing.

"...what kind of ROI you might expect to gain by adopting the AS/400 [i5]... found the costs low when compared to the software and hardware capabilities of the platform, which stand out favorably in many ways when measured against competing servers...

"These servers can be configured to meet the requirements and budgets of businesses both large and small. IBM has enabled technologies that let you run both Unix-based applications and Windows NT and Windows 2000 applications within your AS/400 environment. You might use these technologies to consolidate servers, reduce expenditures, or to improve business process integration...

"From what we experienced during our testing and analysis, the AS/400 [i5] appears ready to provide some stiff competition for its server rivals. You may not hear about the AS/400 [i5] as often as you might hear about other platforms, but just ask any of your colleagues who have worked with the platform and I think you'll hear a positive response."

Amen!

As the client/server revolution went sour and Windows server farms began proving to be more and more difficult and expensive to manage, there has recently been a definite resurgence of interest in the i5 server, fueled mostly by word of mouth. Businesses seeking a reliable, scalable platform are starting to notice that out of all the technology that is inside the i5, the bottom line is that it works and for the most part, it does not go down.

Though it would be good for IBM to let the word out, most i5 heritage shops are not complaining about the ninth generation, 64-bit architecture of the box, in that it continues to benefit from Big Blue's ongoing, multi-billion-dollar investment in i5 technology.

IBM I5 Waiting to Be Successful

The IBM i5 is poised to become the flagship for IBM once again as IBM completes its transition to the all-everything machine and chooses to hoist the flag. Besides having the most elegant packaging of computer basics, its features include enterprise e-commerce applications, native support for key Web-enabling technologies, such as Web servers, Java, Lotus Domino, and IBM's WebSphere server.

Not to be outdone by the big jobs, the server also boasts support for Windows NT, Windows 2000, Windows XP, and Windows 2003 application serving through special bolt-on Intel Processor logic cards that are installed inside the i5 chassis.

The free operating system shipped with the first processor of every machine is on duty from the moment you turn it on. The Windows process of installing the base operating system and then adding all the Windows fix packs is not necessary. The i5 operating system, originally known as Operating System/400, or OS/400, and now known as i5/OS, is pre-installed, and is tested for hours before shipping. As you would expect, like the Spaghetti ad, as you list features that an operating system should have, when you talk about the i5 operating system, you'll find yourself saying, "It's in there!"

Before I close this chapter, I would like to present a quick laundry list (Figure 7-1) of some of the advanced facilities that you will find in your average i5. If you are not technical at heart, it may not be too meaningful. However, the list at least gives an idea of the i5's full capabilities to solve business problems and to provide solutions in many areas that might not at first be obvious

Figure 7-1 Some Major i5 Capabilities

- 64-bit POWER5 RISC-based architecture – IBM's most powerful RISC processors.

- 128-bit software architecture.
- Spooling and job management for multiple users/separate queues.
- Performance management for allocating resources.
- Single level store (i5 unique).
- Technology-independent machine interface (i5 unique).
- Integrated DB2 Universal Database (i5 unique).
- Capability-based addressing for integrated security (i5 unique).
- Object based (i5 unique).
- Clustering--integrated.
- Apache Web Server (HTTP) Server--integrated within system.
- Web search engine.
- Enhanced TCP/IP stack and utility--integrated within system.
- Encryption.
- File serving and client/server integrated features.
- Logical partitioning--advanced system facility.
- GUI application development tools for client/server and Web.
- Intel integration--Windows under the covers.
- Etc., etc., etc.

It's quite an all-everything machine.

Chapter 8

Advanced Concepts in the All-Everything Machine

The IBM i5 Can Do It!

From traditional code crunching to Web services support to Linux, Unix, Windows, and even autonomic computing, the often-underestimated IBM eServer i5 platform can match any IT environment. This truly all-everything computer can do it all.

If you strip from the newest i5 all of the fancy stuff the press seems to be excited about, such as client/server, ODBC, Linux, Windows, logical partitioning, AIX, PASE, QSHELL (Unix CORN Shell), and Java, you are still left with the most elegant, most functional, and most powerful server in the world. It is just waiting to be loved by the masses. Along with a number of other graying IBM i5 lifers who worked with the advanced technology of the System/38 after its announcement in 1978, and saw it become the AS/400 and now the i5, we know that there is no computer that can top the i5 for pure architectural elegance.

In Chapter 1 as you recall, I introduced the architectural elegance and advanced computer science facilities that are built into all models of the all-everything machine. In just a partial list, as you may recall, I identified 42 high profile business value factors for executives along with 60 major technical factors that demonstrate the efficiency and effectiveness of this killer technical machine to the IT department. I am now compelled to ask the question: why shouldn't an organization be able to have no-sweat, low cost high function IT facility while providing the IT department with tools that make the whole thing easier than it has ever been? The answer to that question is simple: There are no reasons to not want an IBM i5.

In Chapter 1, we identified a number of features that affect the bottom line in terms of benefits, costs, and organizational productivity. In this Chapter we highlight just six of those factors and we put substantial meat on these feature bones to help the most doubting of the Thomas's to understand why this all-everything machine provides so much benefit for so little cost.

To Know the i5 is to Love the i5

There is no reason not to love the i5, if you really know it. So I might be so bold as to suggest that the Teddy Bears, a musical group from the 1950s, would have taken notice of the AS/400 in 1988, if the non-IT world were in on IBM's secret weapon. They would have been able to capitalize on a great theme to reenergize their group for a new hit tune to meet the times. Yes, the Teddy Bears could have taken Phil Spector's hit tune and adapted it to the computer world back then and again today by changing just a few of the lyrics: "To know, know, know the IBM i5 is to love, love, love the IBM i5!" I know I do as do many others who use it every day. And there are big reasons for that.

Twenty years after the song, starting in 1978, with the introduction of the System/38, followed by the AS/400, the iSeries, and now the i5, this not-so-well-known IBM server parlayed **advanced system architecture** while never abandoning the notion of **small system ease-of-use**. That's another way of saying you get the error-free, function rich, highly secure computing model that the big companies get with mainframes, but you get it with a personality that's fit for a small business at a cost that a small business can readily afford.

The purpose for this duopoly was to enable powerful customer-oriented applications to be built that would last long into the future, without having to be scrapped or reengineered. If there is any legacy that the IBM i5 possesses, this is it. However, because software code runs forever and for better on this platform, competitors and the Windows-dominated press have chosen to call the IBM i5 itself a legacy system. Yet, if called to task, no industry expert could deny that the i5 is an "all-everything" computer. It is the best, the most productive, and the least cost all-around commercial system that has ever been conceived and built. And, that does not sound much like legacy to me.

IBM i5: Six Advanced Principles

The IBM i5 is the only server you can buy that offers six major advanced architecture facilities as part of its standard, integrated offering. The purpose of this book is not to teach the IBM i5 per se. However, in order to gain an appreciation of this computer system, some things are helpful to know. There is no other commercial system or server that has been able to deliver even one of the below advanced architectural properties. At the core of the i5's machine and software architecture are the following six advanced computer science principles:

1. Integrated system functions
2. High level machine
3. Single-level store
4. Object-based architecture
5. Capability-based addressing
6. Integrated relational database

Because IBM did not announce a seventh principle, I chose not to include this next item in the above list, but, from my perspective it belongs there because it is part of every i5 and it helps make the i5 system a programmer's dream. This principle is explained in Chapter 9, after all of the six advanced principles are fully explained. I would call it principle 7 as follows:

7. Integrated transaction processing

These seven features provide a platform that is renowned for flexibility, large system function, ease-of-use, and non-disruptive growth. To help you get a better appreciation for what these mean, without hurting the non-technical brain along the way, let's take a quick peak at each of these six principles in turn. In Chapter 9, we will examine principle 7 in detail.

Integrated System Functions

The traditional approach to gaining computer function has always been to use add-on software. If you need a database, you buy one. If you need a

transaction processor, you buy one. If you need language compilers, you buy them. For example, you may have heard of Tuxedo as a transaction processor facility for Unix and Windows and CICS for mainframes. You may have heard of Oracle or Microsoft SQL Server in the database area. Moreover, you may know that Bill Gates' company from Redmond Washington got started making language compilers and that Microsoft makes a lot of money in these areas even today. For example, you may have heard of Microsoft C Language, Microsoft Visual Basic, and Microsoft COBOL. All of these are separate products that IT professionals get to install and make them operational in IT shops. To get them, you just have to write Microsoft a check for each one you want.

This traditional approach is an ala carte approach. You never get a full dinner. In fact, as you read the above paragraph you get the idea that the parts of the dinner are coming from different restaurants and so, there is no guarantee that they will fit nicely on your one plate or all be ready at the same time. I like to call this traditional approach to computing *legacy computing* since its style dates back to the 1950's. Most vendors in this legacy software space work with the Unix and Windows operating systems. They have found it easier over the years just to add software function patches and sell them as new products, rather than start over and design the operating system the right way. The System/38 and its successor products changed this paradigm for IBM and for the industry for good. However, Unix and Windows have still not abandoned their legacy ala carte tradition.

No Systems Programming

To put the patchwork quilt puzzle into perspective, it helps to know that there still exists a function in IT called systems programming. In many ways, systems programmers finish the computer vendor's work in the IT shop. When as many as 40 or 50 essential products have to be installed, tailored, configured, and continually monitored, you can bet there is a high-paying job opportunity available for a highly technical person. The systems programmer position which was introduced in the 1960's in IBM mainframe shops is now required in Windows and Unix shops to assure that all of their heterogeneous piece parts fit together well enough to run the data center.

Systems programmers may be called Windows Certified Engineers in some shops but they are merely systems programmers, a throwback to the

old legacy computing days. They don't write programs or add value in any way to the IT shop, yet they are essential because they take piece parts and build and maintain operating systems and software applications on the IT shop floor. Without their efforts, of course, there would be no completely installed servers with which to work.

Only in the most complex, multi-system environments is such a position required in an IBM i5 shop. I know of no small businesses using IBM i5 technology that need a systems programmer. IBM ships the system complete. That is a huge cost savings and it also increases the productivity of the organization because the system is already built when it arrives.

Unlike the Windows and Unix piece parts approach, one of the major design criteria for the 1978 System/38 was to ship a complete product to IBM's customers. The System/38 was designed not to need additional time, effort, or skill for its completion. That's integration. The great grandson of the System/38, the i5 uses the same integration paradigm.

The Best of the Future

Using IBM's famous Future Systems (FS) project design (Chapter 12) concepts as a basis, IBM's lab in Rochester Minnesota spent most of the 1970s building the System/38. IBM had studied the best possible architecture and ingredients for a new system replacement for its mainframe processor line. After being designed for the mainframe division, this advanced architecture became the foundation for the most advanced computer system ever built: the IBM System/38. Integration was at the forefront of this advanced design notion. If announced today, the 1978 System/38 would undoubtedly be the fourth-most-advanced computer ever built. It would follow the AS/400, the iSeries, and the IBM i5.

When you build a computer system in which the hardware, the operating system, and all of the support programs for program development and operations are all built together, you can build a system in which function is distributed to the proper layers and components. You can achieve integration, smaller code paths, better performance, better stability, more productivity, and less functional redundancy. Everything a developer needs in order to be productive can be built together. IBM announced and made available the most advanced system of its time with the

introduction of the System/38 and you can acquire this technology today under the name IBM i5.

No longer did system programmers have to spend hours determining what versions of what products could be built together in a complex system generation process. For the first time, every system model in a computer product line had all of the functions. From top to bottom, every System/38 could be used to build and to run the same application programs. It was in there! It still is with the AS/400, the iSeries, and now, the IBM i5.

Earlier in the chapter we used the dinner analogy of a full dinner vs. ala carte. Another worthwhile analogy is the notion of a house in which the pieces are all designed and built separately by different companies. What if BEA Systems (Tuxedo) built the bedrooms, and Microsoft (Windows) built the bathrooms, and Oracle (Database) built the living room, and Intel (Pentium IV) built the basement. Since these companies do not share one design for a house, but instead use their standard rooms, it is highly likely that when it all comes together on your lot, there will be some anomalies. Since they merely rearrange their standard offerings for different housing needs, it is understandable that the rooms can't all blend well when they eventually come together for the first time.

Surely in this design, you can expect to need a highly paid contractor/builder to get the electricity and plumbing working right, cut doors where there are none, steal pieces of rooms for hallways, line up the steps to open spaces, etc. The same inefficiencies that you see in having home parts built by separate contractors unaware of the total design of your house are prevalent when computer vendors try putting their disparate piece parts together in your computer room.

In computer shops where the four vendors above actually do install their wares, the contractor/builder works for your company, not any of the four whose products you are using. This contractor /builder person is the systems programmer and the house is not complete until he finishes his work at your expense. Unlike a house, however, he does not go away when it is completed. He's on the payroll for the long haul. Because piece parts that fit together well one day in a non-integrated computer shop may not operate well with tomorrow's updates, the systems programmer is essential to making the system work after it crashes, gets whacked with a virus, or simply hangs.

Thus, these guys get hired and they stay on and are part of the ongoing expense until you choose a different paradigm for computing, such as an integrated, custom built once and for all approach. Besides the indisputable fact that it is so much more productive and cost effective, it just makes common sense to have the whole house built together. The new paradigm is integrated computing and it is available today with an IBM i5. It's like having the whole house built together.

High-Level Machine

Quite simply, a high-level machine implementation works in favor of the user, rather than the computer designer. Low-level machines, such as Unix, mainframe, and Windows operate with languages and interfaces that are machine-oriented, not people-oriented. If you like talking in ones and zeros, you'd like the lowest level language--machine language. A high-level machine is another way of saying that user functions are built into the machine without having to worry about the machine itself. In many ways the result from a System/38 or any of its successors is a system-managed system, rather than a user-managed system. A high-level machine is like a high-level language, in that you talk to it in all ways at a level far away from the ones and zeros and the bits and bytes. Thus, this advanced notion brings with it a tremendous increase in operational and system productivity.

Access to the vast array of advanced system functions on the IBM i5 is provided by a powerful, consistent interface, or **high-level machine interface**. IBM calls this interface the **Technology Independent Machine Interface (TIMI)**. Computer scientists would label the high-level machine interface as a full **abstract machine**, since the architecture of the machine that you believe you are working with is only visible at the high level. The actual low-level hardware looks substantially different, but the user or programmer never interacts at the lower levels with the machine.

Frank Soltis, the iSeries Chief Scientist talks how the TIMI came about in many of his speeches and in his published works. The TIMI was an early design decision. Moreover, at about the same time, the S/38 architects decided that the hardware would not interpretively execute the TIMI architecture. Considering that processors were substantially slower in the 1970's when these decisions were made, it was clear that using hardware

to interpret such a high-level instruction set architecture (ISA) would not provide the level of performance needed for a commercial server. Moreover, since most commercial applications are executed over and over again, the translation cost would have to be paid too many times for the notion to be efficient. As programmers know, interpretation as a method, is most useful when a program is to be executed once or only a small number of times.

Because the TIMI would not be directly executed, the architects had to design another lower-level ISA that the programmers above the MI would know nothing about. This "second" ISA had to be created so the hardware could execute. Programs at the MI level would be translated into this lower-level ISA before they were executed. For performance purposes, this translation would occur only once. The translated machine code, along with the original MI version of the program in its template form, would then be stored within a program type object for future use.

Before the grandfather of the all-everything machine (IBM i5) was re-oriented to 64-bits and RISC hardware, it used a CISC ISA as its executable interface. So did its predecessor, the System/38. In 1995 this changed when IBM made the 64-bit RISC hardware modification to the AS/400 model line. The CISC ISA was typical of the ISAs of the 1970s and 1980s, and in many ways, it was similar to the hardware ISA in IBM's mainframes and today's Intel machines. In 1995, IBM changed the hardware and the executable interface on the AS/400 to a 64-bit modified PowerPC ISA. The operating system above the MI continued to work. Today's POWER4 and POWER5 processors in the IBM i5 family implement this same ISA.

The benefit of this overall virtual machine design is that the hardware ISA can change dramatically, as it did in 1995, with no changes required for operating system or application programs. It is worthy to note that no other commercially available system in history has ever been able to accomplish this feat. That includes the ever popular Intel, Windows, Linux, and Unix flavored machines that dot the computing landscape of today.

Windows and Unix machines are not object-oriented; nor are they object-based. Though Windows NT and its follow-on versions, 2000, and XP also have a hardware abstraction layer, it is not nearly as comprehensive as the TIMI approach as used in the i5. If it were built as well as the IBM i5,

Microsoft and Intel would not be struggling today to be able to use the full power of the Intel 64-bit chip in Windows.

Change Made Painless

Even as I write, Microsoft operating systems continue to waste half of the capabilities of the chip as 32 of the 64 bits remain dark and unused in the Intel 64-bit Itanium processor. If the Windows "hardware abstraction layer" were fully implemented as in the IBM i5 TIMI, all 64-bits would be lit up in short order with no programming sweat. But it is not. Moreover, the same goes for Windows applications. Since the OS cannot deliver 64-bit computing, Windows applications run at 32-bit speed on the 64-bit platform.

i5 programmers love the notion of the TIMI, and they don't want to give it up, because they don't want to have to learn cryptic machine code and silly names for normal functions. Anything less is inferior. Even the machine instructions are more like the spoken word, or as we say in the United States, English-like. The interface is at such a high level (more human than machine) on the AS/400-iSeries that machine instructions, not add-on packaged programs, are used to retrieve and update database records, perform multiprogramming, handle storage management, query database files, and create indices over DB files.

Having said all that, as noted above but worth repeating, one of the biggest benefits from a high-level machine interface comes when you are changing hardware. For example, when IBM changed its AS/400 hardware in 1995 from a technology known as Complex Instruction Set Computing (CISC) to the IBM-invented, industry-leading Reduced Instruction Set Computing (RISC) model, even though the hardware was completely different, the company was able to use the TIMI to get the operating system functional without a rewrite for the new hardware.

No OS Rewrite Necessary

Only the very-low-level microcode (IBM calls this licensed internal code.) had to be touched, and this represented less that 5 percent of the code and it existed below the TIMI. The microcode portion presented the hardware machine personality to the operating system. IBM had written the original operating system, called Operating System/400, now i5/OS using the high-level machine interface. Since OS/400 spoke to only the

high-level TIMI, it remained virtually unchanged even though the processor type and the number of bits had changed.

Immediate 64-bit RISC Processing

OS/400 "knew" nothing of the processor architecture. So when the processor architecture was changed from CISC to RISC in 1995, and the hardware instruction set was redesigned, and the architecture shifted from 48 to 64 bits, the operating system programs did not have to be modified. They ran the same after the hardware change because they were always shielded from the actual look of the hardware. They were based on the high-level interface, and therefore continued to run. More importantly, for IBM's AS/400 customer programmer community, the millions of System/38 and AS/400 compiled programs, written by IBM customers and software vendors across the world, were enabled to run, unchanged with the new AS/400 RISC platform.

From a business value standpoint, this feature provides for innate investment protection. Program code written for System/38 computer from the 1978 era runs today on an i5 without recompilation. The TIMI uses a self adaptation scheme with an imbedded program template and the TIMI re-encapsulates the older program using the new interface. Because IBM can change from 48 to 64 to 128 bits and from CISC to RISC and programs do not have to be rewritten and packages do not have to be scrapped or reengineered, there is a tremendous cost savings for the firm to not have to find new software and to not have to disrupt operations in order to migrate to the new wares.

While IBM was changing its AS/400 line hardware to RISC in 1995, it did one more thing at the same time that is historically significant. The company introduced 64-bit processors. Suffice it to say these were much bigger than the Windows and Unix and mainframe 32-bit processors that existed in 1995. Another point in all of this is that the more bits one instruction can carry in one machine cycle, the faster the machine. All this change occurred in 1995, ten years ago, and the technology was immediately available to AS/400 customers and now i5 customers, without even having to recompile their programs.

IBM achieved this in a very short time because of the nature of the TIMI. It took Intel until the year 2000 to create a 64-bit processor. The first Intel 64-bit processor did not run well until later in 2001, and as noted

previously in this chapter, Windows still cannot use all 64-bits. Windows is still saddled with using 32 of the 64 bits. Windows 2000 will never be 64-bit; though it is still possible that Microsoft will eventually get its 2003 offering working with 64 bits. By then, it may be Windows 2005 or 2006 or 2007. As a point of note, IBM's mainframe division finally got its 64-bit processors out in late 2001. So, even IBM's premiere computing division was behind the all-everything machine by six years.

TIMI Saved Users and IBM Lots of Time

All of the time it took other companies to try to get to 64-bits was saved by IBM in the CISC to RISC conversion because of the TIMI. Though all of the technology changed, the interface to the existing operating system did not have to be rewritten. That is a significant advancement and will be the same as IBM moves toward 128-bit hardware implementations in the future. The TIMI gives the IBM i5 a big, big technology edge.

Therefore, in addition to making everything on the system easier to work with, the high-level machine interface protects the programming investments of software companies and IT shops by enabling existing programs to run on new hardware without having to be rewritten. Try that with Windows or Unix!

Why Should Programmers Like TIMI?

The TIMI means a lot to a programmer. The fact of the matter is that in the TIMI architecture, the language compilers unlike other machines do not really generate executable machine code. They generate an intermediate pseudo machine code stored as a "template" in the object that will contain the executable code. The first time the program is run, TIMI compiles the template and generates the actual machine code and stores it in the program object.

This comes in real handy when the operating system environment or the hardware changes. The TIMI looks at the object and detects that it is not compatible with the new environment. Rather than punting as would happen in Windows or Unix environments, the TIMI regenerates the machine code. This is one of the key points about TIMI that provides programmers a big plus compared to all other systems. Moreover,

investment protection is assured since program code works almost forever in this environment.

It is the template and abstraction between the logical representation of language code and the physical implementation on the machine that enabled IBM to move from 32 bit to 64 bit CISC to RISC without requiring a programmer in an AS/400 shop to have to change a line of code.

Rather than take a shot just at Unix or Windows /Intel, or Linux, though they deserve to get their shots, I will use an IBM mainframe as the focal point for this next example. Please note that much of what I say also applies to the other three OS environments. I happen to be friends with a mainframe guy who had to make a transition in the 1980s from IBM's MVS/ESA OS. At the time, the System/370 hardware architecture was being upgraded from a 16-bit architecture to a 31-bit architecture.

At the time, on the old System/370 machines, the addressable space on the system was just 16 megabytes of memory. When IBM moved to a 31-bit architecture they expanded the size of the programs and the address spaces to over 2 gigabytes of memory. IBM worked very hard to prevent mainframe programmers from having to modify or recompile their programs, but just as with Windows, there was this notion of above the line and below the line. Programs could not access memory "above the line". The 16bit code would run fine, just as Microsoft's 32-bit code works fine on 64-bit machines.

However, if the programs really needed memory, (memory constrained) programmers had to modify the mainframe code and recompile programs using the 31-bit compilers and linkage editors of the day. This was as much as ten to twenty or even fifty times greater than the effort for an IBM AS/400 customer to move from 48-bit to 64-bit technology and from CISC to RISC.

As many already know, in the Windows world, this has been the reason for the delay in Longhorn. Microsoft and Intel are going absolutely insane providing a compatibility box to allow 32-bit code to run on a 64bit processor. They can't get the compatibility box to work. They know that they must get the box to work or they can't ship their product. The machine must support existing users for 3 to 5 years before vendors will rewrite their applications to leverage the 64 bit architecture. So we all

know how far behind in the 64-bit game Windows actually is. Moreover, the worst news for Microsoft is that it looks like it's not going to be fixed for Windows any time soon. Now, if only they had a TIMI!

