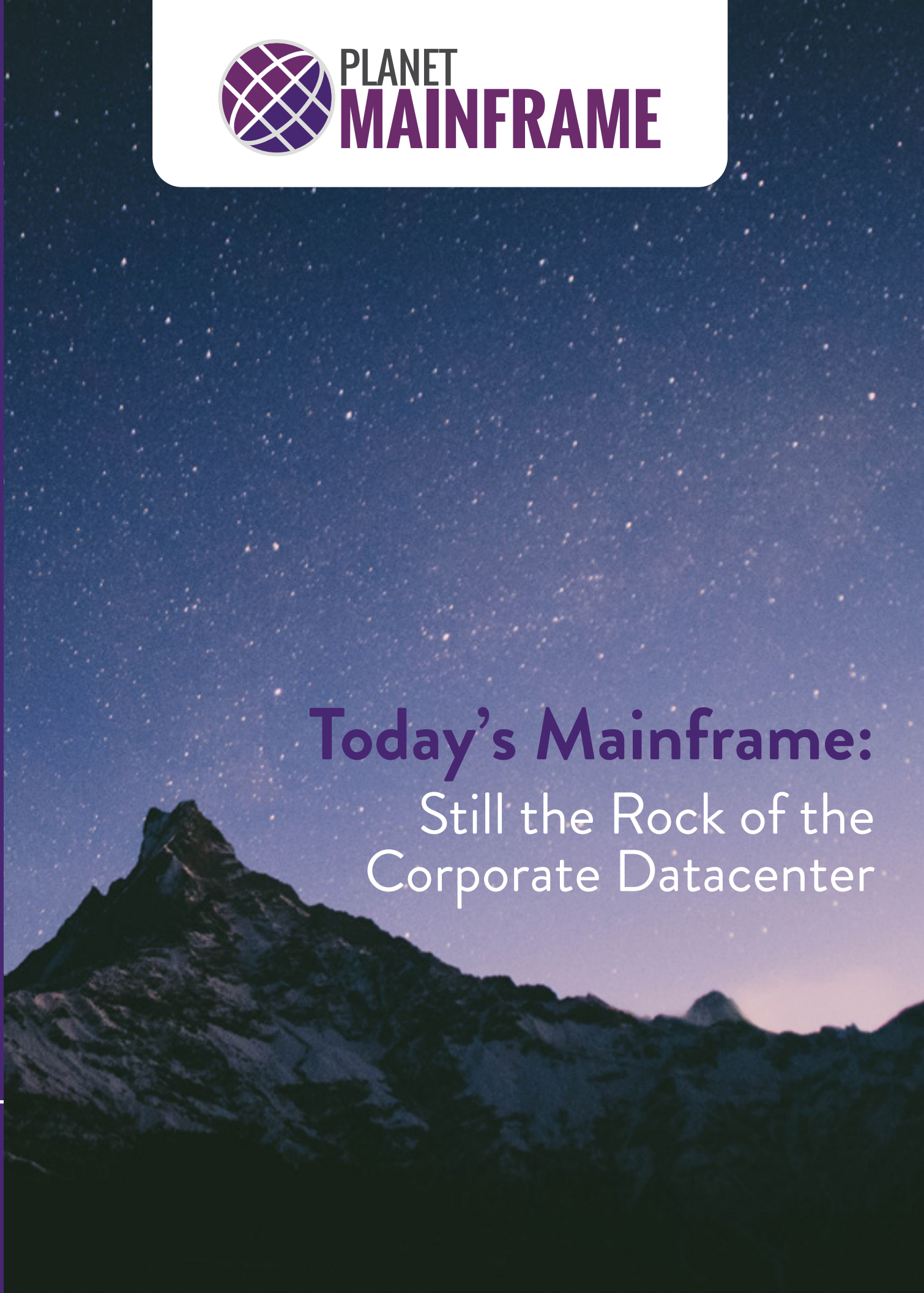
PLANET MAINFRAME



Today's Mainframe:

Still the Rock of the Corporate Datacenter

ISSUE | 01

Articles | Issue 01

Every IT Professional Should Work in a Mainframe Environment (...at some point) pg 4

Are Mainframers too Negative? pg 10

The IBM Mainframe: The most powerful and cost-effective computing platform for business pg 14

Mainframe Outsourcing Transparency pg 20

Has Mainframe Computing Inadvertently Started Its Own Culture War? pg 24

IBM Mainframes: the enterprise computing platform that led to the evolution of the modern "Cloud Computing Technology" pg 30

Published by Planet Mainframe
www.planetmainframe.com | info@planetmainframe.com

Managing Editor
Hanifa Anne Sekandi

Contributing Editor
Keith Allingham

Design
Andrew Brown

Contributing Writers
Craig Mullins
Trevor Eddolls
Allan Zander
Steven Thomas
David Herlich
Subhasish Sarkar

Submissions
Interested in writing for Planet Mainframe? Please contact us with your idea, relevant experience, and any other information that can provide additional details for your article.

Letter from the Editor

At Planet Mainframe, we aim to share and nurture the ideas of mainframe experts and enthusiasts. It is a vendor-neutral publication where differing opinions about 21st-century data centers and personal perspectives on new technologies are welcome. What is the future of mainframe? Also, what do you think is missing? Our contributors open the door to dialogue within the industry. We recognize the importance of sustaining this community not only for mainframe virtuosos whose much-valued experience is a continuum. We believe it is essential to entice those who are juniors within the field or considering it to continue the conversation.

What is the Planet Mainframe ethos? We are value driven. We respect the expertise of our contributors and the value that these perspectives encompass. We are your go-to source when a conversation is needed. Do you have something you would like to share? People are looking forward to reading what is on your mind. In this issue, Craig Mullins, President & Principal Consultant of Mullins Consulting, Inc. and IBM Champion & Gold Consultant, opens the door for a much-needed conversation in the article, *Every IT Professional Should Work in a Mainframe Environment (...at some point)*. What are the benefits of IT professionals working in a mainframe environment? Another great read, *Mainframe Outsourcing Transparency*, by Stephen Thomas, CTO at SMT Data, explores optimal practices and transparency between the customer and outsourcer.

Of course, we must keep the conversation robustly moving with an intriguing article by Allan Zander, CEO of DataKinetics, who provides his perspective on mainframe misconceptions in the article, *The IBM Mainframe: The most powerful and cost-effective computing platform for business*. The notion that mainframe is expensive and archaic is unfounded. This point of view is perhaps held by those unfamiliar with the cost-effective benefits mainframe can offer. Trevor Eddolls, CEO at iTech-Ed Ltd. and an IBM Champion, asks a question that may have crossed your mind, *Are mainframers too negative?* A great debate worth having. David Herlich, retired Senior Mainframe Software Engineer, rounds out this conversation poignantly in the article, *Has Mainframe Computing Inadvertently Started its Own Culture War?* Subhasish Sarkar, a Senior SQA Engineer, adds food for thought as he concisely expresses his view that the *IBM Mainframe computing platform is indeed the earliest cloud computing platform*.

What is on the horizon for Planet Mainframe? Our team will attend the SHARE, IDUG EMEA, and GSE expos. We look forward to continuing these great conversations in person. I am happy to dive into the great ideas you would like to share. As Managing Editor of Planet Mainframe, I hope to bridge the gap and entice new IT developers to see the value of mainframe at the same time inspiring a thought-provoking forum where seniors within the industry can share insightful outlooks. Sustainability of mainframe is reliant on mentorship and community, thus encouraging understanding. Mainframe is alive and well.