Hewlett Packard faced this same situation. They actually shipped a 64 bit machine (DEC Alpha) long before the iSeries did. To this day most HP customers still cannot leverage 64-bit applications. A huge percentage of their customer base is running old 32 bit applications. Since a majority of the HP code is written in C++ they must manually rewrite the code.

The growth for the iSeries, because of TIMI is virtually unlimited. Let's say that IBM moves to 128-bit hardware and ships the 128 bit beast from hell tomorrow, every object program migrated to the new 128-bit machine will have its templates automatically regenerated into new executable machine code and the old programs, without rewrite or even a touch will immediately be able to leverage the full power of the hardware.

This fact alone makes the iSeries a killer machine! But, since we are not looking to kill anything in this book, we continue to call this powerful inanimate animal, an all-everything machine.

Single-Level Store

Many readers may already understand the notion of virtual storage. It has been used in computer systems since the very early 1970s. Virtual storage permits computers to run programs that are bigger than the memory of the machine itself. It does this by permitting memory to be over-committed, running many different programs. It uses the disks on the system to store pages of programs that are not being used at a particular point in program operation. This has many advantages, including not being shut down when the system has inadequate real memory resources. **Single-level store** takes the notion of virtual storage one step beyond.

Single-level store, as with all of the advanced techniques we are exploring, was first introduced with the System/38. With single-level store, a System/38, through the TIMI, believes that all of its objects exist in a 281-trillion-byte memory continuum (based on just a 48-bit hardware address). That's pretty big!

It does not matter with single-level store whether the data actually resides on disk, bubble memory, or bubble gum; though today the storage devices

are limited to disk technology. In 1979, I recall giving my first presentation about the System/38 as a systems engineer with IBM. The presentation guide suggested that the 281 trillion bytes represented the sum total of all of the disk drives that had ever been built at that time. I was impressed, for sure. It took mainframes 20 more years longer to reach this level of addressability.

Auto Managed Disk Pool

As part of the single level store implementation, of course, the System/38 actually used disk drives for hardware. To be able to deliver an image to the user that there was no hardware disk since memory was just one big continuum, the OS designers first had to create the notion of system managed disk pools. No matter how many disk drives (up to 2700) reside within an i5 system, the operating system is written to treat them as one disk drive as it carves out space and places objects on the disk platters in a manner that automatically optimizes system performance.

So, the user is shielded from having to assign files to drive letters and drives never run out of disk space (shutting down the system). Then, at an even higher level, the system thinks that it has no disk and that all memory is managed in a flat memory model. Whether it is an IBM mainframe, Unix/Linux, or Windows, managing disk and memory is an arduous task for a systems implementer.

It is a huge issue and it steals away much time from a systems programmer. This task is so complicated on larger servers that the things one needs to know and do to allocate memory and disk could take about a week per month every month – just in analyzing disk allocations and storage utilization. In the mainframe world, this function alone often justifies hiring a full time person.

To get performance, the systems programmer would have to allocate tracks on a disk to position high use sections of a file to minimize disk and arm movement. Managing the entire disk pool was manual. Every single track of disk space had to be manually allocated. If there was a sudden shift in usage patterns or a major increase in business activity, all the balancing work went into the "toilet" and the systems programmer had to start over using performance reports and brute force analysis.

Even if you were one of the very best system performance people in the mainframe world back then, it would take almost forever to keep things right. The i5 does it all; from managing disk allocations to spreading out files so that they are optimized for performance with automatic allocations across multiple drives. Additionally, the i5 continually rebalances all of the disk segments automatically.

In the Windows arena, Windows, just like the i5, manages the disk pool, but Windows is not so good at it. If you have high activity adds and deletes, for example, you lose disk space until you run a defrag on your system. Of course, you must do the defrag when all users are off the server, including those coming in from the Web. Additionally, depending on the volume of adds to multiple files, your files can be fragmented all over the disks causing far greater physical seek times. This stuff does not happen on i5 servers. Yes, the i5 has a few manual disk management facilities such as "reorgs" and reclaim storage and in theory, operations should run these periodically. But, they don't and it is OK. i5 users have discovered that this is absolutely not necessary. You can run on an IBM i5 for months with no measurable degradation in performance caused by disk fragmentation.

Single Level Store with High Level Interface

At the high-level interface, the single-level store mechanism delivers an image that is unaware it even has disk drives. Memory is viewed as one big continuum, with objects addressed by name. All objects get an address in the continuum. The microcode worries about where the objects and object pieces actually reside on disk. This saves programmers and systems managers (in larger installations) tons of time managing system resources.

As previously noted, unlike a PC system, there are no A. B, C, D, or E drives. Therefore, the C drive never gets filled up, and a D drive is never needed. All the data from many disk drives appears as if it all resides on one disk drive, though it is spread evenly across as many as several thousand disk drives on the largest systems. Objects, including databases are referenced in the continuum by name and the system worries about where all of the segments of the files are actually located. Think about all the time that saves a person by not having to decide which disk or disks something should reside upon in a large system. Moreover, when a file does not fit on one disk drive in other systems this is more work for the

implementer. On the i5, it just goes out there and the system worries about splitting it up – not the user. This is a tremendous time savings because nobody has to worry or work to make it right.

The Car Analogy

To help gain an appreciation and form a proper perspective for the hugeness of single-level store, this next example uses the analogy of a car and miles per gallon, or better yet, inches per address.

If a car could go one inch per address, then mathematically a car with a 24-bit address space would be able to go 264 miles. Say the address width is doubled to 48-bits. Without doing much work, you might conclude that you should just double the number of miles to 528. But that would be wrong. A car with a 48-bit address space could in fact go 4.5 billion miles. You don't double it once, you double the cumulative value 24 times to get the 4.5 billion value. In other words, the car could go to the Sun and back about 24 times. Can you imagine where an original AS/400 RISC system with its 64-bit hardware address would take you? How about a 96 or 128-bit address? This would cumulatively double the 64-bit address, 32 to 64 additional times. We can all agree that the result would be a very big number. Anything more would be nothing less than extra very big.

Object-Based Architecture

In 1978, IBM systems engineers spoke of the System/38 as having an object-oriented architecture, though technically the system at the time was object-based. Only in the late 1980s and the 1990s did the term object-oriented take on real meaning with the use of new programming languages such as Smalltalk, C++, and Java. These used what is known as the object-programming model. As hard as it may be to believe, even the 1978 model System/38 was an object-based system. Much of what everyone has learned about object orientation over the years is contained within the notion of an object-based system.

As an aside, there are no commercial object-based or object oriented servers available today. Only the IBM i5 fits the bill. All other servers are legacy in their design.

The number of object types in the i5/OS operating system is huge. IBM has assigned a three to six character mnemonic for each object type. When this object type is written in "English" or your native language, it is always preceded by an asterisk. To give you a brief snapshot of the vast list of object types implemented in the i5, Table 6-1 shows a list of the most commonly used objects, their mnemonics, and a short description:

Table 6-1 Object Types Found on i5

Object Type	Object Description
*LIB	Library (where objects are stored. Libraries cannot exist within other libraries)
*PGM	Program (for compiled languages: CL, RPG-IV, COBOL, C, C++, COBOL No interface restrictions)
*MODULE	Module (linkable into a program from a compiled language)
*SRVPGM:	Service program (dynamic set of one or more modules, like a DLL file in Microsoft's world).
*BNDDIR	Binding directory (holds a list of modules and service programs and is used when creating programs).
*CMD	Command (an object used for calling programs – used extensively in the operating system interface)
*MENU	Menu (List of options, accessed with the GO command)
*FILE	File (for both devices, data, and program source; described with DDS; files can also be created with SQL)
*STMF	Stream file (traditional file that would be familiar to most Unix and Microsoft users and stored only in directories, not libraries.)
*DIR	Directory (part of the Integrated File System that is equivalent to Unix and Microsoft)
*JRN *JRNRCV	JRN & *JRNRCV: Journal and journal receiver (used to journal changes to files, data areas, and stream files)
*USRPRF	USRPRF: User profile (allows users to sign-on to the system)
*JOBD	Job description (used when submitting/starting jobs)
*JOBQ	Job queue (used to queue up batch jobs to run in a subsystem).
*LIND	LIND: Line description (communications line: Ethernet, token ring, etc).
*DTAQ	DTAQ: Data queue (used to queue up data entries for fast retrieval by other jobs).

*MSGQ	MSGQ: Message queue (used to send message to users, can also be used as a data queue)
*OUTQ	Output queue (used to queue up output to a printer or diskette writer).

With an object-based system, entities are encapsulated so that only operations defined on an object may be permitted (e.g., program object code cannot be modified via a text editor, etc.), and that objects possess an atomicity in that they cannot be split or manipulated except as an entire object. This is in radical opposition to the UNIX and Windows models, in which all objects are regarded as files, and a file operation is permitted against any UNIX system object (e.g., executable code, devices, etc.) This feature alone is one big reason why the i5 all-everything machine is not virus prone. It's just not an easy target.

Everything within the i5 system, database files, programs, job queues, message queues, is an object. Each object on the system has two parts. The first part is referred to as the "descriptive part." This contains text about the object but more importantly, it defines the valid ways of using the data part of the object. The second part then is the "data part," which serves as the functional part of the object.

For example, in a program object, the descriptive part contains rules that state that the data part will be treated as executable, read-only, compiled code. As such, the only operations permitted on this object are those that you would expect to be enabled for a program. You can't add records to it. You can't read it in as input to a program. But, if you have the proper authority, you can execute it. If it were a database object, of course, its rules would permit you to write directly into the middle of the file if you chose, but you cannot write into the middle of executable code since the system just won't let it happen. Thus, the notion of a two-part object design ensures data integrity for all objects in the system. And, so, again, viruses and other malware cannot hide out in i5 objects waiting to attack your system.

With this simple example of what you can do with a database file and a program, you can see that an object-based design has very important security implications. On Windows systems, for example, one mechanism by which computer viruses enter is by masquerading as data. Since programs are just files in Windows, it is easy for a bad program to be

carried around innocently on such systems as if it had some data merit. Once the malware gets inside a Windows system, it tries to become executable code and wreak havoc on your system and even other systems. Such a change of characteristics isn't possible on the i5. If the system lets a package enter as data, it must retain the characteristics of a database file forever. It can't change its mind and become an exe as in Windows and take you for an unwanted ride to McAfee or Norton's antivirus site.

i5/OS Rewritten Using Object Oriented Tools

Though the i5 system is object based, in 1995, IBM's Rochester Lab rewrote the rules of how far object-oriented programming could be taken. In a major redesign and reprogramming effort, Rochester rewrote the under layer (microcode, low-level code below the TIMI) of the OS/400 operating system (licensed internal code) as an object-oriented project. The 95 percent of OS/400 that ran above the TIMI continued to work just as before, after some cosmetic changes. Even more importantly, all of the user code (RPG and COBOL programs) that had been compiled more than 17 years prior, continued to work.

IBM used an object-oriented methodology and object programming tools. No other commercial system had ever been written in this fashion. It was a first: new hardware and a new orientation. Somehow, though a major technical achievement, it did not make the national news. The AS/400 in-crowd knew about it. When you consider that Windows is getting a lot of press because it is working toward running on a 64-bit platform, the AS/400 family accomplished this long ago and yet still does not get a fair shake from the press. That's why perhaps even the reader was unaware of this capability. Today, the AS/400, iSeries, and i5 are the only object-based commercial systems in existence anywhere. Thus the i5 family provides even greater business value through objects because its integrity does not get compromised and the organization can work on business problems rather than fighting the virus du jour.

Capability-Based Addressing

Security is the process of controlling access, preventing access, limiting access, granting access, and revoking access. Capability-based addressing, implemented in the System/38 in 1978, is acknowledged by the experts as

the best way to achieve system security. With the AS/400 family, in the form of today's i5, it is built in. You do not have to buy it.

Research Project

Of course, you are not going to buy a computer just because it has capability-based addressing. But once you have an idea of what it is, you'll want your computer to have it. You will then see all other systems as inferior. This advanced notion is worth discussing. Way back in the 1960s and 1970s, computer scientists were planning the future of computing. One of the first advanced capability-based system designs from Carnegie Mellon was called the Hydra operating system. Interestingly enough, Hydra also was object-oriented, and was built with a primitive machine abstraction layer (high-level machine interface), along with a single-level store and a number of integrated functions.

> Note: Unlike the i5, however Hydra and all of the other advanced computer science research projects noted below are software models and never achieved the integration found in the System/38, AS/400, iSeries or i5.

The KeyKOS micro-kernel operating system emerged in the mid 1980s and was an improvement over the Hydra. In the mid 1990s, yet another improvement operating system arrived with the help of the University of Pennsylvania's Extremely Reliable Operating System (EROS) project. EROS releases sound much like the story of Linux. Now on Release 0.6.0, with prerelease 0.8.3 already shipped, the EROS project, spearheaded by Jonathan Shapiro, has taken the concept of capability-based systems yet another step toward the ideal.

None of these implementations--Hydra, KeyKOS, or EROS--are implemented on a system that you can buy however. The i5 family of machines is the only commercial embodiment of capability-based systems. The Hydra, the KeyKOS, and the EROS efforts are computer science research projects at their best. They may very well be the wave of the distant future for all other machines, but they are not out there today, and from the speed in which the projects are moving and the propensity for today's OS vendors to change, the distant future is way out there. The System/38 was introduced as a capability-based system way back in 1978

when the notion of capabilities was first being kicked around computer science circles. The AS/400, iSeries, and i5 are even better implementations for the 21st century.

Though capabilities pertain to objects, the way the concept is implemented on the AS/400-iSeries platform is unique, in that the hardware and the software are designed together. As mostly software implementations, the Hydra, KeyKOS, and EROS, (DEC, Motorola, IBM S/370 and Intel hardware), even if successful, could not achieve the unparalleled performance and scalability advantages of hardware and software integration and abstraction as done by IBM i5 processors. Again, the IBM i5 is the only commercial machine that you can buy today that offers these unique capabilities.

If you are as intrigued by the notion of capabilities as I am, read *What a Capability Is!* by Jonathan Shapiro, available on the EROS Web site at http://www.eros-os.org/essays/capintro.html.

After taking an informal survey, Shapiro concluded that none of his friends, not even the technically savvy, who worked in the computer field, understood what he did for a living. So he decided to help folks like you and I understand the notion of capabilities by starting from scratch. His article is well written, light in spirit, and assumes little knowledge. It takes the reader on a journey toward a real understanding of the concept of capability-based systems.

Because Jonathan Shapiro has already done a great job in defining the notion of capability, I have chosen not to paraphrase, but to include three paragraphs from his work. I repeat them below, for the technically inclined. If you have no concern for the technical aspects, feel free to skip these.

> "Dennis and Van Horn introduced the term capability in 1966, in a paper entitled 'Programming Semantics for Multiprogrammed Computations.' The basic idea is this: suppose we design a computer system so that in order to access an object, a program must have a special token. This token designates an object and gives the program the authority to perform a specific set of actions (such as reading or writing) on that object. Such a token is known as a capability.

"A capability is a lot like the keys on your key ring. As an example, consider your car key. It works on a specific car (it designates a particular object), and anyone holding the key can perform certain actions (locking or unlocking the car, starting the car, opening the glove compartment). You can hand your car key to me, after which I can open, lock, or start the car, but only on your car. Holding your car key won't let me test drive my neighbor's Lamborghini (which is just as well--I would undoubtedly wrap it around a tree somewhere). Note that the car key doesn't know that it's me starting the car; it's sufficient that I possess the key. In the same way, capabilities do not care who uses them.

"Car keys sometimes come in several variations. Two common ones are the valet key (starts, locks, and unlocks the car, but not the glove compartment) or the door key (locks/unlocks the car, but won't start it). In exactly this way, two capabilities can designate the same object (such as the car) but authorize different sets of actions. One program might hold a read-only capability to a file while another holds a read-write capability to the same file.

'As with keys, you can give me a capability to a box full of other capabilities…"

i5 Security Built-In

Capability-based addressing is a notion that uses the address to provide the capability that permits or denies access to an object. Again, because the i5 is a hardware/software hybrid, this advanced security notion could be explored and implemented within the address scheme of the high-level machine. The i5 uses this advanced computer science notion as its object-level security implementation. i5 object addresses are really not known above the machine interface, and thus even security is enforced below the machine interface (TIMI).

IBM was so proud of its implementation that in 1981, at the International Conference for Computer Architecture, Frank Soltis, a well known IBM scientist and the main architect of the System/38, along with Merle Houdek and Roy L. Hoffman, presented the notion of capability-based addressing as implemented in the IBM System/38 to the Association for Computing Machinery (ACM) Special Interest Group on Computer Architecture.

The System/38 therefore, in 1978, was the first commercial machine that used a capability-based model enforced by capability-based properties. On the System/38, the addressability pointers were built to be 128-bits wide, of which 96 bits are the address, and the remainder represents the authority (capability). The System/38, AS/400, iSeries and i5 hardware use an architecture known as "tagged," which makes it virtually impossible to counterfeit a system pointer.

The i5 therefore handles all security by object through its capability-based addressing. Everything on the system is an object. Everything can be secured very easily at this base level, using the capability-based architecture. Before an object can be used, a capability (authority) must be established to use the object based in the user profile and the object description itself. Security checking takes place at the time you attempt to reference any object on the system. If you are authorized, you get a "key" to it. If not, you are excluded. The beauty is that it is extremely functional and fast, since it was not built as an afterthought. It was built into the machine architecture itself. It's done within the base of the system. In other words, it differs from all other commercial implementations, since it is not an add-on provided only by software.

You may ask how much is capability based addressing worth to your organization. To answer that, you would have to know your security exposures and how much you were paying at the server level or on internal and external firewalls, including the technical expertise. You might find the cost staggering. Capability based addressing does not solve all of that but it is the only machine-based security mechanism available on any computer today, and it helps businesses protect the business value provided by their i5 servers by keeping them secure.

Integrated Data Base

The System/38, in 1978, was the first computer ever built with a relational database that was integrated within the hardware and the very framework of the system. Integration is a common theme in the AS/400-iSeries-i5 architecture. The integrated relational database was and continues to be a hallmark of the i5. There is no other commercial machine in existence, even today, which comes with a built-in relational database. Can you imagine how far ahead of the competition the System/38 was in 1978, when DB2, IBM's mainframe relational database product had yet to be

announced? And with a System/38, it was just there! You got relational database with the machine. With the i5 of course, you still do.

Moreover, since the notion of relational database was part and parcel of the architecture of the original System/38 and continues with the i5, a number of often-used relational DB facilities were built right into the hardware instructions set. Consider that one of the most frequently used operations in a relational database is index creation. The i5 family has implemented this function as one hardware instruction. That is why from way-back, the System/38 would outperform all competing systems of its size in the relational DB area. In fact, to run as well as a System/38, the competition had to execute its benchmark with sequential and indexed file processing to avoid the overhead of an add-on database management system software package. The System/38 had just one performance number as does the i5. Both machines can run database as well as non-database applications with no degradation.

IBM i5 Breaks DB Rules

Most relational databases use mathematical set theory and set oriented operations, implemented through the Structured Query language (SQL). Simple features such as the ability to link a compiler read and write operation to the database are not part of the deal. Language compilers on other machines know nothing about databases. In fact, "compiler reads and writes to a database" are anathemas to the spirit of the original relational database model.

Rather than worry about upsetting the late Tedd Codd, the inventor of relational database, the pioneers in the Rochester labs chose to create a relational database that could support set theory but, more importantly, could work naturally with the problem and procedural programming languages of the day using record at a time processing. Back then IBM did not care if it was different, if different was better than the standard. Therefore, the System/38 developers built a relational database that could not only read and write naturally to the database, but also the language compilers were database-aware.

Since the one and only System/38 relational database would always be present on every System/38, as it is on the i5, the compiler writers and the utility writers did not ignore the opportunity to enhance the productivity of the integrated database within their own software offerings. In fact,

they built their products to take advantage of the presence of the database, and to make their compilers and utilities, as well as the database, easier to use.

Oh, sure, the Tedd Codd database purists hammered the notion that SET theory was not used for all functions as not being true to the relational model. Record-level access was not part of Codd's plan. Ironically, this is a major advantage of the System/38 and i5 implementations. Other relational database implementations continue to be plagued with jury-rigged, unnatural facilities within their high-level language (HLL) compilers because their implementers chose not to aid the programming effort with their designs. For example, to read a record with a traditional system, instead of just issuing a READ command in the natural compiler language, the programmer would have to call a program and pass it parameters. Moreover, the programmer would have to fully describe the input and output in the program.

Integrated Database Makes Programmers Productive

System/38 COBOL and RPG programmers by comparison had life easy. i5 programmers continue to have life easy today. Since the System/38 compiler writers knew about the database because the same database was on every machine, they enabled natural operations in the language, such as READ and WRITE, to access the database with no need for special operations. Programmers did not have to code unnaturally to get their job done, so they got many more jobs done than on non-integrated database systems. Moreover, the data descriptors for input and output popped right into the programs at compile time without the programmer having to code them, saving an additional ton of tedious I-O coding time.

The traditional Tedd Codd databases were often very difficult implementations, requiring high-priced database administrators to manage the systems. Moreover, at the time, databases were either all or nothing. All programs had to use the database if a major file were converted. This created major implementation difficulties. The System/38 database worked first time, every time, with no database administration required. If a file were defined to the database, programs still could use their System/3 or System/34 or System/36 or System/370 internal RPG or COBOL data descriptions without having to convert the program to use the new database field descriptors. This means that cutover to an i5 heritage

machine continues to be a snap and that adding database files is still not an issue. All of this facility permits programmers to build systems faster; and it enables them to bring them online faster than ever before in computer history.

Rather than making it more difficult for programmers, by forcing them to use set theory in their program logic, IBM created the easy to learn data description specifications (DDS) language to accommodate the way programmers actually worked. This helped the programmers who used the database to be even more productive than those who chose to continue to use auto report, copy books, or hard-coded input/output program specifications. In its product-excellence slide presentations that I delivered to System/38 and AS/400 prospects over the years, IBM suggested a five-to-10-fold increase in programmer productivity would be achieved over traditional methods, using these powerful, integrated tools. This improvement still holds with the new IBM i5.

It was real. Actually, it still is. The only difference today is that IBM has stopped saying it. Why? Rex Harrison would surely call it a "puzzlement!" When IBM markets its i5, the company highlights all of the advanced system functions that are newest to the machine. These include the best Java Virtual Machine in the industry, logical / fractional partitioning, native Linux, native Unix, and mainframe class performance. Yet, all programs written for OS/400 or i5/OS in high level languages even today continue to take advantage of the productivity facilities of full database integration. In other words, programmers still write code 5 to 10 times faster than on other platforms. IBM just doesn't highlight that part of the machine anymore since the advances are almost thirty years old. IBM i5 developers have been enjoying this level of productivity since the System/38 was announced in 1978, twenty-seven years ago. One would think that by now, the competition would have caught up. They haven't.

No Name Database

In the early 1990's, IBM did a survey of its AS/400 customers. It is a fact that many i5 users even today feel they need no IT staff or a small staff to keep their systems running. IBM polled its AS/400 accounts back then to see if they knew that there was a database on the system. IBM reported that half of the AS/400 users surveyed did not even know their machine had an integrated database. Yet they were using it! That's when IBM decided to use its DB2 brand for the AS/400 integrated database.

Of course, that marketing move ruined one of my favorite pitch lines that I always felt put the AS/400 DB notion in perspective. At one time I was able to say, "If it has a name, the machine knows nothing about it. If it has a name, it is not built in; it is an add-on software package." Consider the plethora of databases that fit this mold. The list includes DB2 for all other platforms. Sybase, Informix, Oracle, and MS-SQL Server are also examples. They all have names. With these databases, no language compilers can have any built-in DB hooks. There is no READ or WRITE interface from a compiler to any other database on any other system. Now the IBM i5 database has a name, DB2/400 Universal Database, but it is still integrated, and though it is much more capable than the original System/38 database, it is still as easy to use as ever.

Future System Today

When the System/38 was developed in 1978, and deployed in 1980, it was dubbed the "future system today." An honest appraisal by the Windows-loving trade press of the underpinnings of the IBM i5, which still uses the advanced technology first deployed in System/38, would render a far more complimentary identifier than their current label, "legacy."

In recent years, IBM's POWER architecture has entered what is called the POWER5 generation. Though the boxes based on the chips appeared for the first time in May 2004, the new chips blew the socks off the competition. At the same time, the baby POWERPC chip that IBM has been developing for Sony PlayStation also came aboard as well as IBM PowerPC chips for the Microsoft X-Box. Though many, who work with i5, know the box through its operating system, i5 hardware itself has become the acknowledged best in the industry. In fact, IBM calls the i5 system of today a **mainframe for the masses.** This is a big compliment for a mainframe oriented organization. The fact is, the all-everything machine, the IBM i5, today is mainframe-class, and it is as fast as or faster than the mainframe.

In addition to IBM being tops on the large side of business computing with its new i5 offering, the company has a chance to revolutionize the small business area with its new POWERPC chips built for the PlayStation and X-Box. Some day soon IBM should be able to mass produce these inexpensive "Playstation chips" for its own use and deploy

them in small PC-sized i5-type computers, thereby reducing substantially the entry price of all-everything computing.

The Best of the Best

The i5 architecture represents everything IBM knows about computers and probably wishes it could have placed into mainframes over the years. At the risk of summarizing with too many superlatives, I am convinced that the i5 is the most technologically elegant machine within IBM, and in the entire computer marketplace.

Summary: Develop Applications Five to Ten Times Faster

Because of the six principles we have discussed as well as principle 7 which is covered in Chapter 9, as noted in this Chapter, application development on the i5 is five to ten times more productive than on any other platform. This is the innate capability that made the AS/400 of 1988 the DEC killer. Programmer productivity and easy-to-build applications brought the AS/400 and now the i5 to their renowned position in the industry. In 1988, AS/400 programmer productivity not only killed DEC as a company, but there was also some friendly fire. The IBM 9370 and the IBM 8100, both small mainframe computers, also suffered from the success of the system.

i5 Is a Special Mainframe

In a company traditionally managed by mainframe heritage executives, with all products over the years seemingly examined for their mainframe affinity and friendliness, and their abilities to generate revenue, the i5 has survived and thrived. Ironically, the i5 today is a mainframe, but it is completely unlike the mainframe that IBM builds in mainframe plants. After all, it is the all-everything machine.

IBM acknowledges that it is tough competing against the Microsoft marketing juggernaut. Yet, there is nothing else like this all-everything, "Swiss-army knife" machine. It is clearly the best computer technology available and when IBM is ready to take on Microsoft, there should be a

lot of fun and "revealment." Just like beta and VHS, however, the best technology may not win in the marketplace. That's one of the reasons why I wrote this book. I want the best technology to win. The more everybody knows about the all-everything machine, the better its prospects to one day rule the world. And a fine ruler it would be.

Chapter 9

Integrated Transaction Processing

Ala Carte Software

Ala carte system software has been a mainstay of the mainframe and most platforms for many years and IBM has made lots of money on middleware such as VSAM, CICS, MQSeries, and DB2. The same model works for Unix and Windows platforms. In the 1970's and 1980's when transaction processing and database features were invented, they were sold as products to customers with installed systems. To be sold as products, they were given intriguing names and they were used as middleware to enhance existing operating systems. In the mainframe area, for example, there have always been features that besides the operating system, needed to be purchased in order to have a more complete operating system. In the Windows and Unix arena, there continue to be the same plethora of add-on products including numerous database offerings such as Oracle, Sybase, and SQL Server.

These products are all separately orderable, separately installable, and separately maintainable as optional pieces of operating systems that are shipped incomplete. In fact, in many cases, the products come from separate vendors. That is the IBM mainframe way, the Unix Way, and the Windows way. These three platforms continue to be ideal spots for piece parts software vendors to sell their wares.

As we have been demonstrating throughout this book, the all-everything machine is integrated. Thus, essential elements are included within the hardware and operating system and are part and parcel of the overall computer system experience.

The Role of Programming Languages

In today's world, computer science languages such as C, C++, and Java seem to rule the day. However, just about every company out there that has an ERP system or any other type of business software package would quickly find that the package is written in either RPG or COBOL.

The reason for this is simple. Though a computer science type programmer feels better when he or she has full control of all aspects of the machine – even those aspects that could cause the machine to crash, the business programmer is merely interested in producing results for the company in terms of usable software. Moreover, there are just two languages, both with origins dating back to the 1950's that were designed from the ground up for business use. Though computer scientists and academics shudder at the mention of their names, the fact is that almost all back room business software on major server computers is written in RPG or COBOL.

Business Languages for Business Jobs

I have defined and written a database and simple transaction processing program that I call Advanced Hello World. All computer programmers at one time have programmed the simple Hello World program in one or more languages as an entrée to learning the language. Advanced Hello World is a rudimentary but slightly more complex program that provides an inquiry panel for a database access. The results are then brought back to the bottom of the same inquiry panel. It is a very simple program with basic function but it demonstrates both database access and interactive (transaction) processing in one simple program. The Advanced Hello World RPG version is shown in Figure 9-1 and the COBOL version of the same program is shown in Figure 9-2.

One of the first things that you would notice is that the RPG program is substantially smaller than the COBOL program (16 statements vs. 64). That's one of the reasons why COBOL has always been referred to as a verbose language. Java experts tell me that the same program, written in Java would more than double the number of statements.

Considering that programmer productivity is often measured in the number of lines of code produced in a days work, a programming

language that requires more lines of code by definition is less productive than one that requires less lines. RPG as a language and RPG as implemented with the integrated database and integrated transaction processing facilities of the i5 is the most comprehensive, and easiest to use business programming language of all time. Anybody who tells you differently has never worked with RPG. COBOL is the next productive business language. When using an i5, both of these languages benefit from the principle of **Integrated Transaction Processing.**

Bill Gates Hates RPG

As an aside, it may help to better understand why the RPG language is pooh poohed by the academicians and the computer scientists. Being a Business/IT professor myself gives me a unique perspective on this dilemma. In a word, it is *practical*. In other words it is not theoretical. It is purposeful for business, though not totally multi-purpose in nature. In other words, you would not use RPG or COBOL to draw dancing bears or create spinning globes on a display panel. I miss the point of why a business person would want a programmer doing that type of nonsense anyway. Just as many academicians want academic freedom over many aspects of reality, even those that do not apply, computer scientists in academia and outside academia want computer freedom. It's that simple. Languages written to support business productivity do not fit this model of free thought.

One of the greatest hybrid computer scientists and marketing geniuses of all time is Bill Gates. When OS/2, an IBM OS originally written by Microsoft was introduced in 1986, there was a strong rumor that IBM was about to bring out an RPG compiler for its new OS. It never arrived. My perspective is that if it had arrived, perhaps even OS/2 would be a successful operating system in small businesses today. By design, IBM kept its most productive business programming language at the time from its least expensive platform. And the least expensive operating system platform, OS/2, eventually died a slow and agonizing death at the hands of Windows NT.

Note OS/2 is a mostly defunct operating system that was intended to replace MS-DOS for IBM PCs in the mid to late 1980's. While Microsoft was building OS/2 for IBM, it was secretly preparing an

early Windows version that later destroyed OS/2 in the marketplace. IBM and Microsoft ended cooperation over this and a number of other factors and Microsoft got to use its OS/2 work (OS/2 version 3.0) as a starter set for Windows NT.

Back in the late 1980's, Bill Gates, Microsoft's Chairman and Chief Software Architect, told me over a beer that I would never see a Microsoft-built RPG compiler. He kept his word. He said he hated RPG. "It's that language with those… indicators," he told me. As a true computer scientist, he just hated the language. Hating RPG was in his blood. The C language and the C++ language and the Microsoft developed Visual Basic language have all been pushed by Gates because they were "more functional" and lower-level. Bill Gates did not have to worry about rules. Again, computer scientists like languages in which they can do everything unimpeded – even crash the machine.

None can deny that Bill Gates' Windows wares have more than their fair share of crashes. None would deny that Bill Gates is also the master marketer. Through his superior marketing, most new computer scientists coming from colleges today believe in the Gates notion of computing – via C, C++, and Visual BASIC. Most also even believe that it's OK for computers to crash as often as PCs do.

Never being a business programmer himself, Bill Gates either did not understand or did not want to understand that the two most used business languages of all time, RPG and COBOL are well used in business because they are easy to use, stable, and they are far better suited for the job. From my own conversation with Mr. Gates, I don't think that would matter.

Note: For years Microsoft used i5 family machines to run its business. More than likely, since early i5 machines did not perform well with C or C++, Bill Gates was more than likely running his business on RPG while he was insulting the language over a beer.

Transaction Processing Software

Regardless of how good RPG and COBOL were in the early 1970's for batch processing, the new wave of video terminals that found their way to business desktops in the mid 1970's demanded even more than these business languages could naturally provide. As noted in Chapter 5, IBM answered the call for terminal support early on mainframes with its Customer Information Control System (CICS). CICS is a large transaction processing monitor that companies can purchase for mainframe computers. It enables interactive transaction processing.

On small System/3s at roughly the same time, IBM developed the Communication Control Program (CCP), another transaction processing monitor, which brought a lower level of transaction processing to the System/3. The major transaction processing program for non-IBM platforms today is clearly Tuxedo, which came to life in 1983 at Bell labs and was perfected by 1989. It is now marketed by BEA Systems.

Programmers writing for CICS, CCP or for Tuxedo have many more jobs to do than merely send and receive screen panels. For example the programs must check to make sure that the screens reach the users and that the data that is returned is valid. Such error checking and correction added many lines of code to transaction processing programs. Terminals are foreign to all other system compilers so, unlike normal disk or tape support in the file section supported by business languages, there is no support for terminals. Thus non AS/400-iSeries programming languages are written to be completely unaware of terminals.

To talk to CICS or CCP or Tuxedo, a programmer must invoke a call to the TP monitor and pass it arguments directing it to perform a specific operation such as "send a panel" or "receive a panel." Suffice it to say, with these compilers, it takes lots more than simple Reads and Writes to a display a simple panel or to manage an interactive conversation with a user terminal.

The Beginning of Integrated Transaction Processing

During the development of the System/38, the notion of a workstation (WORKSTN) device was brought forth in Rochester Minnesota. Even before the System/38 was ready to go, in 1977 IBM used the in-process

work for the System/38 as the basis for System/34. The company announced and delivered a WORKSTN device capability that changed the nature of interactive computing forever on IBM small business systems.

Programmers from System/3 who had been toiling with the rigors of CCP were amazed at how simple it was to work with the compilers on this new System/34. In 1980 when IBM released the System/38, the notion of a WORKSTN device was perfected with the introduction of the *display file* object.

Just as a tape monitor is not needed or a card monitor or a printer monitor or a disk monitor in compilers, the IBM Rochester Software Engineers chose to eliminate the need for a terminal monitor in their OS and compiler design. Instead, they chose to treat a terminal as a real device that should not require a complex monitor. One might say that they integrated the TP monitor such as CICS or CCP or Tuxedo within the system itself, rendering it invisible. But, that would be an understatement. IBM built the operating system so that it could work with non standard devices. Thus, its compiler writers were able to provide natural links to the operating system support for such devices right within the compiler. It literally made programming for interactive terminals a piece of cake.

Moreover, unlike the System/34 WORKSTN device, the implementation of the WORKSTN device as a display file with the System/38 brought along support for multiple users as an innate operating system feature. In other words, when coding for interactive users with a System/34, a programmer had to know how many users at one time would be working with the same interactive program. When a program was coded for the System/34, the programmer needed to designate it as a multiple requester terminal (MRT) program or a single requester terminal program (SRT). Each SRT request caused a program to be loaded. Just the first MRT request caused the program to be loaded and subsequent requests permitted the new terminal user to be attached to the same user program. In the System/34 MRT environment, the programmer was responsible for keeping track of the data of the various users who were using the program at any point in time.

With the Display files and the further tailoring of the notion of a job on the System/38, all programs had the benefits of being MRTs without having to code for multiple users. The operating system kept one copy of

the program in memory to be used by all, and it also provided a set of working storage called a *process access group* for each user who signed on to that program. If WORKSTN files made System/34 a cake walk, the innate multi user facility of display files in each compiled program added a thick glob of whip cream icing to the cake when the WORKSTN notion was used on a System/38. There was and is no easier way to code for interactive transaction processing.

RPG Coding for Interactive Work

Take a look at the first line in Figure 9-1 to see how simple it continues to be to code the WORKSTN display file in an RPG program. Inside of the file named PANEL in line 1 is a screen panel defined as SCREEN1. In line 7 of the program, you can see an operation called EXFMT. Next to it you see the word SCREEN1. This very powerful EXFMT (execute format) operation sends the panel to the user, and puts the program to sleep. When the user presses a function key or an ENTER key, the program wakes up and processes the returned information from the display screen. Thus, this one operation is both a write and a read. No other compiler in history has ease of use facilities as this. That's why programmers using the all-everything machine have always been the most productive in the industry. They still are and that's a fact.

Figure 9-1 RPGIV Version of Advanced Hello World Program

```
1  FPANEL     CF   E             WORKSTN
2  FLANGUAGE  IF   E           K DISK
3  D ERRMSG          C                   CONST('HELLO
WORLD TRANSLAT-
4  D                                     ION NOT FOUND,
TRY A-
5  D                                     GAIN')
6  C     *IN99         DOWEQ     *OFF
7  C                   EXFMT     SCREEN1
8  C     LANGUA        IFEQ      'END'
9  C                   LEAVE
10 C                   ENDIF
11 C     LANGUA        CHAIN     LANGUAGE
90
12 C     *IN90         IFEQ      *ON
13 C                   MOVEL     ERRMSG        MESSAG
14 C                   ITER
15 C                   ENDIF
16 C                   ENDDO
```

Figure 9-2 COBOL Version of Advanced Hello World Program

```
*************** Beginning of data
********************************
.......-
A+++B++++++++++++++++++++++++++++++++++++++++++++++++++
+++
        PROCESS
        IDENTIFICATION DIVISION.
        PROGRAM-ID. HELLOAC001.
        ENVIRONMENT DIVISION.
        INPUT-OUTPUT SECTION.
        FILE-CONTROL.
            SELECT DB-LANGUAGE
               ASSIGN TO DATABASE-LANGUAGE
               ORGANIZATION IS INDEXED
               ACCESS MODE IS RANDOM
               RECORD KEY EXTERNALLY-DESCRIBED-KEY
               FILE STATUS IS MF-STATUS.
            SELECT DISPLAYPANEL
```

```
               ASSIGN TO WORKSTATION-PANEL
               ORGANIZATION IS TRANSACTION
               ACCESS MODE IS SEQUENTIAL
               FILE STATUS IS WS-STATUS.
        DATA DIVISION.
        FILE SECTION.
        FD  DB-LANGUAGE
            LABEL RECORDS ARE STANDARD.
        01  LANGUA-RECORD.
            COPY DDS-REFFMT  OF LANGUAGE.
        FD  DISPLAYPANEL
            LABEL RECORDS ARE STANDARD.
        01  PANEL-RECORD   PIC X(150).
        WORKING-STORAGE SECTION.
        01  PNL-INPUT.
            COPY DDS-SCREEN1-I OF PANEL.
        01  PNL-OUTPUT.
            COPY DDS-SCREEN1-O OF PANEL.
        01  WS-STATUS          PIC XX.
        01  MF-STATUS          PIC XX.
        01  INDON              PIC 1 VALUE B'1'.
        01  INDOFF             PIC 1 VALUE B'0'.
       PROCEDURE DIVISION.
        BEGIN.
            OPEN I-O DISPLAYPANEL.
            OPEN INPUT DB-LANGUAGE.
            PERFORM SCREEN-IO THRU EXIT-SCREEN-IO
                UNTIL IN99 OF PNL-INPUT = B'1'.
          CLOSE-ALL.
              CLOSE DB-LANGUAGE DISPLAYPANEL.
              STOP RUN.
          SCREEN-IO.
              WRITE PANEL-RECORD FROM PNL-OUTPUT
               FORMAT IS 'SCREEN1'.
              READ DISPLAYPANEL INTO PNL-INPUT
               FORMAT IS 'SCREEN1'.
              IF IN99 OF PNL-INPUT IS EQUAL TO B'1'
                  GO TO EXIT-SCREEN-IO.
              MOVE LANGUA OF PNL-INPUT TO
                  LANGUA OF LANGUA-RECORD
              READ DB-LANGUAGE
                INVALID KEY PERFORM LANGUA-NOT-FOUND
                 NOT INVALID KEY PERFORM LANGUA-FOUND.
          EXIT-SCREEN-IO.
              EXIT.
          LANGUA-FOUND.
              MOVE CORRESPONDING REFFMT  TO SCREEN1-O
OF
```

```
                PNL-OUTPUT.
           LANGUA-NOT-FOUND.
              MOVE 'HELLO WORLD TRANSLATION NOT FOUND,
TRY AGAIN'
                TO MESSAG OF PNL-OUTPUT.
****************** End of data
********************************
```

eCommerce Transaction Processing

With a simple WORKSTN file, IBM eliminated the need for a major cost component and a major customer programming effort as would have been required with CICS, CCP, or Tuxedo to support interactive terminals. Today, on all other systems to support transaction over the Web, a Web monitor program such as Bea's Weblogic, or Microsoft's .NET, Apache's Jakarta TomCat, or IBM's WebSphere is absolutely a necessity. This is a very similar notion to the requirement for CICS and CCP and Tuxedo as much as thirty years ago. There is no eCommerce transaction processing engine built into any system today, including the all-everything machine. For the all-everything machine, the solutions today for Web transaction processing are Jakarta Tomcat and WebSphere Server.

Having said that, the AS/400, iSeries, and i5 are positioned well for a major compiler enhancement. Just as IBM was the first and only company to initiate integrated dispay file transaction processing, when the company chooses to create a WEBSTN (Web station) file, the RPG and COBOL compilers that today work with terminals can simply be retrofitted to work with Web Pages without even touching the program logic. Carrying the notion further, IBM can also create a WWSTN file for a combination Workstation and Web Station file and provide the same code to be callable via a terminal or via a Web browser. Since this is the natural way for an all-everything integrated box to talk to devices through its languages, I would expect that IBM is working on this methodology as we speak.

In the meantime, of course, the all-everything machine is positioned well for the Web by being able to use the same or similar Web transaction processing monitors as all other servers out there.

The future for programming transactions on the i5 is bright indeed.

Chapter 10

Bill Gates, Steven Jobs, Otto Robinson

Pleased as Punch

Show me a business with a computer shop that runs an AS/400 or iSeries with a reasonably competent staff, and I'll show you a set of very pleased IT professionals. AS/400, iSeries, and i5 people love their servers. It is a modern-day phenomenon. In one independent survey after another, AS/400 users, display more computer bias and are downright bigots regarding their machine, compared with all others. They have very good reason.

David H. Andrews is one of the most respected consultants in midrange computer marketplace. As proprietor of the D.H. Andrews Group, he tests the attitudes of AS/400 customers periodically. Through his consultancy, based in Cheshire, Connecticut, over the years, Andrews has conducted countless surveys of IBM AS/400-iSeries customers and others in the industry. Andrews' work offers powerful insights for customers to examine and for IBM to evaluate in making future plans for its product set.

The results of some recent Andrews' surveys have long been available for analysis, and they reflect the attitudes that IBM i5 customers have today, and have had for many years. AS/400-iSeries users are arguably the truest and bluest of all IBM's customers, and are perhaps the most loyal customers in the 30-plus-year history of the midrange computer. The study concluded that the AS/400-iSeries would continue to be the primary platform for the majority of respondents for some time to come.

For those of you interested in reading D.H. Andrews' information first hand, go to his web site at www.andrewscg.com.

The Most Reliable System in the Industry

As noted previously, the most cited reason that AS/400-iSeries users continue with the platform is that it is built like a brick house. It just does not go down. It does not check out in the middle of the night for unknown reasons, forcing a business into a panic. It is stable; it is reliable; and it is there when you need it. While the average PC server experiences several weeks of down time each year, the AS/400-iSeries checks in with a measly five hours. Most AS/400-iSeries shops claim no unplanned downtime whatsoever.

No matter how reliable a machine may be, nobody buys anything just because it is reliable. My pencil doesn't go down either, but I would not pick a pencil as the main data processor to run my business. The reason why the i5 get such high marks is that they provide high-quality business solutions, which are more customizable than on any other platform. AS/400-iSeries also allow businesses to react to change more rapidly than any other platform. With other systems you can get a software package just like all of your competitors, but with an IBM i5, you can also tailor the package or write custom code to help get the competitive edge.

If you are Bill Gates, Steven Jobs, or Otto Robinson, you chose your AS/400-iSeries-i5 because it is the only machine that can give you the competitive edge necessary to win your market. With the AS/400-iSeries-i5, these three people have been able to plan for change in their industry and be leaders rather than followers in molding their computer systems to fit the ever-changing complexion of their businesses.

Bill Gates Used AS/400s to Run His Business

Business managers and executives typically are unconsciously unaware that their production data processing systems and decision support systems are using AS/400, iSeries or i5 technology. Perhaps the most unconscious IBM AS/400 customer of all is Bill Gates, the "barbarian leader" from Microsoft. For many years Microsoft executives slept restfully at night, knowing, according to many observers, that their business was safe because it was running on 23 silent AS/400s in a back

room someplace, way out of sight. Though the evidence is no longer as obvious, the rumor mill suggests that Gates and company still process on AS/400-iSeries, but they do not take D.H. Andrews satisfaction surveys.

In fact, as recently as the year 2000, Alex Woodie wrote in Midrange Technology Showcase that Microsoft had been processing on 23 AS/400s around the world until 1999. Then, rather than address the Y2K issue head on, the company decided that it would switch to 1200 Windows NT Servers. Yes, I said 1200. Unfortunately for Microsoft, according to Woodie, they found that they could not make these work for the company and one year later were plugging AS/400s back in to get their business straightened out.

Woodie attributed his knowledge of the Microsoft saga to Dr. Frank Soltis of IBM who ought to know what Microsoft is or isn't doing. From Internet threads I have been able to find that Microsoft went ballistic with this announcement and apparently threatened legal action to get a correction and /or a retraction. Woodie writes to Ted in this Thursday, Nov 16, 2000 email discussion thread:

> Ted,
> Microsoft vehemently denies that they have any AS/400s anymore (or use any applications that run on AS/400s), and is pressing hard to get a correction or a retraction in SHOWCASE. Please contact me if you have any useful information regarding the validity of the rumors, or if you know them to be false.
> Alex Woodie
> Products Editor
> Midrange Technology SHOWCASE

The rumor mill from here suggests that what actually happened is a combination of all of the above. Microsoft stopped running its own AS/400 shop indeed. But, when NT could not do the job, it farmed out the work to a service bureau who use, guess what? The i5 family. Again, it's just a rumor but, if it is true, it is no wonder why Microsoft does not want that story out.