Hanifa Anne Sekandi
Planet Mainframe—Managing Editor

Every IT Professional Should Work in a Mainframe Environment

(…at some point)

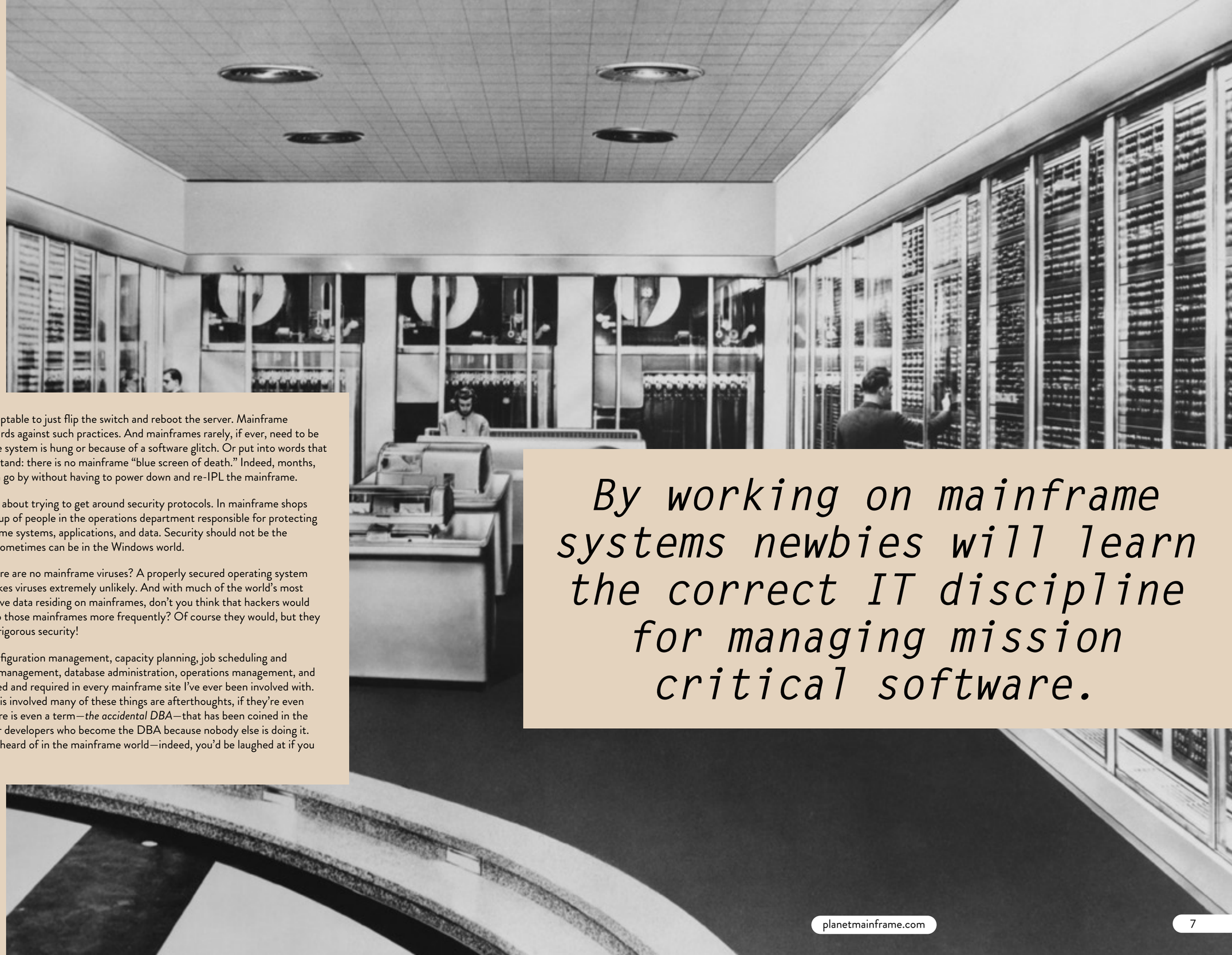
Craig Mullins

Talk to anyone who has ever worked on a mainframe and you will see that they are acutely aware of important factors that are sometimes overlooked on other platforms. Things like security, control, scalability, and reliability are second nature to mainframe computer systems and applications. Unfortunately, though, the bulk of new IT developer and programmers are not mainframe literate. This should change. But maybe not for the reasons you are thinking.