The point of this reference is not to cast aspersions on Bill Gates or Microsoft. However, there is nobody familiar with the IT industry that

would not agree that Bill Gates has more reasons than anybody to run his business on the Windows NT/XP type platform. It speaks volumes to the character of the AS/400-iSeries product line to have been chosen by Microsoft as its business system of the 90's. It says even more about the i5 heritage machines if Microsoft could not really replace all of their work on the i5 heritage platform over the years with as many as 1200 Windows servers. We don't know this for sure but the folklore sure makes the i5 look like a winner even for those who try to defeat it at every turn. So, It says a lot if the rumor has it right that Microsoft returned from its adventure and to keep its business running effectively had to go back to AS/400 servers, though in an offsite, outsourced location.

Steven Jobs Uses i5s to Run His Business

Steven Jobs and Apple, many years ago, decided to switch from the five DEC VAX units on which they were running their highly profitable microcomputer business, to the IBM System/38 platform. The System/38 is, of course, the direct predecessor to the AS/400, the grandfather of the IBM i5. Many industry analysts who are familiar with both the former DEC (swallowed by Compaq, which was swallowed by HP) and IBM give credit to IBM's AS/400 box for actually taking DEC out of the midrange computing business. The AS/400 killed the DEC VAX and made the company easy prey for the PC leader of the day, Compaq, to acquire. Now, as noted, even Compaq has disappeared from the computing scene. It is interesting to note that Apple moved to the System/38 long before DEC was dead. Apple has yet to die.

When I look back at Apple's decision to move to the System/38 product set, it is obvious that there had to be a compelling reason that was not obvious. At the time, Apple's major product was the Mac. As a terminal to DEC machines, the Mac worked quite well. It had a natural serial interface and terminal emulation software. DEC users could just plug a Mac into the Network, and with the proper software it would just work. The same was true for Mac users. Apple was able to place DEC servers on their Ethernet networks or serial networks, and they would connect with few technical issues.

The System/38, never in its lifetime supported serial (ASCII) terminal devices, and it never supported Ethernet or AppleTalk or any other local area network protocol. In other words, the Macs could not connect to

the System/38. Being a renegade company, Apple saw something in the System/38 that it did not see in any other computer in the industry. Apple knew it would be able to react to business changes more quickly with a System/38 than any previous computer system, including the DEC boxes. It was so important for Apple to use the System/38 that the company created Rube Goldberg special devices and then jury-rigged the company with the devices to enable their Macs to talk to the System/38.

When the AS/400 came out, it had what Apple needed without the jury rigging. It eventually supported serial (ASCII) and Ethernet, as well as AppleTalk, so that the Mac became a natural device to the AS/400. But Apple had selected the System/38 when industry observers would have concluded that there was no way for the Mac to participate. Thus, there is no doubt that Apple Computer loves its System/38s, and now its AS/400, iSeries, and i5 systems. Today, there is actually more reason for the i5 and the Mac to be friends. They are, in fact, relatives. The underlying technology in the new POWERMacs is a similar to the POWER5 technology that IBM uses in its AS/400, iSeries and i5 lines.

The early Apple says a lot for the desirability of the System/38 and AS/400 systems as IBM products and as tools that provide major business value. Back in the mid 1980s, Apple saw that there was a definite competitive advantage in using the box as its business system, and the company made sure that it did what was necessary to allow that to happen.

Otto Robinson Takes IBM i5 to the Bank

From the outset in he early 1980's, at Penn Security Bank in Scranton, Pennsylvania, bank president Otto P. Robinson Jr. was told outright by IBM that the System/38 was not to be used as a modern banking computer. IBM clearly told Robinson on numerous occasions that the System/38 was not a banking machine and it would never be a banking machine. IBM suggested that the bank president look at other IBM systems for banking, such as its mainframe line.

It was clear to me that IBM had invested its software and support dollars for banking in its mainframe line and the company believed that Otto Robinson would best be served with a small mainframe rather than trying to go it alone on a system not designed to support banking. IBM did not

want his business if he wanted a System/38. Mr. Robinson, a very bright individual who, besides being bank president, is also a lawyer and a mathematician, was perplexed that IBM would purposely deny the banking industry the use of what he believed to be its finest computer system of the day.

Robinson was relentless in his dealings with IBM, and he never gave up. Despite IBM's desire not to sell him one, Robinson ordered a System/38 for the bank. Because IBM had created an adapter for the magnetic ink character recognition (MICR) reader that the bank needed to process checks, his programming team converted his System/3-based batch banking software to the System/38 platform. Meanwhile, Otto Robinson was actively lobbying IBM for banking devices (teller terminals and ATMs) to be natively supported on the System/38. I had the pleasure of being the assigned account systems engineer to Penn Security Bank, so I got to see all of this action first hand.

Robinson just would not take no for an answer. Eventually, his notoriety in doing things with the System/38 that nobody else was able to do brought him invitations to speak at COMMON and other computer and banking trade shows. As you read the rest of this story, you will see that after telling Mr. Robinson that he was on his own, IBM did more than handstands to help him make all of his ideas work. Ironically, the same IBM that had told him that he should not use a System/38 invited the outspoken bank president to speak at various IBM-sponsored banking seminars across the country to demonstrate his effective use of the System/38.

Robinson did not sit still in his own shop, either. He discovered his own hardware solution for the teller terminal incompatibility. Just as Apple could not naturally connect its Macs, Otto could not connect IBM's leading-edge teller terminals. The System/38 supported just one terminal type. It was known as the IBM 5250. It was a boxy green-screen terminal that at the time was the small system version of IBM's 3270 terminal.

Besides the terminals not connecting naturally via a local adapter, they did not connect even using communications adapters. IBM had stopped including its old time communication protocols on the System/38. These protocols had very technical sounding names, such as the BISYNC telecommunications protocol or the ASYNC ASCII protocol. IBM supported its green-screen 5250s through the then new Systems Network

Architecture/Synchronous Data Link Control (SNA/SDLC) protocol. Working through all that technical mumbo jumbo, it meant that the System/38 box could not even attach the older mainframe style model 3270 terminals and it could not attach IBM's newest 3600-style BISYNC banking terminals. Clearly, the System/38 had been left out of the banking picture, since this was traditional IBM mainframe territory.

Enter the Wild Ducks

Within IBM over the years, I had the pleasure of meeting and working with a number of "wild ducks." Sometimes to Thomas Watson Jr.'s pleasure, these ducks were left alone to achieve greatness in IBM. One such duck was a talented engineer named Ed Brucklis from IBM's Boca Raton, Florida, plant. When I met Ed, he had just written a program for IBM's Series/1 minicomputer that could be used to enable the attachment of unsupported terminals, such as 3270 BISYNC terminals, to the IBM System/38. In essence, Brucklis did for IBM what Apple's engineers did for Apple. Through his program, 3270 BISYNC terminals were able to talk to the IBM System/38.

Since Brucklis's Series/1 front-end creation was developed in the same Boca Raton facility that offered limited banking support to IBM's midrange customers, he was persuaded to carry his creation one step further. He added the translation for IBM 3600 Teller Terminals and ATMs. It did not take long for Otto Robinson to get word that an ATM hardware solution (through Brucklis in the Rube Goldberg vein) for the System/38 was now available. (Okay, so I told him!) Brucklis himself eventually helped make it work for the bank president.

After he realized the boxes could connect and talk, Robinson discovered an old ATM software package that had been built for the System/3 line of computers in the early 1970s. This program, written by IBM's Bill Pinkerton and others, permitted IBM's ATMs to be controlled by very old System/3 programs. Robinson worked with his local IBM systems engineer, yours truly, to research whether this package could be made to run on the System/38. I offered my endorsement and recommended how to proceed. Robinson ordered the package and some IBM ATMs, and I worked with the programming team to make sure the ATMs would light up and deliver the cash.

Before going live, Robinson once again beseeched IBM. This time, he argued for an encryption routine for the AS/400. IBM again reminded Robinson that the System/38 was not a banking machine. In frustration, Robinson ordered the BASIC language for the System/38 and wrote his own data encryption standard (DES) routine, using the BASIC programming language. I don't think he had ever written a program before in his life.

ATMs were so important to small banks around the world that Robinson opened his doors to any and all to see the marvels of the System/38 controlling a network of ATMs. From as far away as Indonesia, System/38 banking people came and were impressed, and many moved forward with their own System/38 implementations.

As nationwide ATM networks began to spring up everywhere, Cash Stream, Cirrus, and Mac were the big players. Robinson contracted with Cash Stream, and his programming team then had to modify the Pinkerton ATM package even further to accept ATM cards from non-Penn Security customers. This was also a success.

System/38 Home Banking? Why Not?

In the early 1980s, banks were experimenting with some innovative notions like bill paying systems and home banking. An astute banker, Robinson saw the need to enter this marketplace. At the time, not even the big players had a presence. Robinson went to IBM again and asked about ASCII terminal support for what he termed video text. IBM again reminded Robinson that the System/38 was not a banking machine and that it supported only the 5250-style terminal data stream, and there were no plans to change this.

Robinson called over his local IBM marketing team again to discuss his dilemma. He did not want to know what the System/38 could not do. He was already using ATMs on the System/38, and IBM had said that he could not do that with the System/38. I had been working with Series/1s at the time, since IBM was pressuring its branch offices to sell these systems. IBM gave me the job of seeing what we could do with this most unpopular box in our local branch office.

I introduced Otto Robinson to the idea of using another Series/1 running the Yale ASCII terminal package. This package could support any type of

ASCII terminal in existence, including the RCA Videotext Terminal, of which Robinson was particularly fond. The problem was that the Yale ASCII Series/1 wanted its host to speak the BISYNC 3270 data stream. It would then convert it to ASYNC ASCII, the necessary protocol of the dial-in device. Unfortunately again, IBM's System/38 spoke only SDLC and the 5250-style data stream.

Once again, Ed Brucklis came to the rescue. As noted above, the original intent of the IBM 3600 teller terminal translation software originally written by Mr. Brucklis was to permit 3270 BISYNC terminals to attach to the System/38. This was just what the Yale ASCII package wanted. So again Mr. Robinson was pushing the IBM envelope trying to use technology that was not yet available for the System/38.

The Rube Goldberg Home Banking Solution

Long before Internet computing, in his model home-banking scenario, Robinson envisioned a bank customer with an RCA Videotext terminal dialing the Yale ASCII Series/1 at the bank. He saw the Yale ASCII Series/1 converting the ASYNC ASCII data to BISYNC 3270 for the original Brucklis Series/1. The Brucklis Series/1 would then convert the BISYNC 3270 data signals into SDLC 5250 signals and send the twice-converted data stream to the System/38. The System/38 would think it was talking to a directly attached native 5250 terminal. In reality, the connection was from a dialed-in terminal device three systems away. (Phew! If you had a hard time following that, there is no need to worry. You are not alone.) Eventually it worked, but not right away.

Not knowing if this would work, IBM agreed for Ken LeFevre, a Series/1 specialist from Philadelphia to make a house call with yours truly on Otto Robinson. Though he thought it was a very novel idea that may have unforeseen issues, LeFevre could not offer any reason for this approach not to work, and gave it his stamp of approval. Robinson then bought his second Series/1, and in short order, in the test environment, the System/38 was talking to dial-in RCA devices using the two Series/1s in between. But there was a problem.

Hang Up! Please!

Since the AS/400 had no notion of dial-in terminals, there was no way to tell the System/38 that the dial-in banking customer had disconnected.

This created a big problem. If another banking customer called into the same phone line, after a prior customer had hung up, he would be connected to the same session the prior user thought he had exited. Obviously, in the banking industry especially, this compromised security. Clever as it was, it would not do the whole job.

Robinson went back to IBM, which, of course, again reminded him that the System/38 did not support banking or ASCII terminals. Otto Robinson reminded IBM that it had taken the money for the second Series/1 and the Yale ASCII package. Every now and then, the lawyer in Robinson would show his face. IBM agreed to have Ed Brucklis himself visit the bank, but did not imply that this technique would be supported or that it would ever work.

When Brucklis arrived from Boca Raton, it was snowing in Scranton, and he did not have an overcoat. Soon after Brucklis' arrival, we went to lunch about a block away from the bank. Mr. Brucklis got a taste of Scranton, Pennsylvania winters that he would not soon forget. It was food for some gentle jabs when we sat down at Shookey's Restaurant. At lunch, there was some peppy conversation between the bank president and the software engineer. The two hit it off and formed a bond that was quite understandable. Both men would never accept the decks they were dealt, and when faced with what others would call insurmountable obstacles, they were able to devise methods to surmount them.

Robinson muses sometimes about the wild duck characteristics he saw in Ed Brucklis. They were a good team. When Brucklis saw the home banking workshop, he was obviously tickled that his work was being used so cleverly. The RCA Videotext terminals were set up using TV sets as monitors.

The Home Banking Skunk-Works Demo

Robinson demonstrated the home banking skunk-works setup and showed the problem with the dial disconnect. He asked Brucklis how the product could possibly be usable with such a major flaw. I can still remember when Brucklis stood, undaunted, and gently fired back at Robinson: "When this product was written, nobody ever thought it would ever have to talk to a Philco TV." Both men roared with laughter, and Brucklis vowed to make it work. He did. Over time, he became one of Robinson's favorite IBMers.

When the AS/400 came out, Penn Security Bank was in line for one of the first. The bank made the transition painlessly from the System/38. When IBM announced RISC-based AS/400 models in 1995, again Penn Security was one of the first IBM customers lined up to make the transition. And, again, it was mostly painless.

Otto P. Robinson Jr. is still the bank president and still uses the AS/400-iSeries to give him the competitive edge he needs in the banking industry. Thanks to Otto Robinson and his unrelenting input to the IBM planning processes, unlike the System/38, the AS/400 and the IBM i5 are able to handle the unique requirements of banking, as well as home banking.

Who's the Fool?

Bill Gates, Steven Jobs, and Otto Robinson are not fools. What did they see in the AS/400 predecessor (System/38) that would encourage them to go through one hoop after another to be able to deploy the i5 heritage platform in their businesses? What makes the AS/400-iSeries so special that Microsoft, with a now less than amicable relationship with IBM, and an operating system (Windows Server) that directly competes against i5s, persisted in its use of the platform?

They did not know or care that the AS/400 or System/38 had 48-bit or 64-bit hardware. They did not know that the system uses 128-bit software addressing. In some cases, they did not even care that it did not have the hardware support to allow for essential devices to be attached. It was not hardware. It was not the IBM Company. Then, what was it?

What they saw initially in the System/38 was a machine that could help them run their businesses with minimal issues and disruptions. More importantly, in many ways they saw a system that would give them an edge over their competitors so that they could adapt their business systems to the changing times at speeds unattainable on any other system. Otto Robinson saw it as a survival issue. Steven Jobs saw it as a business issue. I've got to believe that Bill Gates, like Otto Robinson, saw it as a survival issue. He needed a system to make his rapidly growing business survive. Quietly, the AS/400-iSeries, using OS/400 (now i5/OS), did the job for all three.

i5 Plusses

The time from conception to implementation has always been far less with the IBM i5 family and the System/38 product line before it. Some developers will say 5 to 1; others as much as 10 to 1. This ratio is the relative speed that application development and program maintenance and updates can be performed on the i5 compared with all other platforms.

For businesses wanting the competitive edge, there is no time to wait for the important functions and features to be rolled into the industry-standard packages. Therefore, you must build them yourself. The i5 heritage platform plays well in this arena. Ask Bill Gates! Ask Steven Jobs! Ask Otto Robinson!

Chapter 11

The Rise of the RISC Machine

The PowerPC Is Coming

In 1994, as IBM prepared to refresh the CISC based AS/400 product line with bigger and more powerful processors, Dr. Frank Soltis, iSeries Chief Scientist, and others freely discussed the coming 64-bit PowerPC architecture RISC processors that IBM was cooking up in its labs. IBM had pre-announced the coming of RISC processors to its existing customers like nothing else I had ever witnessed. The company was usually very tight-lipped on future products. So intent was IBM on bringing RISC processing to the table in short order that it announced a new batch of AS/400s in new "RISC" cabinets about a year before RISC emerged.

The new black systems that the company introduced in May 1994 were dubbed **RISC ready**. The cabinets used for the RISC-ready boxes were substantially different from the white racks that had been used in prior CISC systems. The days of rack-based AS/400s had passed and would not be back until the i5 was introduced in 2004. When the RISC boxes did arrive, the cabinets were so similar to the RISC-ready boxes that it was obvious they were intended for a 1994 announcement of RISC boxes. But the RISC processors were not ready for prime time in 1994, so IBM did the next best thing. Even though they were not RISC-based, the new black models again energized IBM's AS/400 sales.

> Note: RISC processing stands for reduced instruction set computing. The late John Cocke, a very bright IBM engineer who worked for the company until 1992, invented the notion of RISC. John Cocke passed away in 2002.

Cocke's concept of RISC resulted from his detailed study of the trade-offs between a number of advanced notions available at the time. He demonstrated that a small (reduced) number of instruction circuits on computer chips could be appropriately defined to exploit the instruction set and thus realize very high performance with relatively few circuits. So if made correctly, each computer chip could be less expensive, and along with some additional sophistication in software compiler design, the resulting machine would perform substantially better than the complex circuitry of the day. Cocke's notion was contrary to the established direction of the functionally more complex instruction sets and machines. Once RISC was established, it was not long before the more complex notion of instruction sets was dubbed complex instruction set computing, or simply CISC.

Advanced 36 – First RISC Box

In 1994, it was just over six years that IBM had introduced the AS/400 system as the replacement for the System/36. However, in those six years, the company had little luck in attracting many of its System/36 customers to its AS/400 line.

On October 4, 1994, IBM took bold steps to bring its System/36 customers to the AS/400. In an act of marketing brilliance, the IBM Company announced that it was using its first set of AS/400 PowerPC RISC chips to introduce a brand new System/36, built from its yet-to-be-announced AS/400 RISC hardware.

The System/36 instruction set was very limited, so IBM was able to "etch" the entire set on the new PowerPC RISC chips, even before the technology was ready for the more expansive AS/400 instruction set. The new box that was built on the RISC chip was introduced as the AS/400 Advanced 36, and it was an immediate success. Its constituency had waited six long years for its arrival. The pent up demand was fulfilled and IBM received its share of new orders for this new RISC based AS/400 with a System/36 personality. Thus, IBM rescued its System/36 customers and gave them exactly the system for which they had been asking the prior six years.

RISC Is Ready

RISC-ready did not last long before RISC was ready. On June 21, 1995, IBM finally introduced its RISC line of AS/400 processors, based on POWER technology. In addition to being RISC processor driven, the new machines offered the industry-first implementation of 64-bit processor hardware. It would take another six years for Intel to produce a 64-bit processor and another four years after that for Windows to be able to begin to use its power.

The new AS/400 boxes were being slotted into two different environments. The "Advanced Systems" RISC-ready boxes were replaced by the "Advanced Series" machines that were fully RISC processor enabled. The name Advanced Server continued with the new RISC server models.

One more historical change occurred with the introduction of the RISC-ready models and continued with the RISC boxes. IBM had introduced what it called server models of the AS/400. These were substantially more powerful and less expensive than the typical AS/400 system models. They were good for client/server computing, Web computing, and batch computing, but they were not as good for typical AS/400 interactive workloads. IBM announced these units to compete more vigorously against Windows servers, which had no interactive AS/400-type requirements.

With the change from CISC to RISC, IBM did not change the name of the AS/400. Despite the fact that the hardware had completely changed, there was no real name change. To an extent, the name did change, however. The "AS" no longer had the same meaning. In 1988, the box was known as the **Application Server/400**. In 1995, the AS/400 got two new names and became the **Advanced Series** and the **Advanced Server**. The subtlety was missed by many. IBM again subtly changed the name of the AS/400 on August 19, 1997. At this time, the company was interested in adding that little "e" that Lou Gerstner, IBM's chairman at the time, had fastened next to the word "business." Gerstner had coined the term e-business, and so all IBM servers were on a clear track to becoming eServers.

Lou Gerstner's notion of e-business spilled over to the AS/400 product line immediately as the faithful servants at Rochester painted the little red

"e" next to the word AS/400 on all new shipments from the plant. The "new" AS/400e models were made available August 29, 1997, less than two weeks after they were announced. See Figure 11-1.

Figure 11-1 AS/400e Models Circa 1999

AS/400 Keeps Growing in POWER

Sitting at the top of this new line was a model called the 650. It was a 12-way processor, a first for the AS/400, and it delivered phenomenal overall performance for systems of the day. Its relative power rating was 2340 for the 12-way (12 computers in one) in terms of the Commercial Processing Workload (CPW) benchmark measurements. CPW numbers are all relative. There is no magic to the CPW benchmark. It is simply that the higher the number, the faster the machine.

Note: A processor is the computer part of the computer. On larger systems, it is known as the central processing unit, or CPU. For example, the Pentium IV, or the Celeron, is the PC's Intel processor. The computer industry uses the term n-way to describe how many processors exist on a particular system or server model. Thus, if n=12, a 12-way system would have 12 CPU chips, each being able to process data and perform computations. There has been a law of diminishing returns regarding n-way systems on most vendors' servers. In other

words, if a server delivered 120 CPW with one processor, two processors would not deliver 240. There is always overhead associated with processor switching and keeping all processors busy on a server. So a two-way might yield 220 CPW, a three-way might yield 300 CPW, and so on. As IBM and other vendors have been making n-way systems more efficient, more and more processors can be added without negatively affecting performance.

To put the CPW number in perspective, let's compare the 1997 AS/400 with the 1978 System/38. When the System/38 was announced in 1978, the fastest model at that time would clock in at less than 2.25 CPW. In less than 20 years, as you can see, the processing power had grown over 1,000 times.

In 1998, IBM again added to the AS/400 hardware line by jacking up the power of its top-of-the-line Model 650. This was the fastest AS/400 processor at the time, coming in at 4550 CPW from the prior year's maximum of 2340 CPW.

On February 9, 1999, IBM made more AS/400 announcements. The company introduced a new RISC-based computer line call the 700 series. At the top of the model 7XX line stood the Model 740. Like the model 650, it was a 12-way machine. Also like the Model 650, the 740's top rating for a 12-processor system was 4550 CPW. The 7XX line was basically a new packaging scheme, and it introduced a new notion called interactive and batch CPW. The Model 7XX machines could act as interactive systems (standard AS/400 terminal programs) and as client/server systems. Thus, by combining the batch and interactive capabilities of the systems in one box, IBM was able to eliminate the need for two different model types: Advanced Series and Advanced Servers. The 7XX machines were known only as servers.

On May 22, 2000, IBM was at it again. This time, the company introduced its 8XX series of processors. The 7XX series had lasted just over a year. The 8XX line also included 12-way processors, just like the Model 740 series. However, with the new S-Star POWER processor, the company juiced the individual processors so that the 12-way systems were capable of firing out an amazing 10,000 CPW of computer processing power. At the same time, IBM introduced its first 24-way (24 processor) AS/400 model, known as the 840. Its CPW rating for a 24-processor unit

was 16,500. This is better than 6,000 times more powerful than the original IBM System/38. The 8XX line also included low-end single-processor units that proved very attractive for small and midsized businesses.

Also, on May 22 and June 12, 2000, to help the smaller customers, IBM announced a smaller sized AS/400. It was called the Model 270 AS/400 line. The boxes were very powerful for client/server computing, but to keep the cost lower, IBM limited the amount of the machine's power that could be used for traditional terminal-oriented computing.

At the same time, the company announced the smallest box in the line, the AS/400 Model 250. This tiny, almost portable unit prices out at less than $10,000 for a very basic machine. This unit was a further constrained machine with a limited growth path. Its intended market was Intel server customers and IBM AS/400 developers who could not afford a large AS/400.

IBM's Total Rebranding

The year was not over. In fact, just five months had passed since the introduction of the Model 8XX, when, in October 2000, IBM held a major all-IBM announcement meeting. Every server, from mainframe to AS/400, was affected by the announcement. The company re-branded all of its computers as eServer models. The AS/400 received the name **eServer iSeries 400.** Many AS/400-iSeries observers note this as a turning point in IBM's overall attention to the AS/400-iSeries product line.

The year 2001 was not so special in terms of AS/400 hardware or software announcements, especially since new models of the AS/400 had now become the iSeries. All other AS/400 systems were not renamed, and all of these remaining non-iSeries boxes, some relatively new and some very old, continue to use the OS/400 operating system and continue to be called AS/400s.

On May 14, 2001, IBM had its one iSeries announcement for the year. The company enhanced the speed of the RISC processors again. With this announcement, IBM took the wraps off its latest POWER processor, known as the POWER4, and made it available on iSeries boxes. These

chips had been used successfully for about a year in its pSeries processors, which were formerly the RS/6000 product line. POWER4 processors have more sophisticated technology and achieve higher speeds than predecessor RISC processors. To highlight the whopping power of the new processor chips, IBM introduced its iSeries Model 890, 24-way processor. This behemoth with all 24 processors running delivers 29,300 CPW of power. With this announcement, IBM just about doubled the performance of the iSeries models.

Concurrent with the juiced up 24-way processors powered by IBM's POWER4 technology, IBM stretched the processor limit of the iSeries one more time. The new 32-way Model 890 was off the charts. It delivered a whopping 37,400 CPW of power with its 32 processors. Again, that's well over 12,000 times the power of the original System/38.

IBM also introduced better and faster disk technology. With the introduction of the Model 890, the company offered over 72 terabytes of disk along with these powerful processors. For those of you who are counting, that's about one quarter of the 281-trillion-byte addressability of the 48-bit processor in the original System/38. Even at this new level of capabilities, the old System/38 hardware could address every piece of real estate on the disk drive and still have room to spare.

On January 24, 2003, IBM gave the iSeries still another facelift. New models were announced, called the 800, 810, 825, and 870. Because IBM believed it had finally solved a problem with interactive performance, the announcement has historical significance.

The end is not in sight with power boosts on the iSeries hardware. In 2003, Dr. Frank Soltis pre-announced the 2004 server lineup. He said,

> "Our 2004 Armada box-based, POWER5 chip-powered systems will scale up well to 64-processors. So not only is there a major boost in the n-way capability but in combination with the POWER5, the new box achieves well over 50,000 CPW."

In May, 2004, IBM upped the ante again, by increasing the power of each RISC chip with the introduction of POWER5 technology. Additionally, the company upped the number of processors on one machine to 64 from

32. With all that juice and its new name, "eServer i5", the AS/400-iSeries family runs applications with a staggering 165,000 CPW. And the new i5s come in both floor standing and rack mounted versions. And, it ain't over yet for the all-everything machine. See a picture of the new IBM i5 in Figure 11-2

Figure 11-2 eServer i5 Models Introduced May 4, 2004

64-Bit RISC, No Buts!

While Intel and Microsoft have been and continue to be praised for a partial implementation of 64-bit computing, with the POWER5 and the i5, IBM is already on its ninth generation of 64-bit RISC processors. For over ten years and counting, since 1995, AS/400-iSeries-i5 boxes have been enjoying the benefits of 64-bit hardware and software computing, and the press, for its own reasons, has chosen not to make a big deal about it.

Press accolades or no accolades, the all-everything machine is here, and it uses 64-bits, and unlike Wintel, there are no buts!

Chapter 12

IBM's Future System (FS) Project

From the Best Computer Minds of All Time

Back in the early 1970s, in the deep recesses of IBM, a number of exciting things were going on. The first that comes to mind is that the government was investigating IBM. There was concern that the U.S. government, in order to help foster competition in the computer industry, was about to break up IBM into a bunch of tiny little IBMs. Company executives knew that none of these little IBMs would be able to wield as much marketing power as one Big Blue, and so there was deep concern that this would be bad for the company. Unlike Microsoft's arrogant posture in the late 1990s and into the 21st century regarding its Justice Dept. case, IBM took this threat very seriously and devoted significant resources to defending its interests.

Another thing that was happening in the 1970's was that IBM's mainframe division, which, at the time, was the real champion and hero of IBM, began a top-secret project dubbed internally as *FS*, for the *Future System* project.

Part of the motivation over FS can be attributed to IBM's concern about the mounting software inventories that were accumulating in its mainframe customers' shops. Programmers were writing more and more programs every day. About every five years, IBM was changing hardware and operating systems, and this was forcing customers to rewrite programs just to stay current with IBM's new offerings. The more money the customer spent in making the transition to IBM's latest and greatest, the less money they would have to pay IBM for the latest and greatest.

Conversion Costs Too Much to Afford New Computers?

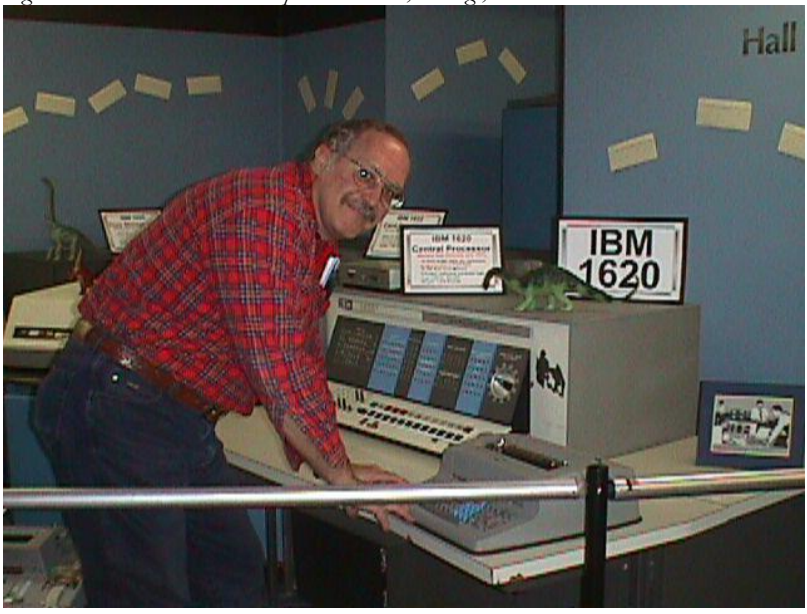
IBM's thoughts focused on whether its customers' huge investment in software would be able to continually be migrated to future IBM mainframe systems, ones that IBM had yet to develop. Without customers being able to move their software investment to these new systems, IBM feared, it would be inordinately difficult for them to migrate to new computers. This would substantially reduce IBM's opportunity to sell new systems to existing customers.

In the mid 1960s, IBM had tested this scenario when it bet the whole company on the success of its System/360 family of computers. These were introduced in 1965. IBM almost went bankrupt winning that bet and so company executives in the early 1970s remembered all too well that to gain the benefits of the System/360 computer family; its customers were forced to rewrite their programs in new languages. But in the mid 1960s, the program inventories were not as significant as they would become over time.

Before 1965, the IBM systems of the day were always named with numbers. For example, the commercial processing machines of the late 1950s and early 1960s were the 1401 (see Figure 12-1) and its follow-on, the 1410. The scientific machines included the 1620 (see Figure 12-2) and the 1710. These all used very primitive programming languages, with names such as Symbolic Programming System (SPS) for the scientific machines and Autocoder for the commercial boxes. To help its customers move to the faster System/360 computing system, IBM built an emulation facility so that this old code could run on the new boxes. Unfortunately, the emulation gobbled up enough resources to translate the old code during the emulation process that the new machines, when running the old programs, were not substantially faster than the old machines had been.

Figure 12-1 Huge IBM 1401 Business Mainframe Computer – CPU

Figure 12-2: IBM 1620 at Computer Museum, Billings, MT – G. Mohanco at Console



The 1965 Rewrite

IBM's overriding recommendation for System/360 (see Figure 12-3), therefore, was for its customers to rewrite their applications to take advantage of the new machine. This was a very expensive undertaking. To help minimize future changes, IBM recommended writing programs in higher-level languages, such as the newly introduced COBOL language. Theoretically, these high-level language programs would then be able to be ported to subsequent machines without the same difficulty as prevalent machine-oriented languages, such as the Symbolic Programming System (SPS) and Autocoder. However, COBOL suffered from some of the same disadvantages of the emulation software. COBOL programs ran slower than the lower-level languages, such as Autocoder, which had preceded it.

Figure 12-3 IBM System/360 Circa 1965 (Unknown Model)

When IBM introduced the System/370 (see Figure 12-4) in 1970, the company touted the fact that programs did not have to be rewritten to move to the new iron. It was reasonably easy for a mainframe System/360 shop that was out of gas to choose to move to the System/370. The System/370 was not a radical departure in computing, and was in fact very similar to the System/360 line. Nonetheless, IBM was very concerned about what subsequent systems would look like and whether they would handle current programs while allowing customers to use all of the new bells and whistles.

There were a number of technology breakthroughs that were imminent, and IBM wanted its customers to be able to benefit from these without spending tons of money on program conversions. The company was planning for the next computer science revolution to be delivered as an IBM solution. High tech facilities such as database, data communications, and interactive computing also were just around the corner. The future system would have to handle both the current software inventory, as well as these new capabilities.

Figure 12-4 IBM System/370 Model 125 Circa 1972

Design the Best Computer Possible

Hoping to plan the future, rather than have it plopped upon them, IBM gave its FS project team a mission to design the finest computer architecture possible, given all of the advances that were known and on the horizon, as well as those further off. IBM had a long list of features that were going to become available, such as bubble memory, and any new system would have to be able to seamlessly accomplish adding such advanced hardware to the mix.

This elite assemblage, though mostly a mainframe project team, included a few representatives from other IBM divisions. One of the represented divisions, the IBM lab at Rochester, Minnesota, made small, reasonably inexpensive business systems (System/3). The machines from Rochester

at the time of the formation of the FS group were so small that nobody really thought that any of the FS designs would be usable in a Rochester product.

The group had at their disposal the finest computer scientists from both inside and outside of IBM. Moreover, they had access to all of the customer and internal IBM requests for additional functions and enhancements to all of the existing products. They had the full customer wish list. They knew where technology was heading. They understood the imminent breakthroughs and the concepts worth pursuing. They knew the time frames. They were the most capable and the best equipped assemblage of computer designers ever formed in IBM. They were the cream of the crop, and their output was expected to be the future blueprint for advanced computing for all IBM mainframe products.

IBM invested substantial time and money in this advanced project, and was more than hoping for a big payoff. The company expected it. When the committee finished its work in the mid-1970s, it had designed the finest computer of all time. Integration of hardware and software was the cornerstone of the project. It was so complete that it was to take IBM's computing plan out at least another 20 or 30 years. It represented all that IBM knew about computing. A system built to this architecture would be splendid indeed.

Seeking Approval to Build the Best System Ever

Since the project had such high-level attention in IBM, at a certain point in the cycle, the FS committee had to present its findings and recommendations to the IBM Corporate Management Committee. If it did not get past this committee, it would no longer be funded and, as a matter of course, no systems would be developed using these specifications. The meeting with IBM's executives was crucial, as it would shape the color of IBM computing for decades to come.

As the presentation unfolded, IBM's executives were surely impressed by the excellent work that had gone into the project, as well as the ultimate capabilities of any product line that might be built using this design. But there was a dark cloud looming for the FS team.

Because this architecture was so special, it was also substantially different from any machine that had ever been built. To move to this new architecture, the presenters acknowledged that many customer programs and procedure job streams would have to be completely redone. This of course would require a substantial amount of customer reprogramming and would add a substantial additional cost for any IBM customers moving to this new architecture. This cost was viewed by IBM's management as an impediment to the possibility of selling a system based on this architecture. Not only would the customer have to afford the new IBM system, but also the customer would have to invest an even greater amount to get the programming inventory converted to operate within the new architecture.

The IBM Corporate Management Committee viewed the customer work required as a yeoman task. By adding this effort as a prerequisite to moving to the new technology, IBM executives were concerned that many customers would not be able to afford the whole tab and perhaps would therefore be unwilling to make the change? Then what?

Many of the IBM executives had lived through the System/360 experience, in which IBM had bet the company, and could have lost it all while forcing its customers to scrap all their programs. They had sworn to IBM's larger customers never to do this again. After the FS presentation, they were no less steadfast in their resolve to maintain an evolutionary, not a revolutionary course.

The FS Answer Is History

IBM's executives were not prepared to annoy their mainframe customer set, and they were certainly not prepared to bet the company again, no matter how significant the notion was at hand. And so this spectacular FS effort of multiple IBM divisions would never get to be IBM's mainframe architecture of the future. IBM gave the team a clear "no!" and broke up the group and sent them on their way. Countless millions of dollars were spent in this failed effort to change the face of IBM technology forever and for better.

Chapter 13

The Pacific Project: The Beginning of the All-Everything Machine

Moving On

Besides the Future System project and the government's big antitrust case against IBM, there were lots of other exciting things going on within IBM in the 1970s. The company had hoped to use FS to help position its mainframe product line for the future, but from reading Chapter 12 you already have learned the outcome of that effort. IBM executive management forced the mainframe division to drop the project.

With or without FS, IBM was not about to sit idly by and let the U.S. government dismantle the company without a fight. Big Blue knew that it had to take action to position itself for the future, independent of efforts such as FS, to help shape its product lines. One of IBM's most significant choices of action for this was a reorganization of the company. Though a good part of the General Systems Division (GSD) reorganization had already occurred in 1969, it was hard not to notice the coup de gras when it occurred in 1975.

Building a Company to Be Broken

At this time, IBM boldly reorganized to be able to function in the future under the looming threat of the Justice Department's dissolution efforts. The biggest action that IBM commenced was its completion of GSD as a fully functional company within IBM. This was a big deal for many reasons. Unlike the addition of plants and products and people and the

creation of new divisions or special purpose companies within the company, the formation of GSD was much different, and its real purpose was unlike any other, ever, within IBM.

The General Systems Division of IBM was established in just about every way as an entire, self-contained company within IBM. By 1975, GSD, as it was called, was missing nothing that it needed to exist outside of the IBM womb. IBM gave the division all the pieces necessary to operate independently. For example, the new division had its own development lab and a manufacturing plant in Rochester, Minnesota. It had its own research and development budget. It had its own service group. It had its own marketing department and its own advertising budget. It even had its own independent sales force, which operated from the same branch offices as the Data Processing Division (mainframe) sales team. In 1975, along with a number of other small-systems engineers and marketing representatives," I was a Systems Engineer assigned to the General Systems Division.

Unlike the Data Processing Division (DPD) of IBM, the former one and only direct computer sales force in IBM where I had previously worked, the products that GSD sold were mostly built and marketed by GSD. DPD was just a sales organization. It sold the products produced by IBM's other plants, such as the Systems Products Division (SPD). Of course, there were a few products that were made outside of GSD in other parts of IBM that were also in the GSD salesperson's kit. These included common devices for systems, such as tape drives, disk drives, and CRT/terminals and printers.

Why would a company create a new division that could operate autonomously within its own borders? The fact is that IBM had not forgotten the antitrust case; it was absolutely preoccupied by it. Thus, many of IBM's actions during this period were done with thinking that demonstrated that the U.S. government was to be feared more than the competition. As a preemptive action to a required bust up, with GSD, IBM pre-positioned the wholly contained company as a ready spin-off in case the government was successful in its antitrust efforts. Rather than risking the company being busted into parts that together might not equal a whole company, IBM was resolute in its contingency plan to bring two strong IBM's to bear in the computer industry. Each IBM would have the plant, the people, and the wherewithal to do well.

The Need for a Fighting Product

The big computer product for GSD back in 1975 was the System/3 Model 15D. It was the top-of-the-line small business system. GSD executives, headquartered in Atlanta, did not need much coaching to understand that the System/3, originally announced in 1969, was on its last legs and needed to be replaced with a machine that could win in the marketplace, even if the competition was the IBM company itself, with its smaller mainframe product line.

By 1975, the System/3, though still very easy to use, was tired and slow, compared with the competition, and it offered no technology advances of any consequence to the marketplace. The system was well out of its heyday, and it was not long until it would be blown away by competing minicomputers and small mainframes. GSD had to act quickly. It needed a follow-on replacement product for the System/3.

As the GSD executives looked at IBM, the mother company, they saw many advances in technology being made ready for their appearance within the mainframe line of computers. They understood that, to compete against DEC and Hewlett-Packard, Wang, Data General, and others; they needed these capabilities for their follow-on product line. They were concerned not only about the real competition but also about whether the division, if released to operate on its own, would need its own advanced technology to compete directly against the IBM mainframe line.

From my employment and understanding of IBM history, I have never known anybody who analyzed the internals of the company and did not conclude that in the 1970's, the mainframe executives ruled the company with an iron hand. The company belief was that if it was not good for mainframe, it was not good for IBM. From my own observations and the IBM bottom line, it was absolutely true. The mainframe was the reason for IBM's success. This fact was not lost on the little development lab in Rochester or the division president in Atlanta. They knew that to gain approval for funding, any system design they came up with had to be substantially less powerful in terms of processing capability than any mainframe in the product line. If not, mainframe dominated IBM would be motivated to never let it see the light of day.

The Pacific Constraints

No company in IBM's position would want to create a cheap little computer that could run like a mainframe. It would be confusing to customers and it would be self-defeating. IBM saw no reason to give its mainframe customers an alternative to being loyal mainframe customers. Some would say that IBM was not interested in "eating its children," though I find that term obnoxious. So, as noted in previous chapters, IBM was forced to provide real constraints to the Rochester lab to assure that the system they produced addressed the market for which they were funded.

One of the ways that IBM kept Rochester in line was by giving them dollar ceilings. For example, in the System/3's early days, as I recall, the dollar ceiling was about $3,000 per month. Rochester could not build a system that would rent for more than $3,000 per month.

At the time, IBM was mostly in the computer rental business. It was a great business. Year after year, once a system was placed, IBM would reap the rental income. The only expense absorbed by the rental was maintenance, so it was a very lucrative business for IBM.

There was no lid placed on the architecture or the components, just the overall price to the customer. A generous IBM profit was included in the rental, so you can bet that the $3,000 was not all technology. Rochester was free to develop as sophisticated a system as possible for the money. It kicked off the *Pacific Project* to do just that.

The Pacific Plan Unfolds

As Rochester put its design team together to develop the System/3 follow-on, it included Dr. Frank Soltis and a small contingent that had represented Rochester in the defunct Future System project. As it turned out, these guys had taken lots of notes about FS, and were bright and creative. So it was no surprise that the Rochester Lab, the development arm for GSD, took a good chunk of FS project output, dusted it off, and used it as a starting point for their new computer design.

IBM's mainframe customers had been writing code for its systems since the 1950s and supposedly had massive numbers of programs. But GSD executives could argue that GSD's customers had only been writing code for their systems since the early 1970s, when the System/3s first became available. Therefore, it could again be argued in the mid-1970s that the System/3 customer's average code inventory was not substantial. If it were necessary to abandon this inventory for the new FS-based architecture that the lab was preparing to introduce, most customers would accept it.

Moreover, the GSD plan would include migration programs to help move customer-written code to the new system from the System/3 as painlessly as possible. Thus, the General Systems Division would ultimately build its system based on the high technology results of the FS project, as well as the innovative hardware designs brought by Dr. Frank Soltis and his numerous Rochester cohorts.

Giving the Small System a Big Heart and Big Paws

If you were a GSD executive, and you were aware of the inroads being made by the government in the antitrust case, in the beginning of the lawsuit, you would have had to make sure that your new system architecture was scalable. One of the very first competitors that GSD would face if IBM were split would be the old IBM minus GSD. The next system choice, therefore, for GSD was very critical, since it would be the system that it would use to compete against IBM. However, with IBM keeping the performance and capacity to a minimum, the first processor used with the system could not appear to be threatening to the mainframe business.

Of course, there was no resolution to the antitrust case, and IBM remained whole. The government eventually dropped the case in 1982. So the biggest problems, which IBM GSD actually experienced in 1978, when it announced the System/38 as a big part of the embodiment of FS, was getting all of the whiz-bang function to work in an underpowered system and getting the mainframe guys off their backs.

Of course, GSD also had to make sure that any new system was an acceptable percentage faster than the System/3 computers that its customers were using; otherwise it would not sell. The System/3 Model 15D was at the top of the list. The System/38 would have to provide superior price/performance compared with the Model 15D.

When you look at all the work that IBM Rochester did in the Pacific Project to bring the System/38 to life, it's a wonder that it ever got announced and out the door. Rochester had no real experience in such a large-scale endeavor. However, a look at the history and one might conclude that their inexperience contributed to the box being built.

The Rochester scientists and engineers did not know they could not build a system such as the System/38, so they just went ahead and did it! They built 48-bit hardware and a 128-bit software machine interface with the major bells and whistles brought from the FS design and from Dr. Frank Soltis' magnificent mind. The six underlying principles from FS, as described in Chapter 8 as well as integrated transaction processing as described in Chapter 9 were at the core of the new design. To top off the challenges, the Rochester engineers and developers had to make it all work on a small scale. IBM had decreed that it had to be built on what today we would call a "resource-deprived" hardware box.

What Did Businesses Want?

In the mid-1970's, just like today, businesses were crying out for more functions in their applications, more applications, and more access to informational byproducts of applications. But there was not the plethora of software packages that are available today. So when businesses asked for such changes, their IT shop was the only place that business applications could be changed.

In the 1970s, instead of getting results when they went for services, knowledge workers were met with increasing organizational constraints. Often there was no budget for additional IT work, and the department had no way of paying for the work itself. Yet the computer-emboldened report and information users of the organization demanded more and more timely and accurate information. Unfortunately, in many cases, the existing IT staff did not have enough time or resources to satisfy many of

their requests. In some ways it seemed hopeless for a business trapped within its own budgetary constraints. There was not much anybody could do but hope that one day IT would get to the problem at hand. These were tough days, and no computer system existed that was capable of changing the situation.

What Type of Computer Functions Solve Business Problems?

Let's say you were a computer vendor, such as IBM Rochester, and you understood the frustrations of the day and the challenges to make IT programming more productive. There is no doubt that if you could build a system that could address these frustrations and help bring management information from the disk drives to the workstation or the printer, you would have a system that would be a winner.

It would have to be a new system, designed specifically for business needs. If you could, you would address this programming and operations dilemma through new and integrated technology. Any computer company wanting to solve this real problem would have to create a totally new solution that would have to meet a number of key objectives.

A new system would have to be:

(1) Easy to implement new applications
(2) Easy to modify or maintain existing applications
(3) Easy to access stored information
(4) Easy to secure data
(5) Easy to grow the system in a non-disruptive fashion
(6) Easy to afford with better performance

Though the first five points in the list would provide the major benefits that should enable a business to find enough savings to justify the system, because of the ever decreasing costs of hardware, for the system to be salable, you would have to provide all of its capabilities at an even lower price, and you would have to supply better performance than any existing system. That is clearly a tall order.

The six points above address the people-productivity side of computing. It is a simple generic laundry list of function and facility for a new computer system to help provide the computing solutions as needed. The problem for us of course, is that a system had never been built that addressed the people side of computing. But if it could be built, maybe, just maybe, it would help reduce the prohibitive and spiraling cost of program maintenance and development, and help businesses get some new work from their "MIS" staffs.

Maybe businesses would be able to migrate to the new system with major productivity enhancements and without major issues. Maybe they could grow over time in a non-disruptive fashion to faster models with even more productivity features. So, it would be important to provide more capacity with virtually no issues at all, including price. Thus, the most obvious benefit that you would have to provide would be a system like none before, at an improved price performance level (6) so that your customers would believe they could afford your new offering.

Well, neither you nor I are computer manufacturers, but even to an expert, this is a tall order. Let's just say that you had scoped out the market requirements properly and believed you could deliver a machine with these attributes. You would certainly expect such a machine would be immensely successful. And you would be correct!

Building a New Machine to New and Unusual Specifications

These were IBM's six major objectives for the Pacific project. The Pacific system was to be the embodiment of integration and productivity in one new box. In IBM's System/38 Product Excellence marketing slides at the time, these six objectives were the cornerstone of IBM's canned presentation. When prospects for the system were introduced to all of the capabilities built into the new System/38 as well as the rationale behind the system, it often became the motivation for them to place an order.

Of course, in order to accomplish these heavy objectives, if you were a hardware and software computer system developer, you would not be able to use many things from the past. The things from the past already had a

track record. They had already not succeeded in achieving the above objectives. You would need new concepts, and would have to devise a completely new computer architecture. You could not rely on the old hardware models of the past, even if they could be made to run faster. To achieve all this function, a **new architecture** would have to provide **integrated systems functions,** which took advantage of the new machine's inherent people productivity capabilities.

Large-System Function, Small-System Ease of Use

Because your marketplace would not be General Motors or the New York Stock Exchange, but rather small and midsized companies, high schools, colleges, hospitals, etc., where there are small staffs to deal with technical issues, you would have to hide all of the complexities of the internal machine. In essence, you would have to achieve the impossible by providing **large-system performance and function** with **small-system ease of use**. If you were able to do this, your system would be unique indeed, and would meet the six goals as noted.

Before 1978, IBM designed and built the Pacific System as the fulfillment of these six requirements. Announced in 1978 as the IBM System/38, the new machine was created with a new architecture built around the notion of scalability and ease of use through integration. Not coincidentally, even today IBM will tell you that the "i" in iSeries and i5 stands for integration.

Figure 13-1 Small IBM System/38 Front View

Figure 13-2 IBM System/38 With Users

The large-system functions of the System/38 were thus integrated into the firmware and the operating system to provide a small-system, simple interface to an extremely powerful box. In essence, the System/38 was

announced as the first small computer in the industry that was built with an automatic transmission. That's the story in a nutshell.

IBM used all of its knowledge of computing to fashion the System/38 as a business machine, and then 10 years later the company upped the ante with the introduction of the AS/400. Using all of the resources known to it at the time, IBM delivered on all six items with its historic introduction of the System/38, and the business benefits have been flowing to over a quarter million businesses ever since.

System/38 Is Still Outstanding–27 Years Later!

The special qualities of the System/38 have yet to be surpassed by new technology. Its architecture was born again in a new frame with the AS/400, and again with the iSeries, and again with the i5. Overall, the 1978 model System/38 is still unique and good enough architecturally that no system, inside or outside of IBM, has yet caught up with it. If you add all that has happened to the AS/400, iSeries, and i5 on top of what the System/38 originally brought to its customers, the picture of a system you get is quite formidable and very impressive. Pages and pages of new function each year, with more coming from the IBM Rochester Labs, make it very difficult to succinctly describe this machine. But one thing is for sure: Other than the AS/400, the iSeries, and the i5, there is nothing as good in the marketplace or on the drawing boards as the original Pacific machine, the System/38.

So if you were to design the system with the six points as described above, it would clearly be the System/38. That's exactly what the IBM Rochester Lab did when it introduced the system in 1978. The good news is that neither you nor I have to design it, because it has been available for over 27 years.

The reincarnation of the System/38, in the AS/400, iSeries, and i5, also has it all. The benefits include a product cost range of between six thousand and several million dollars, lowest cost of ownership in the industry, thousands of available applications, unprecedented ease of use at the operations and programming level, beat-all development productivity tools, top-flight Internet capabilities, all integrated with machine-based security features and packaged for a no-sweat installation.

One could say with certainty that the i5 provides the best environment of all for application developers. With the System/38 and the AS/400-iSeries, for years software developers have declared that **tough programming jobs become easy.** In essence, by making the System/38 into a functional computer system, with its advanced architecture, IBM had indeed invented the all-everything machine.

For Techies Only

The rest of the topics in this chapter may be a bit too much for the casual reader. I include them at this point for those who are picking up a notion of the many special attributes of the System/38 and AS/400-iSeries-i5 boxes. Rather than beg the argument, presenting these additional technical strengths of the IBM i5 family at this point reinforces my case that the i5 has become the all-everything machine, one of a kind, and the finest computer ever built.

Layered Computing

To truly understand the integrated nature of the i5, all one need do is take a look at the software layers built on a "traditional" hardware architecture. These are found on every other computer, from IBM's mainframe and p5 line to Windows, Intel, HP, and Unix boxes. Without trying to be overly technical, a number of the architecture layers are as follows:

Top Layer

Languages
Utilities
Spooling

—

Middle Layer

Sort
Graphics
Communications
Workstation Support

Database
Data management
Operating System
Machine Interface

—

Lowest Layer (Machine)

Instruction Set
Microprogramming
Hardware

Traditional Architectures

It is worthy to note that most of the function layers in traditional-architecture machines are completely separate from each other. On most machines they are provided by additional (add on) products and are not integrated well into the machine itself. You buy them as separate piece parts. In fact, one of the most significant disadvantages of a system with a traditional architecture is that, because the system is not integrated, you cannot count on all of the layers of function being present on every machine for the same hardware type.

Moreover, not everybody buys all the pieces. Since the most essential software on these machines is purchased ala carte from multiple vendors, by definition, all functions do not always exist. Moreover, when all functions do exist, there is nothing to ensure that the layers of all machines are the same. With multiple choices in the database arena, for example, other machine layers depending on a database being in the mix cannot depend on any particular database being present. The interfaces to the particular database product would be different based on Sybase, Oracle, SQL Server etc., and therefore, real integration is highly unlikely if not impossible.

In other words, when developers build software in such an environment, within each layer, there may be many different-named products from which to choose. Therefore in order to use that software as developed, a business would have to buy the same database software, such as Oracle,

for which the package was developed. This can present many issues including trying to deal with the existence of another database package already on the server.

Try baking bread when you are not sure all the same ingredients will be there every time. The bread can never be the same. Since the layers in the System/38 and AS/400-iSeries architecture are provided with every machine, there are no haves or have-nots with this system. Developers have the same affinity for a well-stocked System/38 as do bakers for a well-stocked kitchen. It's in there. Other servers start with an empty kitchen.

From the bottom up in the layers, it is safe to say that all processing on all machines is done by **hardware**. All of the other layers help the hardware know what to do. Each lower layer makes it easier to operate at a higher layer. Depending on the machine, various instructions or groups of instructions are set in **microcode**, or as IBM now calls it, the **licensed internal code**.

The **instruction set** sits above microcode**.** This allows a machine to present to software a notion that it actually has instructions in hardware that are not necessarily implemented in hardware. They may actually be materialized in the microcode layer through software. One high-level machine instruction may very well cause 1,000 hardware instructions to be executed as directed by the microcode. When an instruction in the instruction set is executed, it uses the microcode (**microprogramming)** layer to translate the instruction into a sequence, which the hardware understands.

Above the instruction layer is the machine interface which provides the personality of the hardware to the **operating system,** which provides the overall personality of the machine. On top of the operating system, most often delivered as part of the operating system, is the **data management** layer. On PCs this would be the file system that you see when you look at directories and folders. On mainframes and other ala-carte-style machines in which data management has a name, you may have to purchase a named product for this layer. A software piece part providing this function on mainframes is named VSAM. It falls into this category.

The next layer, which is optional on other systems, is the database. PC Servers and Unix boxes and mainframes have a database only if you buy

one. Oracle, Ingress, DB2, SQL Server, etc., are names for modern day server databases. You choose the database as a separate component; therefore it cannot be integrated. The layers above the database are optional on most other systems.

A key point in this traditional architecture is that it is ala carte. This means that higher-level functions in the machine cannot depend on lower level functions being there. The vendor cannot force you to buy any or all of the ala carte piece parts. If the database layer were present on a machine, which did not really care which database you used, system commands could not be built to reference any particular database in a standard way. The point is that, on such systems, the software developer (programmers) would have to work harder to make up for this. Whenever anybody in your business or my business has to work harder to get the same job done, things take longer and you pay more to get them done. This is obviously not desirable.

Integrated Architecture

The IBM development team who wrote the language compilers on the top of the architecture knew that with a System/38 or an i5, the database is a standard part, and therefore they could put hooks in the language to make it easier for a programmer to perform database functions. Otherwise, the developer would be forced to access each database in non-standard ways that would be different based on the product used. For example, and as fully explained in Chapter 9, RPG, COBOL, and other AS/400-iSeries-i5 languages are database aware, but no language has to worry about your using Oracle or Ingress or SQL Server as a database, since none of these products is integrated into the Pacific Machine (System/38)--or any other machine on the market, for that matter.

When an i5 developer issues a READ to the database in a high-level language, it is issued as a READ, not as a complex request for special functions. No other computer system from Microsoft or Sun or Oracle can claim the same. Moreover, just as database, the same notion applies to all other layers. With everybody else's system, nothing is integrated, because it can't be. It's all bought separately. Consider that a PC box, when it is shipped, sometimes does not know whether it will be running Linux or Windows as its operating system. Since Oracle on Linux is

different from Oracle on Windows, by definition no software integration can be built into the PC/Windows/Linux environment.

The message is that major components of the Pacific system have been part of its basic architecture for over 27 years. They are not add-ons and are not available ala carte. It's just in there. IBM built Pacific to be integrated, and 27 years later the integration is inside the i5. Knowing all of the pieces makes it much easier for Pacific programmers to develop new function without having to work hard and without the system having to travel through unknown layers to get the job done.

Integrated Architecture Summary

On traditional systems, you have to build the system from component parts before you start using it. You have to install tons of software before you can even use the system for anything productive. You are responsible for picking and installing the pieces you want, such as database and workstation support. You have to install those pieces and make sure that they work in your environment. The resulting puzzle often has missing pieces in all other environments and is never complete, by design. With the all-everything machine, the IBM i5, whose ancestry dates back to the Pacific Project, there is ***no assembly required,*** and there are never any missing pieces.

Chapter 14

The Fort Knox Project

Changing Structure

As noted in Chapter 13, the System/38 was the result of an internal IBM project with the code name Pacific. IBM announced it in October 1978 as the system for the 1980s. Of course, IBM expected the product to be ready long before 1980, but at least it was able to get the new product out the door. The first live System/38 that I saw in a customer account was shipped in November 1980.

In addition to the nagging delay in first shipment, the Pacific Project was also plagued by mainframe IBM's careful strategy of ensuring that the box was not permitted to expand quickly or to grow big (See Chapter 13, "The Pacific Project"). When first made available, for example, its total disk capacity was 384 megabytes. This was delivered via six 64-meg internal disk drives. The System/38 had less capacity initially than the System/3 Model D, its predecessor. Many businesses that had migrated from System/3 to early System/38 technology began to outgrow these limitations shortly after installation. To assure its customers success with the new System/38 IBM very quickly announced more disk storage capacity in the form of its mainframe heritage 3370 disk drives (Figure 14-1) to satisfy the needs of its larger GSD customers. Each 3370 unit held over 500 MB of storage formatted for the System/38environment.

Note: 384 MB is lots less than the disk storage on the smallest PC today.

Figure 14-1 IBM 3370 Disk Drives Drives for System/38

Fort Knox Secret Objectives

The System/38 and most of its successors were not well loved products within the mainframe divisions of IBM. Quite frankly, the mainframe division with its New York culture, and its Cambridge architecture underpinnings, felt that it could handle both small and large computing for IBM. It did not think that it needed the help of a little lab in the Midwest.

The mainframe contingent in IBM perhaps had forgotten that it was not their design that was heralded by small businesses across the world. It was not a small mainframe. A series of small business systems designed for ease of use attracted more and more IBM customers from their manual systems or from the competition. From System/3 to System/32 to System/34 to System/36 to System/38, IBM's small to mid sized business customers in the 1980's were able to be successful with computers because IBM's Rochester Lab made it easy for them. They were able to grow their businesses because they had been attracted to the IBM Rochester style of computing and it worked.

There were a number of documented attempts within IBM, as well as secret plans developed by the mainframe teams in IBM that had the System/38 and even the Rochester Lab itself being totally eliminated. In this scenario, of course, the work that had been done in Rochester would be moved to Endicott, New York, IBM's small mainframe plant. Rumors of the death of the System/38 persisted within the IBM of the early 1980s. The mainframe captains in IBM did not understand small system ease of use. In today's terms, "they didn't get it!" They did not really understand the System/38, but they quite understandably viewed it as a threat to the small mainframe products that they built, and if you asked them, they would probably admit that they saw no real value for the System/38 as a product for the IBM Company.

They wanted the System/38 eliminated as soon as possible after they made sure that they could ship a new consolidated small mainframe box that would replace it. Again, it bears repeating, the mainframe contingent did not fully understand that hardware alone was not what attracted System/38-loyal subjects. Without trying to learn about the System/38, they concluded that the small mainframe was its equivalent or at least good enough.

As noted previously, the mainframe division's attempts to scuttle the usability of Rochester products is legendary. Those of us in the field offices heard the strong rumors and thus were well aware that the mainframe influence would rather have us working on Endicott or Poughkeepsie mainframe boxes, than the wares from Rochester. If it were not for IBM customers insisting that the IBM's Rochester systems continue to expand to meet their needs, one can be sure the mainframe forces would have had no problem in completely eliminating Rochester machines, and perhaps the Rochester Lab itself.

The fact is that programmers in System/38 customer shops were attracted to the System/38 because of its overall personality and the opportunity to gain major productivity improvements both in IT and in the business as a whole. Such productivity could not be expected in a small mainframe environment, but the mainframe contingent, accustomed to working with large customers with large staffs, did not fully comprehend this reality and this customer sentiment.

As growing businesses quickly outgrew the System/38, they became concerned about the limited growth of the system that ran their

businesses. Some became frustrated in not having a logical growth path. So, rather than embracing the recommended mainframe route for a growing company, a number of companies, which needed more System/38 computing power than was available, chose not to look at another IBM system. They voted against the mainframe by purchasing competitive gear. They jumped to selected systems from DEC, WANG, HP and Data General. All of these companies were in their prime in the 1980s, just waiting for IBM System/38 defections. Because of IBM's ambivalence in the mid 1980's the competition did well.

Consolidation Need Was Valid

If we can eliminate from the formula the biases held by many on both sides of the question inside IBM in the early 1980's, there were few who would deny that IBM had too many disparate computer systems designed for the same sized customer set. Few executives in IBM were pleased with the company's overall small business server lineup in the early 1980s. The PC and IBM's RISC system had not even become factors as servers.

At the time, IBM was selling five different boxes that addressed the same set of customers and needs. One system could do the job for sure. But which one? That was the problem! Enter corporate and customer politics. To solve the problem once and for all with a hardware / software consolidation solution, IBM formally instituted a major project in the early 1980's called Fort Knox. Its mission was to address this issue head on, and to solve it by developing a consolidated system that could handle all of the various customer needs. Was it just wishful thinking?

The Fort Knox Pre-Mix

The Rochester Lab at the time provided two of the five systems that were to be consolidated. The Systrem/38 was the most capable architecturally, and the System/34 was the darling of the small-business set.

System/38 and System/36 from Rochester

The System/36 had already been planned as a follow-on to the 1977 System/34; in fact, IBM introduced it in 1983 while the Fort Knox

project was well underway. Though the System/36 and System/38 used the same terminals and printers, as well as a similar RPG programming language, that's where the similarities ended. The System/36 clearly was the easiest computer of all to install and make operational. Customers loved its total ease-of-use characteristics. It was also easy to learn. In many ways, the System/36 was an extension of the old System/3; whereas the System/38 was a completely new architecture and quite dissimilar internally from a System/36.

There were many more capabilities built into the System/38 than the System/36. If you start with the notion of an integrated relational database at a time when mainframe systems were not yet using relational database, you get an impression of how advanced the System/38 was for its day. In describing the ease of use of both systems, I like to suggest that the System/36 was the easiest to learn and the easiest to use right out of the box. The System/38 and i5 heritage machines are lots more trouble to learn, but the System/38 and i5-type boxes are the easiest systems to use, even easier than System/36, once you have learned them.

If it were not for the System/36, the System/38 would have been the easiest system to use of all time, but with the existence of the System/36, I had to come up with another saying to describe why I preferred the machine over the System/36. My catch-line for this notion is "hard to forget." I found myself having to use the manuals with the System/36 though nothing was extremely difficult. The Help text and the prompter on the System/38, and the logical structuring of the command set was enough to know what commands wee available and how to sue them. And, once learned, the commands were just plain hard to forget. Thus development on System/38 was much easier than on System/36.

The IBM 43XX Small Mainframes from Endicott

IBM's mainframe Systems Products Division provided the 43XX series of mini-mainframes that were all built in Endicott, New York. Some of the specific model numbers used over the years on these machines includes the following:

1. 4321
2. 4331
3. 4341
4. 4361

5. 4381

The Series/1 from Boca Raton

At the same time, a group that seemed to belong to nobody in the corporation, the non-PC side of the IBM Boca Raton, Florida, plant made a product called the Series/1. There was a dotted-line connection between Boca Raton and Rochester from the GSD days, in that the Series/1 was seen as a GSD box.

The Series/1 was a bona fide minicomputer in the fashion of DEC's VAX and HP's 3000 series. Unlike the HP and DEC boxes, however, the Series/1 did not have a loyal following in the manufacturing industry. The technical gurus of the minicomputer era looked more favorably on the upstarts, such as DEC, and the Series/1 paid for their disaffection with IBM products.

The Mainframe Distributed Mini: IBM 8100

The fifth small computer in the early 1980s lineup was a little-known box called the IBM 8100. Though this machine technically was a bona fide computer system, it was a special-purpose unit that mainframe IBM had built as an intelligent distributed terminal so that its large customers with remote plants would not have to use an IBM System/36 or System/38 to connect to headquarters.

In the early 1980s, the 8100 was mainframe IBM's principal distributed processing machine. However, more mainframe customers liked the System/36 and System/38 as distributed processors than the mainframe folks wanted to admit. The 8100 was never really much of anything, in terms of being a viable, fully functional computer system, but it had its own processor and hardware frame. It provided some local stand-alone processing capability, and mainframe implementers got to choose from two incompatible operating systems. They were known simply as *DPPX* and *DPCX*.

The 8100 was not such a big success. It was not innovative and did not fill a product-line gap. It should never have been built, and even after its availability, it should have been scrapped. It seems however, that it was easier for the mainframe folks to recommend an 8100 from Endicott than

a S/3X from Rochester though both would do the job of distributed processor quite well.

8100 Represented Dead Technology

Even when the 8100 was most alive, it was moribund, since IBM host-based networks were on their way out at the time, with the coming acceptance of TCP/IP, the protocol of the Internet.

More Fort Knox Background

You may not have ever heard of these Fort Knox machines, but they are all historically significant. The folklore suggests that IBM in Rochester, Minnesota, had bluffed and stretched the truth to originally get into the small computer business. The 96-column, card-based System/3 was supposed to be a machine whose sole purpose was to replace aging unit record equipment. Through some back-room activities, Rochester produced a computer system instead of an electromechanical Tab machine, for which it was commissioned. Because the System/3 had become a success, Rochester quickly became IBM's small business computer manufacturer, as well as the operating system developer for small business systems.

As one would expect, this totally annoyed the Systems Product Division personnel who made IBM's large and small mainframes for the mainframe marketing division (DPD). SPD believed that it had already addressed the small business computer marketplace with the SPD-built System/360 Model 20. This was IBM's least expensive mainframe-based computer designed for small businesses. It was hard for the System/360 Model 20 to compete with a System/3 that was less than half the price and provided the same function.

The Justice Department Had a Role

During the late 1970s and early 1980s, IBM was still very concerned about the Justice Department's case against it. As noted in Chapter 13, Frank Cary, IBM's chairman, saw the distinct possibility of splitting IBM along

the small and big computer lines. Rochester was earmarked as the plant for the small computer division. To coordinate small system operations, as noted previously, IBM created a full division called GSD so that when the Justice Department wanted to settle the case, IBM would offer the split on its own terms.

GSD's product to compete against mainframe was the System/38. Rochester had announced and delivered the System/38 to be able to sustain the new GSD Company. In 1982, when the Justice Department withdrew the suit against IBM, it became obvious that there were definitely too many products in the IBM small systems stable.

Was it an Impossible Task?

To complicate any thoughts of merging the product lines was the fact that they were all substantially different from one another. With the Series/1, the System/36, and the System/38, the GSD management team owned three of these five smaller computers, and none of them was compatible.

The 8100 and the 4300 series machines were built by mainframe IBM. Always wary about what the little guys in Rochester were up to, and to tell the truth, constantly plagued by a bad case of "not invented here" syndrome, the mainframe division kept pushing its product set lower, while trying desperately to keep a big spending cap on Rochester's systems. Its mainframe optimization rationale was that by pushing the product set lower, it could take more business from Rochester, and by keeping Rochester's computers small, the mainframe would look more attractive to emerging large accounts. These were undeniably good business strategies and may have worked if IBM's customers were not given a voice. During the time, both parties (GSD heritage and SPD mainframe) knew there was a fight going on. The secret game was fraught with encroachment, illegal procedures, off-sides, holding, and roughing the passer and other violations. But since mainframe IBM was in control, no flags were thrown.

Five General-Purpose Computers

The whole notion of a general-purpose computer is one that can do anything. You simply put in a different program and the machine does a different thing. It was hard to believe that IBM actually had five different competing systems to cover one small business sector. Since these computers were truly general-purpose, though the IBM 8100 was targeted for a special purpose, a reasonable prudent being would conclude that there was definitely no need for five distinct families.

Ten Operating Systems

The one major item that IBM could not overcome, as it scrambled to combine all these boxes into one, was that the animals had already been released. Customers who trusted IBM were already using these machines. Each machine had at least one operating system. In fact, the mainframe had three, the 8100 had two, the Series/1 had three, and the System/36 and the System/38 had one apiece. Since program compatibility depends on the program's relationship with its operating system, the merge problem was not simply five systems that had to be accommodated; it was 10 operating systems and tons of programs built over many years for each of the operating systems. If it could be done, it would be a large project indeed.

Ironically, the problem of incompatible systems and software, which the Future System project in the early 1970s was convened to solve for the big systems division, had reared its head in spades in the little systems division (three systems and five operating systems). When you add the encroachment penalty exacted by the big systems division by building two of its own competing little systems with distinct operating systems, there was a new problem in IBM that was 10 times as bad as the problem that Future System was supposed to fix.

How could it possibly happen that a company could wake up one day and find itself with five competing computer lines with 10 different operating systems looking for the same customer? It is no wonder that IBM wanted to solve this. Though nobody at the time wanted their favorite system eliminated by consolidation, you can't blame Big Blue for wanting a solution.

In this chapter, we have not even discussed the big water-cooled, Poughkeepsie-built IBM mainframes and the PC-like IBM 5100 or 5120 Data Master computers, which were also very much alive at the time.

Add to that the fact that IBM had not yet shipped its first PC Server and the company had not yet entered the RISC processor / Unix business, and it is obvious that the situation was more out of hand than even five little systems and 10 operating systems.

Figure 14-1 IBM 5100 Small Business Computer

You Need Fort Knox to Solve That

IBM decided to invest a ton of money on studying the matter for a resolution. With all the money to be spent on the project, it was understandable that IBM dubbed the project **Fort Knox**. The objective of the study was to develop a convergence strategy so that one box would emerge with the powers and capabilities of all five boxes. It was a noble goal, but the project was virtually managed or perhaps unmanaged from the get-go. IBM really could not depend on the good will of the participating labs to assure that the best solution, if any, was brought forward.

Four separate IBM laboratories devoted some of their best scientists and engineers to the Fort Knox convergence project. Most of the work on the project was performed remotely in the home laboratories of the participants. Because the mainframe captains inside IBM expected that, as a side benefit of the project, they would be able to eliminate the

System/38 and the System/36 lines with the expected capabilities of Fort Knox, Rochester was not asked to be a prime player in the project.

There is some undocumented history in IBM that suggests that there were powerful people in the mainframe division of IBM who did not expect Rochester to have a requirement for any scientists or engineers after the Fort Knox hybrid was introduced. Some insiders would argue that Fort Knox was a vehicle that mainframe IBM was using to free the corporation from the tyranny of the "wild ducks" in Rochester.

Mainframe Had Specific Objectives

The mainframe engineers and scientists had to be careful. There was always the possibility that those renegades from Rochester were right on the money. Right or wrong, GSD systems engineers and Rochester's customers always believed that the Rochester Lab, if given the chance, could do just about everything better than any other company in the computing industry, including the rest of IBM. The danger for Rochester at the time was that such non-mainstream (non-mainframe) thinking did not play well in the mainframe-dominated halls of IBM.

Rochester did get a few sparse assignments in the Fort Knox venture, but the Rochester scientists involved believed that the project was doomed to failure from the beginning. There was no real management of the project, as each lab more or less did its own thing to attempt to achieve its own objectives. It is no wonder that, as the ideas were brought forth, each lab saw its product as the heartbeat component of the Fort Knox project. Though Fort Knox was supposed to have its own converged RISC processor, each lab's design had a different shining star.

Because the new Fort Knox processor would not run any of the programs from any of the other systems or operating systems, the team decided to build one compatible coprocessor for each of the variants to cover all of the facilities that the main converged processor could not handle. The Rochester scientists and engineers believed intrinsically that it was not going to work, and so that team spent most of its time working on the notions that they might use in a follow-up convergence of its own System/38 and System/36 lines. Not much input from Rochester was permitted nor given in the overall Fort Knox schema.

The goals of Fort Knox were quite lofty, and with no one identifiable person historically pegged as the project head, the efforts naturally took on the characteristics of the participant systems. Instead of harmony and convergence, this structure created home-team pride and division.

As noted above, but still somewhat laughable and worthy of repeating, in order to supply the five personalities that the participants believed were necessary for the box, it was determined that personality co-processors would be required. So if it were ever built, you would buy a Fort Knox with its RISC processor, and then buy a System/38, a System/36, a Series/1, an 8100, or a 43XX co-processor if you wanted to run any of those programs. This was cause for some pretty good jokes in the lab and in the field, which is where I toiled.

Instead of the five one-humped camels that Fort Knox began with, the end vision became one camel with five humps.

So after four years, the best that could be done from all that work was a common hardware infrastructure (bus) developed by the Series/1 group so that all processors could use the same housing. However, the real issues of program compatibility were not addressed. The five-humped camel may very well have had just one operating system driving it, but the operating system, like the camel itself, would need 10 personalities. It would be a difficult task to keep that level of complexity from appearing in end-user form, making the machine, if it were ever built, at best, unwieldy to use.

As noted, the software compatibility issue could be solved initially, at least theoretically, by including a processor for each personality that the machine needed. So if a software package that was needed for a former System/38 happened to be written for an 8100, the 8100 processor personality would have to run the package. Theoretically, five different software packages might require all five different coprocessors. Unlike Intel 8088s and 80286s of the day, midrange processors were quite expensive in the mid 1980s. So there would certainly be a big cost penalty for IBM customers if the mix-and-match-software notion could possibly be accommodated by Fort Knox.

Fort Knox Laid to Rest

As history has recorded the big Fort Knox event, nobody would ever find out for sure if it had any value at all. After four years of work, and hundreds of million in IBM gold, Fort Knox was put to rest. It had failed, as many expected. There was no real convergence product on the drawing board to prove that the project ever existed. Oh, there were bits and pieces of usable parts all over the place in the various labs associated with the project. Some of the pieces all of the labs knew about, and some were undisclosed.

The worst result of the Fort Knox divergence was that during all the time that Fort Knox was the anointed future, IBM's small system future was not being addressed. While Fort Knox was preoccupied for four years trying to figure out how to get rid of the System/38 legacy by design, DEC, HP, WANG, and others were trying to take down all five IBM systems by outselling IBM to the same customer set.. With no real follow-on products coming from the Fort Knox venture, IBM had dug itself into a hole compared with its competition. It would take a yeoman effort to get the small systems divisions back on track with their natural follow-on products.

Just as in Orwell's *Animal Farm*, however, some computer divisions were more equal than others. By the end of 1986, just after the collapse of Fort Knox, the Endicott plant miraculously was able to announce its successor system to the 43XX line. It sure did not take it long. The plant used a bus architecture that was developed for Fort Knox in the Series/1 division, added an I/O rack complex, and built a new, small mainframe processor, more powerful than the 43XX series. It was basically completed when Fort Knox collapsed. The processor that was built was to have been the mainframe co-processor in Fort Knox. They slapped the same three IBM mainframe operating systems on the package and had their follow-on product. They called it the IBM 9370. It was completed with minimal effort, in minimal time.

Even before the AS/400 was announced, the mainframe division made its second strike. At least in some place Fort Knox seemed to be paying off. The IBM Systems Products Division in Endicott, NY was able to converge its 8100 system onto the back of the new 9370. In March 1988, when IBM implemented the 8100 operating system called DPPX/370 for the 9370, the 8100 as a separate machine was no longer needed. The new capabilities on the 9370 permitted it to perform all of the necessary functions of the 8100 for a reasonable price. Thus, no follow-on product

was required. Fort Knox was responsible for the convergence of the two-mainframe system lines into one. Of course, it could be argued that since Fort Knox was a mainframe driven project, it is understandable that the Fort Knox entrails that were usable first happened to be mainframe vintage.

Since the Rochester lab had become a computer shop, because of clandestine activity, it had learned well. While it had no funding for the research and development of the System/38 or System/36 replacement systems, the lab persisted in moving a secret 3X convergence effort through Rochester, while being mostly excluded from the Fort Knox project. When Fort Knox failed, the lab was quickly called on to create its follow-on system. Without the clandestine meetings and theory testing that went on during Fort Knox, Rochester would have had a more difficult time performing this task than if it had chosen to swim Silverlake.

> Note Silver Lake is the lake in Rochester Minnesota that the famous Silverlake project (AS/400) was named after. It is an artificial lake coming to life as part of the damming of a portion of the South Fork Zumbro River just below where the river merges with Silver Creek near the city center. The lake had been used extensively over the years as a cooling pond for a nearby electrical plant. The inviting warmth of Silver Lake was noticed by the many Canadian Geese who would fly by on their way south. Over time, the geese came to like the warmth of Silver Lake so much that they ended their trip south in Rochester. Now, in the winter time 30,000 Giant Canada geese make Silver Lake their home.

The all-everything machine had survived its greatest attack and was now poised for greater achievements.

Chapter 15

The Silverlake Project

The Search for the Follow-On

The failure of an important project can create the emergency atmosphere needed to ensure the success of a follow-on effort. Fort Knox was a major failure, and even though Rochester was not to blame, the funding that Fort Knox gobbled up included what would have been used for the lab's next set of products. Rochester was hurting for a follow-on machine to the System/38 and even more so, the System/36. There were several hundred thousand System/36 customers, some of which had been hanging at maximum capacity for way too long. There were only 20,000 to 50,000 System/38s. Nonetheless, the needs of all these customers had been neglected for four long years during the travails of Fort Knox, and it was about time that they reappeared on the planning charts.

The goal of Fort Knox to unify the hodgepodge of smaller IBM computers was no longer important, but the company was still looking for a computer star to put DEC and its imitators on the run. These guys were growing too fat, at the expense of IBM. During the Fort Knox fiasco, these competitors had stolen many prospects and installed accounts from IBM. The mighty IBM had noticed and was alarmed, and through Rochester it was preparing to strike back.

The Silverlake Project Begins

With the death of Fort Knox, there were some 5,000 IBM employees in Rochester Minnesota, scientists and engineers included, who could

breathe a sigh of relief that all their work would not be going to Endicott. It took little time for the realization of freedom to reach them, and the hard working people from Rochester knew what they had to do. In little time, the Silverlake Project, for the convergence of the System/38 and System/36 computer lines, was underway.

From the outside, it appeared that all of IBM was in a state of confusion after the Fort Knox bust. However, some smart people in Rochester had already been doing some work in preparation for the ultimate demise of Fort Knox and the need for a new set of products for the midrange. Many of you already know that the outcome of the Silverlake Project was the Application System/400, or AS/400.

So how did Rochester pull it off? During the off-season (the four years of Fort Knox), the lab in Rochester had not stopped thinking about its future. Though no future was funded, a few clandestine projects were wrapping up, and the conclusions began to make a lot of sense. Among other things, Rochester had "discovered" that through an emulation mechanism it had built, System/36 programs could run under the control of the System/38. Though this function would never be released for the aging System/38, it would become the key difference between the System/38 and the AS/400-iSeries-i5.

Need More Powerful Processors

After four years of no hardware development, Rochester needed more than just System/36 emulation facilities to have a successful renaissance. It needed lots more than software in order to stay alive. It needed a new processor, new packaging, new hardware architecture, as well as higher-speed, higher-capacity input/output circuitry and devices. This is where most of the project work was going to take place. Of course, after the hardware was developed, Rochester would also have to extend the operating system to support the new system hardware. The Technology Independent Machine Interface, as explained in Chapter 8 would help make this a less arduous task.

Following the Fort Knox debacle, recognizing that there would never be funding enough for both a System/38 follow-on and a System/36 follow-on, the Lab decided, and gained the approval of Steven Schwartz, an IBM general manager, to build the combined follow-on product. Along with the okay, came about $1 billion in funding. This did not come without

the Lab making a few promises to IBM. Among the many promises, Rochester would take as many pieces of Fort Knox as were readily usable, including its own System/36 emulation, and the Lab would work on a follow-on product that would combine the strengths of the System/36 and the System/38.

The resulting box would not be Fort Knox, but it would accomplish the consolidation of two very important small business computer lines. Without the mainframe part of Fort Knox, Rochester knew that it could handle this convergence mission. Rochester had complete control of the project, the resources, and its own destiny to ensure that it would actually happen, and that it would happen right.

A big plus was that there were no mainframe folks on the design committees whose agenda was for Endicott to assume Rochester's role in the company. IBM had been beat up so badly by DEC and company during the do-nothing years that just about everybody in the company now wanted to take out DEC a lot more than it wanted to take out Rochester.

Silverlake Goals

In Rochester, a handful of engineers and scientists, Dr. Frank Soltis among them, formed an ad hoc group and brought forth a plan for the next generation of Rochester computers.

The ground rules that the team had to work from were clear and formidable:

1. The product they designed had to be on the market within two years.
2. As much as possible were to be salvaged from the Fort Knox wreckage.
3. A proof-of-concept prototype was needed.

This was the hallmark of the Silverlake project. I lived through these times in the marketing arm of IBM as a systems engineer in the local branch sales office. Rumor after rumor was flowing out of Rochester at the time. It seemed that IBM had lifted the cloak of silence from the plant to help stop the erosion of clients from the System/36 and the System/38.

Promises, Promises

Rochester had promised Steve Schwartz and the rest of IBM that it could deliver a new system in record time. To help get the job done, IBM went through the Fort Knox shopping cart and found a few items that could help. Among them was something called the zero insertion force (ZIF) packaging cages that permitted solid logic cards to be more reliable and easier to insert and remove. The cage, and thus the logic cards, had contacts on three of four sides. This was the same rack and cage technology that was used in the 9370, direct from the vault of Fort Knox, and was developed for Fort Knox by the Series/1 Lab in Boca Raton, Florida.

There was always somebody who wanted to give Fort Knox the appearance of having been a project with some merit. Powerful players in IBM with their Fort Knox t-shirts and leftover mentality imposed their will on the Rochester lab. Of course, to get its systems approved, the lab acquiesced. As I recall, the early AS/400 units shipped to customers had some hardware issues related to the ZIF technology. They were very sensitive and created downtime issues on early shipments. Eventually, this was corrected and the Systems Product Division (SPD) bus, developed by the Series/1 lab, stayed with the AS/400-iSeries until just a few years ago.

A Project Full of Lesser Heroes

With a number of heroes working many long shifts per week, Rochester was able to complete the hardware and the software for the new system in little more than two years. As noted, a follow-on system change was always a big project, typically completed in the five-year range. When Rochester brought this in, kudos were bestowed on the lab from all parts of IBM for bringing in the project in record time.

At the time, at $1 billion in funding, it was the most expensive computer development project ever undertaken by Rochester, but the goals were lofty. The two-year time frame was less than half of the normal time. So there was a high degree of risk with the project. However, unlike the Pacific (System/38) project that was an exercise in one-of-a-kind, never-been-done-before computing, the Silverlake project was an extension of an existing all-everything machine architecture.

Without minimizing the Silverlake project at all, since it was ambitious and required a yeoman effort by Rochester, all of the extremely creative foundation work had already been done with the introduction of the System/38. Building the follow-on hardware and integrating it with the existing System/38 operating system was a big challenge in itself, especially with just a two year window. And, Rochester was up to the task..

AS/400: An Instant Success

In June 1988, two months later than promised, the AS/400 family of computers was announced. As I have said several times in this book, I had the personal pleasure of presenting the announcement in the IBM Scranton location. Our announcement was held at Marywood University's Center for the Performing Arts. It was a beautiful setting. Hundreds of IBM field managers, marketing representatives, and systems engineers from across the world presented the good news of the AS/400 to several million people in just that one day. IBM was very serious about the AS/400 becoming an overnight success. It did. It's amazing how successful a company can make something when it intends to.

The AS/400 quickly became one of the most popular computer product lines ever introduced. With the follow-on iSeries and i5 products, it remains a major source of profitability for IBM today. The Silverlake project was highly ambitious, but the lab had placed strict controls in place to keep its scope from spinning out of control. Instead of everything being invented from the ground up, key building blocks were taken from anywhere they were available, including the completed parts of the Fort Knox project, as well as from the two products being replaced: the System/36 and the System/38 minicomputers.

Nothing about Silverlake, which by announcement day was called Olympic, looked like the System/36 or the System/38. It was a rack-mounted unit compared with the two console type systems that it had replaced. It had the same physical appearance as a 9370, because it used the same racks and cages, but internally it was completely different.

It really did not matter that there were no big functional announcements with the system. It was substantially faster and had substantially more capacity than both the System/36 and the System/38. Industry watchers

were able to observe the big performance-limiting thumb of mainframe IBM ease off the Rochester lab for this brief period, as it was able to announce and deliver a powerful hardware solution that was about to knock the socks off the competition.

Note: Projects in IBM are given interesting code names at their inception. *Fort Knox, NorthStar,* and *Silverlak*e are several examples. It was so important for IBM to prevent the erosion of System/38 and System/36 accounts from IBM that the company was leaking internally facts about the project and everybody knew it was Silverlake. IBM's field sales force was leaking externally to its customers that something good was on its way. Silverlake was so over-hyped as it approached reality that the company changed the project name to "Olympic" before the product that would emerge from the project was ready for announcement.

Because all of the hardware was being replaced, major software enhancements were pushed off until after the basic system was operational. You've got to credit Rochester with doing what it had to do to get a definitely new system out on the market on time, with lots of hoopla. There were few who did not notice that IBM was in business big time with its new AS/400. It was a combination of Tom Furey's (Rochester Lab director at the time) great leadership in the Lab, a tight schedule, rigid scope control, extensive reuse of Fort Knox and other existing technology, and an exceptionally motivated project team. Silverlake remains one of the great success stories within the IT industry.

Silverlake is also a great testimonial to the Rochester Laboratory and what it could actually do if IBM were willing to free it from the constraints of bondage designed to prevent competition with the mainframe.

The mainframe and the IBM i5 today are the only proprietary server lines left in IBM. IBM invented both systems, and both machines use all-IBM technology. Neither platform needs Intel for a processor or Microsoft or Unix for an operating system. Many speculate that with the introduction of POWER6 technology, the mainframe will be using the same processor as the i5. Thus, the makers of the most desirable and most powerful and reliable computers ever built, (eServer zSeries and IBM i5) will soon share a hardware package. Both Divisions can then have a major role in

cooperating for the IBM Company's success, and my bet is that IBM stock will be an even better buy when all this happens.

Chapter 16

The Cost of Owning the All-Everything Machine

Not Yet a Million Sold

IBM anticipates selling about 50,000 i5 systems each and every year. Recently the number has been as high as 90,000. With less than 100,000 units per year, one can see that the all-everything machine is far from a commodity, and thus many of its unique parts are not acquired or built at a commodity price. So, quite naturally, the box costs somewhat more for IBM to build than if the quantity shipped were more like one million units per year. However, it is also true that IBM, to replace one-of-a-kind parts, is using more and more commodity components. So the system's overall cost is coming down. Moreover, with Sony and Microsoft using POWERPC chip technology for their game products, the cost of developing and manufacturing the i5 processor is also coming down substantially.

If you ask an IBM i5 customer, you will find that they know intrinsically that the money they pay for their system is well worth the price. First of all, they don't have to worry about the system going down several times a week. It's always available for business. They don't worry about the system becoming locked up and having to be re-booted and, unlike Windows units, they don't have to worry about the machine being down for almost three weeks of every year. Moreover, because it is a multi-user server machine, IBM i5 staffs do their support thing to just one machine and do not have to worry about a "farm" of hundreds or thousands of independent PCs to accomplish the company's business processing mission.

If It Costs More, Doesn't It Cost More?

IBM has had a hard time over the years selling the industry on why the same hardware costs more in its prior systems (AS/400-iSeries) than any other machine. Over time, with the help of various industry consultants IBM has been able to formally quantify some of the value of its systems using a technique known as the *total cost of ownership*. The moral is that all of the things that you must pay for in a non-i5 solution are part of the total cost of computing and the things you get that you don't have to pay for with an i5 should be considered in the formula.

One of the best ways to demonstrate the substantial cost difference between the i5 and all other machines is to discuss what makes machines expensive and what makes them inexpensive. To do this, let's look at something that has become very familiar to many of us over the last 20 years: the proliferation of PCs in business and the costs that are incurred in various areas of the company when PCs, PC servers, and PC networks are deployed. This will form a basis from which to calculate the total cost of computing.

PCs Cost a Buzillion Per Year

There are numerous issues that all cost money when companies must deal with PC networks and PC servers. Some are easy to spot while others may be invisible until they take their toll. Let's take a look at some of the common PC issues in many corporate offices today.

Independent Islands of Office Computing

Each PC is an island. Each user in an office believes that the PC on his desk is his to use as he sees fit. Though certain office suites may be installed on each PC, many PC staffs do not cripple (lock down) the users' ability to modify and customize the application to their own pleasure. This freedom creates anarchy for the organization in that after a while, no two systems are the same. It may be a boon for individual creativity and specialization for the individual user but it is anathema to comprehensive, coordinated technical support **when** the PC begins to act funny. The support staff must research each individual PC when a down situation arises because there is a lack of common software and common

components. This causes extended downtime. Extended downtime costs the company in at least two ways:

1. More support resource is used to fix the problem
2. The user, whose PC is being worked on, is out of business for a longer down time.

Since the company pays for this wasted effort, its effect on the cost of computing is amplified.

Recovering Corporate Data Assets

There are always problems with PC networks. It is axiomatic that a PC-based network creates problems. A manager who denies that the installation has any problems either is the exception to the average or has a consulting company fixing his problems. Things break and there is a higher propensity for them to break when PCs and PC servers are networked together.

Hardware breaks, software crashes, viruses infect, and data become corrupted. Each organization must have a tried and tested plan for backup and recovery or such problems become nightmares. The PC network that is built on the cheap often has no plan for down time or recovery. It is even a problem in many installations to suggest that a particular somebody is responsible for daily backups. It is an even bigger problem to require recovery procedures. And, how do you test the recovery procedures if they do exist? There are lots of issues in recovery such as locating the original installation applications and the operating systems. If there is no real plan, the CDs for these applications may not be in a secure location? Then after all is said and done, when the machine does go down, who gets to re-install everything and who gets to recover the corrupted data? The gentle recovery plan in most PC shops includes no names. And the longer a server or a client station is down, the more it costs the company.

The less likely a system or network is going to suffer from these situations, the less it costs the company. If a PC server based network is the system of choice then the firm can no doubt expect to save money on hardware and perhaps on software compared to an i5 solution. However, the cost of recovery and even worse, the inability to recover and operate

effectively adds substantially to the overall cost of ownership and quickly exceeds the cost of owning an IBM i5.

The Network Impact

When a company has standalone PCs and it chooses to network them all at once, it's got a better chance of creating a reasonably reliable "error-free" company network than if the network is homegrown piecemeal, one system at a time. Yet many networks start with several units and are expanded by need, not by plan. It doesn't take too long for a company to realize that it needs a person to take care of day-to-day network operations and there are often issues when the responsible party is out of the office. Companies need emergency plans in case the network crashes or the manager suddenly cannot do the job.

Since the data needed for daily operations is often not on the user's PC, the network is needed to get to the proper server and the proper printer for the job. Having the network up is just as important as the backup and recovery procedure for applications and data. What happens when network components or strategic server PCs go down? The company must have hardware maintenance agreements for both the PCs and the network. If the typical response for service is measured in days, there is a definite cost of not being able to do business as well as having idle personnel.

The PC Impact

A PC is not a PC is not a PC. You may have heard the old adage that you get what you pay for. When you are adding up the costs of computing, there's another adage you should keep in mind. The lower the price of the PC, the less reliable the unit and service will be. If you find a bunch of el-cheapo white boxes from the guy down the street, you may be a temporary hero in the organization but your day in the sun will be short lived. And the clouds will come!

The total cost of ownership of a PC is inversely proportional to its cost. The majority of inexpensive PCs are cheap PCs. The costs come when the machine fails or applications fail and you are looking for the CDs that are supposed to help you recover, but there are none or they are incomplete. Try looking for device drivers for your PCs when they are not supplied on removable media such as CDs. The options are few

when you don't have what you need. You must ask yourself if your company can afford to buy new hardware, wait to get it in and have it installed for each PC that needs recovery. If you can afford the costs, the next question is whether there is a company nearby or a person nearby that you can count on to perform the needed installation / recovery tasks in a timely fashion. Sometimes there is and sometimes there is not.

Downtime Impact

We have introduced the notion of downtime in a number of the areas already discussed. It is a critical factor in the success of a PC server based network. You must be able to handle downtime by plan or your business will suffer and it will cost you more than you'd ever want or expect to pay.

When PCs fail, your users are forced to work at a lowered level of efficiency and their effectiveness is also reduced. Their expected work products drop in direct proportion to the degree they need a computer to get their jobs done. Downtime on the network or on any required network resource such as a printer, fax server or Internet connection can create a critical business situation. This costs the company real dollars.

The i5 Solves Most PC Problems

Though none of the above scenarios can be viewed as positive, if the system or network provides no value, then the cost of downtime can be minimal. This is not a joke. There are many companies who get sub par information and processing assistance from their own PC network because it is also easier to buy inferior software or simply use spreadsheets for many business functions.

All companies who use computers effectively, however, have what are called mission critical business applications and these applications must be run on a stable and reliable platform such as an i5. When a company does a careful evaluation of all the costs involved in computing, including downtime and recovery, the total cost of ownership is much higher for a PC when used for serious network application serving than when using an i5 as the main system.

Even in organizations in which PCs are the principle desktop machines, those who carefully evaluate their mission critical needs trust them to an

IBM i5 rather than risk disaster with a PC solution. The IBM i5 has a 99.9% reliability rating for uptime and it serves to lower the total cost of ownership (TCO) over the course of its use.

When an IBM i5 is used, the data and programs are secure and continually available to the users. The i5's unprecedented reliability addresses the one major single point of failure. Of course, even with an i5, disaster recovery plans need to be prepared and tested and updated, but the chances of needing the plans are substantially reduced as is the overall cost of doing business.

Being Good Lowers TCO

As we have shown in the previous chapters, there are plenty of reasons why a business should want to deploy the all-everything machine compared to other less reliable choices for its daily computer work. Understanding fully the notion of TCO can help businesses appreciate that they not only can afford the all-everything machine, but they really can't afford the real cost of "cheaper" systems. Experience suggests that in almost all cases it cost less for an i5 than a "cheaper" system.

Chapter 17

The Future of the All-Everything Machine

Lead With the Best Product

If I were sitting at the top of IBM right now, having watched what looks to be about $30 billion in hardware business fritter away since Lou Gerstner's arrival in 1993, I would look to recapture that $30 billion that is no longer mine. Those dollars, however, are not going to come from PCs. It's too late for IBM in the PC area. Finally IBM is out of the consumer PC business, a major distraction for its server business. Those dollars however, can certainly come from displacing PC servers from Dell, and from HP and from Sun with the all-everything machine.

There is also a tremendous opportunity for IBM's small business solutions in new accounts. As many already know, there is one machine that is uniquely qualified to be IBM's lead dog in its revenue reclamation project. The same trusty i5 heritage machine that killed DEC can again be used in the form of the new i5 as the secret weapon to bring back all that lost cash. With IBM's May 4, 2004 announcement in which the AS/400-iSeries became the eServer i5, I am counting on Big Blue already having this work in process.

Corporate Strategy or Accident?

About a quarter million IBM customers already know about the IBM i5.. Most IBM i5 and AS/400 watchers observe the all-everything machine as IBM's best-kept secret and most assume that the company would like nothing more than to have everybody know about it, as long as its other servers are not diminished. Apparently IBM has not fully figured out how to do that yet. However, with the new i5 and recent major changes within the iSeries organization in the presence of Mike Borman, the former iSeries chief, and Mark Shearer, the new iSeries chief, and Peter Bingaman the new marketing maestro, IBM is again positioned to lift the all-everything machine from obscurity to being boldly displayed on your living-room TV.

IBM surely knows how to help the i5 to be more part of its mainstream-computing scene, and I am counting on it happening. Big Blue can no longer afford to have the i5 as a back room after-thought with Bill Gates fully prepared to collect i5 customers if IBM chooses to not fight his advances to woo IBM i5 customers to Windows. I expect IBM in the short term to show Microsoft that Big Blue is not easy pickings and to use the all-everything machine to simply blow Microsoft away.

For almost five years now, the IBM Company has spent its eServer marketing dollars pumping up the eServer brand. With the i5 in the lineup, and the recent management changes, IBM has already begun the i5 renaissance by advertising its eServer i5 on TV – for the first time in many years. It's just a start but a good start. When IBM perfects its ad message, and informs even non AS/400-iSeries-i5 users and small businesses that the IBM i5 is a fine business machine, Windows dominance will no longer be a sure thing.

Prove to Me You Love Me

If IBM were to testify in court today about whether it supports the i5 product line in a big way, the answer is in. Though the company does not always get credit for all the good it does, IBM would win in court, hands down.

Many in the AS/400-iSeries crowd have complained about IBM not pushing its i5 heritage product line hard enough over the last five years. It is possible; however, that IBM has been spending its dollars exactly where it should. For example, if IBM had to go to court tomorrow and testify to defend what it has done for the eServer i5 lately, its testimony would actually be compelling and impressive. After all, from 1995, IBM has invested tons of money into the platform. For example, it transformed the AS/400 server from a 48-bit hardware platform running on CISC architecture to a 64-bit hardware platform, first in the industry, running on RISC architecture. Through the 1990's IBM has walked the system through nine iterations of POWER chip technology and brought it to the POWER5 level, which is the envy of the industry. Moreover, IBM has discussed its vision of the POWER Architecture with POWER6 and POWER7 technology coming on board the i5 boxes out to the year 2010. Now, that sure does not appear like a company that is not advancing its flagship.

IBM has also marketed the AS/400-iSeries-i5 though perhaps not in the same light as its customers would request. Under oath, again IBM would come out shining like a rose. In the last part of the 1990s the company included the AS/400 in its expensive "magic box" ad campaign and highlighted its unique capabilities, such as Domino support. In the year 2000, again IBM spent tons of marketing dollars on the AS/400-iSeries as it re-branded the unit and it included (not excluded) the new iSeries under its massive eServer umbrella.

Thus, IBM can argue that when someone sees the eServer brand, they can carry that on down to the specific models and there it is-- the iSeries, and now the i5 as one of the included brands. A rational and prudent Judge would find in IBM's favor. So, what can IBM do better that it's not doing now? The future for the i5 and follow-on versions of the all-everything machine would be brighter indeed if IBM embarked on a more focused advertising campaign, the objective of which would be for the executive in the board room to know about the all-everything machine from having heard about it in his or her living room.

IBM was once able to help its prospects get under the eServer umbrella and under the covers of the AS/400-iSeries to see its uniqueness and elegance. As IBM again figures out how to get this architecture message out to the masses, it has the opportunity to gain great rewards and to defeat Microsoft in its weakest areas- technical facility and reliability.

We spent the early part this book highlighting the wonderful unique features of the IBM i5 that very few regular people even understand. Even high tech people do not understand the value of these facilities or they would be using i5 technology instead of inferior machines. When IBM again figures out how to get this message across, and that time is coming soon, it will move the acceptance of the i5 platform ahead in leaps and bounds.

No other system has capability based addressing or single level store or a full object orientation, or an integrated relational database, so IBM marketing should have little problem in getting this message across now that the company is doing i5 TV ads again.

The fact that IBM can exact a premium for its i5 has not been lost on the marketers in IBM today. Though its market share is not increasing, its many customers love the i5 product set more and more each day, and they are repeat buyers. This means that investments in i5 applications are safe, and that IBM will continue to support and enhance the i5 product line

Linux is IBM's eServer OS

Under this backdrop, it is very logical to conclude that IBM's enhancement of its i5 product line must be conscious. It must be intentional. The facts about what IBM is up to with all of the advanced enhancements to the i5 are becoming clearer and clearer and the facts explain quite a bit of IBM's behavior regarding the i5. There is a major assimilation agenda at work at IBM in hardware, software, and branding. It is not at the level of the Fort Knox project but there is no doubt that IBM is again trying to get the most from its development dollars.

Just the other day I attended an IBM presentation in which the speaker highlighted the new i5 models as being that much better because to run Linux applications, the new models would not need i5/OS, the standard i5 operating system. Hah! I thought to myself. The new "hypervisor" for i5 does not need the i5/OS operating system. That makes things very interesting.

Some say that IBM has almost fully completed its hardware convergence (or blending) of the eServer iSeries i5 type units with the eServer pSeries p5 type units. Since the 1990's both products have been rolling off the same production line in Rochester Minnesota, they both have been using the same POWER processors for some time now, and they share more and more parts with each change iteration. Observers say that you can not tell the difference between the two machines as they are coming off the Rochester manufacturing line.

On December 10, 2004 IBM took a bold step in the ultimate convergence of all its server lines to one. The company began to ship a special CD with its larger eServer p5 models. Until this time, these p5 servers could run just Linux or Unix. Now, with the CD containing i5/OS, the IBM Unix box, known as the p5 can now run i5/OS applications in several of its partitions. Of course there are limitations. First of all, there is a charge for the CD and it does not work on all p5s and those with which it does work can use it only on two processors. However, those of us who have seen IBM marketing ploys in the past know that these restrictions are artificial and for marketing purposes only. There is no real reason why the i5 and the p5 units cannot be identical. In fact, the may already be identical. This says that the convergence of the p5 and i5 hardware is either complete or it is well on its way to completion.

Linux Can Make Development Better at IBM

If you are IBM, and you've got your hardware act complete for two key server systems, and full platform convergence is your goal, your next logical step would naturally be to unite the operating systems and other software. However, since users recognize their systems by their OS

personality, this has repercussions that a hardware merge would not have. IBM is not about to forget its System/36 lesson when this community of loyal IBM users refused to fully embrace the System/38 as the replacement product. So, with convergence as its goal, IBM will have to approach this matter discretely and with great concern to protect its customers perceived operating environments.

Eventually for full convergence, Windows and mainframe operating systems will be enabled to run on the same hardware as the i5 and the p5. If you are IBM, you would find a great facilitating operating system already available and waiting to become the software convergence vehicle for all other operating systems. Rather than eliminate the operating systems, IBM has an opportunity to build common middleware for all of its systems and place this ware in a hidden Linux partition on the POWERX iSeries.

It makes sense that if IBM can build software just once for the Linux environment, since Linux runs on all of IBM's platforms, even the i5/p5 hardware platform, its development job for all platforms could be done just once. With function made available through Linux and partitioning, when Windows and Mainframe z/OS run on POWER6 processors in the near future, there would be no need for multiple versions of IBM middleware such as MQSeries. Middleware could be written once for Linux.

There would be no z/OS version, no i5/OS version, no Unix version, and no Windows version of IBM middleware required. If you are IBM and you write your software (DB2, CICS, WebSphere etc.) so it works for Linux, and you have made Linux work on all your servers, it follows that your software works on all your servers – and you have just written it once, not four or five times.

What a deal for IBM? It's so good that even an IBM i5 buff can see its logic. In the Linux scenario, IBM would build once for Linux; put Linux on every eServer in a real or invisible partition, and save the huge development dollars required for three or four operating system ports. That is a lot of money. For the i5 aficionado, if IBM does it right, there would be no loss of OS/400 control and integration. IBM can make it so

there is no usability price or any additional price to pay for IBM's tremendous development savings.

i5/OS would remain stable and gain new function by adding application programming interfaces (APIs) for Linux cross partition support. These would be the same APIs that would be usable on all other systems. In other words, IBM would only have to build a few hooks so that i5/OS would seamlessly cross over to the Linux partition to gain access to any new functions. The local program need not even know. Just as IBM seamlessly integrated the Apache HTTP server within the AS/400-iSeries, it can do the same for the Linux functions accessed across the partition boundaries.

There is another possible advantage. Since IBM would have its software working on Linux, and Linux is an ala carte operating system (you build it), IBM's software division could get to sell the same one product across four different platforms, thereby increasing IBM's software revenue

IBM's own software development costs would become less than half of what they otherwise would be, plus there would not be four different labs trying to keep their software versions bug-free and up-to-date. When IBM hauls the mainframe personality to the POWER architecture in the next several years, all of IBM's three operating systems will be hardware and software compatible. The mainframe will be the Linux box. The business machine, IBM i5 will be the Linux box. The Unix box, IBM p5 will be the Linux box. The PC Server has been a Linux box all along.

No wonder IBM is pushing Linux on the i5! Even if it does not make sense for your business because you see no application value in Linux, you can see how this is a logical, coherent strategy for IBM. Your value will come from having advanced development features brought to all IBM platforms, including IBM's i5, sooner than if the company had to write the same function four or five times. It is clear that if this is IBM's plan, and it seems to be, it is not a product of ad hoc helter-skelter thinking. Therefore, you can almost expect that this is a lot more than rumor. This will happen.

The IBM Plan Is Rational

Try to imagine how IBM must think in this complex world. This company spent hundreds of millions of dollars trying to converge five product lines and 10 operating systems with Fort Knox in the early 1980s. This kind of expenditure was folly but it was not accidental. It was planned, but not planned well enough. In the 1980s, IBM was invincible, so it used bravado, not patience. It used revolution, not evolution. It used disruption, not harmony. For its hundreds of millions of dollars and for all the turmoil it caused and for its big failure, IBM came home with its early convergence objectives unaccomplished.

Though Big Blue was clearly unsuccessful with Fort Knox, its goals of having just one system capable of all functions never left the corporation. In its latest iteration, the fox, played by IBM, has been using harmony, evolution, and patience in achieving through time, what it could not do by decree. It's not there yet, but it is well on its way. With Windows and the mainframe on POWER6 as the last piece on the horizon, the all-everything machine will be complete.

Let's look at the evidence. From a hardware perspective, the i5 and the Unix box (p5) now share the same processor, hardware chassis, and major components. They are both made in Rochester, Minnesota. As noted above, the machines look almost identical coming off the line. They may even be identical as I write. The i5 can run AIX, the native OS on the p5. The p5 can run i5/OS, the native OS on the i5. This happened quietly, post Fort Knox, over the last eight years or so. In a few more years, the mainframe will be part of the POWER hardware mix. The mainframe integration and convergence project actually has a secret code name in IBM. It is the "universal" or "Mach 5" server. Many components in all three systems are basically the same already.

No Marketing Problem for IBM!

On the marketing side, IBM's marketing department has performed its job letter perfect. Some AS/400-iSeries folks think there is a big

marketing problem but inside IBMers know the execution has been perfect. IBM marketing has successfully created the eServer umbrella brand, which makes all systems the same from a marketing perspective. That is part of the overall unifying convergence strategy. The AS/400 is now an eServer and soon as the eServer, the all-everything machine, it will be able to run all of the applications that today run separately on all of IBM's other servers – mainframe, Windows, and Unix.

For almost five years, IBM has been advertising eServers, though there are no such things per se. The products exist in the series names. However, if all IBM hardware boxes actually become the same, and if Windows servers were made to run on the IBM POWER processors and not just Intel, there would be no need for series names. Even Intel's processors and PCs would be irrelevant in IBM's plans. The eServer i5 would run it all, and the i5 part would be dropped and it would be called simply the IBM eServer. Of course I would still call it the all-everything machine.

The All-In-One-Hardware eServer

The most important element that having the same hardware would provide is that the eServer would actually be a product. Again, there would be no need for a small letter series of computers. The eServer (AS/400) would actually be the Fort Knox II do-everything-machine that IBM dreamed about in 1981 but could not make happen then. There would be no need for co-processors since all of the "guest operating systems," z/OS for mainframe, OS/400 (i5/OS) for AS/400-iSeries, Linux and AIX (Unix) for the Unix box, and Windows for former PC Servers, would run natively on the same IBM POWERX-based processors at the same time in different partitions. Though this may seem an IBM dream outside of the realm of possibility, the ingredients are all in place.

Even if the hardware and the marketing were the same, the thing that separates the systems today is their respective operating systems. The mix includes

Mainframe	z/OS (formerly OS/390), VM, VSE, Linux
AS/400-iSeries	OS/400 (i5/OS), Linux, AIX
Unix Box	Unix (AIX), Linux
PC Server	Windows, Linux

The All-In-One-Software eServer

As noted previously in this chapter, today each of these boxes also runs one common operating system. You guessed it: Linux. If IBM were to concentrate its future on Linux in more ways than you could imagine, the primary OS for the all-in-one eServer box, the all-everything machine would naturally be Linux. All other operating personalities that were necessary could be worked in with guest services running in logical partitions of the big Linux eServer. That is a super technical achievement for IBM and it would make the all-everything machine a resounding success. However, it would have to be done very carefully to preserve the IBM i5 as its customers know and understand it.

Let's digress a bit here and remember where we got off. Today, IBM supports partitioning on its i5 and p5 systems though a mechanism known as a hypervisor that sits in a hardware box called a Hardware Maintenance Console (HMC). Perhaps IBM has already begun the Linux is invisible transition since the HMC is actually a Linux machine that IBM has locked down for its hypervisor functions. If this separate box were moved to a partition on the i5/p5, it could be invisible and provide the shared software facility that enables IBM to write once and use on many operating systems.

In order for i5/OS to use this facility without much issue at all, IBM needs to build another TIMI-like facility within the i5 or the i5 main partition. If IBM is looking for a name, I'd call it the ***MIMI***. That would be the middleware independent machine interface. It would be another virtual machine on the eServer. Its job would be to take the requests for standard function on i5/OS and if the software is not on the local i5 machine, the MIMI would find the software on the Linux partition. It

would also enable and send the request for a software service transaction and bring back the response from Linux to i5/OS to the user program.

The IBM i5 is known by its operating system, i5/OS, not by its hardware. IBM has shown over the years that, other than reliability, the hardware does not matter. The original eServer branding gave all machines the same hardware surname. A future IBM i6-like RISC box can actually be the converged all-everything hardware server running all personalities and running Linux as the smoothing factor. From IBM's perspective, that would complete the successful building of a converged system without the revolution caused by Fort Knox. Though IBM's accomplishments will have been done surreptitiously, for the AS/400-iSeries crowd it would be quite serendipitous.

The Big Three

If you get your checklist out, the three big convergence items are in process. (1) The convergence hardware is on its way. (2) The marketing piece is done. (3) Linux has gotten the call.

With the long-term strategy in place and firing on all four cylinders, what can IBM do in the meantime to soften the blow to an unwary constituency? Well, you might not be surprised that the company has actually been doing it for years. The IBM AS/400-iSeries-i5 message has focused on partitioning, Linux, and Unix for the last several years. The message is well out there. The message has been Linux, Java, on-demand computing, and logical partitions. This is not by accident. IBM is not just advertising to its customer set, it is advertising to competitive professionals and AS/400 professionals to get them ready for the big day when OS/400 and Linux get to live together on all eServer boxes on the all-everything machine.

A Rose by Any Other Name

AS/400-iSeries loyalists may initially resist IBM's call to view the new machine as a rose. Since an operating system gives a computer its personality, having Linux control a machine on which the words IBM eServer i5 is emblazoned will not easily convince the AS/400-iSeries-i5 crowd that the machine is an i5 box at heart. If it is running Linux, it is a Linux machine. If it is not running OS/400 as its primary OS, no matter what IBM calls it, is it really an AS/400-iSeries?

I predict that IBM will figure out how to sell this and that is not a big problem if IBM does it right. If a set of APIs were buried below i5/OS in the form of say, a Remote Virtual Machine (RVM) as in the notion of Java Virtual Machine, the TIMI, or the MIMI, i5/OS would not need to know the difference. The job of the RVM with the MIMI would be to accept those calls for support functions such as WebSphere, DB2, etc. and get them over to one or several companion Linux partitions or hypervisors for execution. Ideally, the controlling Linux partitions should be hidden from visibility and seen only through APIs to the MIMI in the RVM.

The high level languages operating above the RVM/MIMI would need to know nothing about how the new eServer gets its job done. Calls to functions such as WebSphere, MQSeries, etc., from i5/OS can be intercepted by the RVM/MIMI and executed in the Linux partition without the user program or the programmer knowing the difference. That's an easy sell.

There is no reason to develop every software facility four times when once under Linux will do quite well. By making the Linux function appear to be running from i5/OS, IBM will win over the IBM i5 crowd. Integration will take on new meaning as eServer shops use Linux from i5/OS machines as if the function were embedded in the native OS. This will help convince IBM i5 shops regardless of the reality that the similar look and feel of the new integrated OS with its API capabilities built into an RVM/MIMI is just as good as what they had. In other words, done right, IBM can show that the new all-everything machine is their ever-faithful AS/400-iSeries-i5 and lots, lots more.

The all-everything machine is poised to do it all. That adds up to big value for companies looking for a reliable server ready to do every possible computing job that there ever was or ever will be.

Index

LETS GO PUBLISH! Books:
(sold at www.itjungle.com, www.mcpressonline.com, www.iseriesnetwork.com)

LETS GO PUBLISH! www.letsgopublish.com is proud to announce that more AS/400 and iSeries books are becoming available to help you inexpensively address your AS/400 and iSeries education and training needs: Our titles include the following: email info@ letsgopublish.com for ordering information

Getting Started With The WebSphere Development Studio for iSeries .
Your introduction to the new IBM strategy for Application Development. Includes a case study and examples of UI / Logic separation and CPW savings techniques.

The iSeries Pocket Developers' Guide.
Comprehensive Pocket Guide to all of the AS/400 and iSeries development tools - DFU, SDA, etc. You'll also get a big bonus with chapters on Architecture, Work Management, and Subfile Coding.

The iSeries Pocket Database Guide.
Complete Pocket Guide to iSeries integrated relational database (DB2/400) – physical and logical files and DB operations - Union, Projection, Join, etc. Written in a part tutorial and part reference style, this book has tons of DDS coding samples, from the simple to the sublime.

The iSeries Pocket Query Guide.
If you have been spending money for years educating your Query users, and you find you are still spending, or you've given up, this book is right for you. This one QuikCourse covers all Query options.

Getting Started With The WebSphere Development Studio Client for iSeries (WDSc) Focus on client server and the Web. Your introduction to the client server and web development tools. Includes CODE/400, VisualAge RPG, CGI, WebFacing, and WebSphere Studio. Case study continues from the Interactive Book.

The iSeries Pocket WebFacing Primer.
This book gets you started immediately with WebFacing. A sample case study is used as the basis for a conversion to WebFacing. This interactive 5250 application is WebFaced in a case study format before your eyes. Either learn by reading the book or read while working along on your own system.

Migrating to WebSphere Express for iSeries: Your Roadmap for Migrating Applications to WebSphere Express
A Comprehensive guide designed to be your roadmap for moving to WAS Express for iSeries. It is loaded with examples and structured for easy learning. Through an easy to understand sample case study, you experience a real migration, and you learn the gotchas before they getcha! This book is designed to be a companion to all of your WAS Express migration efforts in the iSeries environment

Getting Started with WebSphere Express Server for iSeries: Your Step-by-Step Guide for Setting Up WAS Express Servers
A Comprehensive guide to setting up and using WebSphere Express. It is filled with examples, and structured in a tutorial fashion for easy learning. The book is designed to take you to a point

at which you understand the notion of a servlet server, what WebSphere Express is, where it came from, how to order it, how to set it up, and how to make it work in your shop.

The WebFacing Application Design & Development Guide:
The Step by Step Guide to designing green screen iSeries applications for the Web. This is both a systems design guide and a developers guide. Using this guide, you will understand how to design and develop Web applications using regular workstation interactive RPG or COBOL programs. When you learn the tricks, and observe the sample code in action, you might choose to develop all your applications using this approach.

The iSeries Express Web Implementor's Guide. Your one stop guide to ordering, installing, fixing, configuring, and using WebSphere Express, Apache, WebFacing, iSeries Access for Web, and HATS/LE.

Can the AS/400 Survive IBM?
Exciting book about the AS/400 in an iSeries World.