Yes, I am a mainframe bigot. I readily admit that. In my humble opinion there is no finer platform for mission critical software development than the good old mainframe. And that is why every new programmer should have to work a tour of duty on mainframe systems and applications at some point—preferably right after graduating from college.

Why would I recommend such a thing? Well, it is because of the robust system management processes and procedures which are in place and working extremely well within every mainframe shop in the world. This is simply not the case for Windows, Unix, and other platforms. By working on mainframe systems newbies will learn the correct IT discipline for managing mission critical software.

What do I mean by that? How about a couple of examples: It should not be an acceptable practice to just insert a CD and indiscriminately install software onto a production machine. Mainframe systems have well-documented and enforced change management procedures that need to be followed before any software is installed into a production environment.



Nor should it be acceptable to just flip the switch and reboot the server. Mainframe systems have safeguards against such practices. And mainframes rarely, if ever, need to be restarted because the system is hung or because of a software glitch. Or put into words that PC dudes can understand: there is no mainframe "blue screen of death." Indeed, months, sometimes years, can go by without having to power down and re-IPL the mainframe.

And don't even think about trying to get around security protocols. In mainframe shops there is an entire group of people in the operations department responsible for protecting and securing mainframe systems, applications, and data. Security should not be the afterthought that it sometimes can be in the Windows world.

Ever wonder why there are no mainframe viruses? A properly secured operating system and environment makes viruses extremely unlikely. And with much of the world's most important and sensitive data residing on mainframes, don't you think that hackers would just love to crack into those mainframes more frequently? Of course they would, but they can't because of the rigorous security!

Project planning, configuration management, capacity planning, job scheduling and automation, storage management, database administration, operations management, and so on—all are managed and required in every mainframe site I've ever been involved with. When no mainframe is involved many of these things are afterthoughts, if they're even thought of at all. There is even a term—*the accidental DBA*—that has been coined in the SQL Server world for developers who become the DBA because nobody else is doing it. Such a situation is unheard of in the mainframe world—indeed, you'd be laughed at if you even suggested it!

By working on mainframe systems newbies will learn the correct IT discipline for managing mission critical software.

Growing up in a PC world is a big part of the problem. Although there may be many things to snark about with regard to personal computers, one of the biggest is that they were never designed to be used the way that mainframes are used. Yet we call a sufficiently "pumped-up" PC a "server"—and then try to treat it like we treat mainframes. Oh, we may turn it on its side and tape a piece of paper on it bearing a phrase like **"Do Not Shut Off—This is the Production Server"**... but that is a far cry from the glass house that we've built to nourish and feed the mainframe environment.

Now to be fair, the infrastructure and best practices for managing distributed systems are improving. There are many shops that have begun to build better processes for controlling their non-mainframe computing environments. Indeed, some organizations have built an infrastructure around their distributed applications that rivals the mainframe glass house. But this is more the exception than the rule. With time, of course, the policies, practices, and procedures for managing distributed systems can improve to mainframe levels. But will they? That is hard to do when things are constantly changing with open source software, cloud computing, big data, and whatever the next heaping of hype turns out to be.

The bottom line is that today's distributed systems—that is, Linux, Unix, and Windows-based systems—typically do not deliver the stability, availability, security, or performance of mainframe systems. As such, a forced tour of duty supporting or developing applications for a mainframe would do every IT professional a whole world of good.

Are mainframers too negative?

Trevor Eddolls

Self-help gurus have become the equivalent to pop stars and TV idols. They are well-known and have thousands of followers on social media. And their message is always to be positive. We should always think about successful outcomes to anything that we want to achieve. And if we want those outcomes enough, then we will achieve them. It's a simple and compelling message, but is it the right message for people looking after a mainframe? Shouldn't all mainframers spend their day hoping that everything will work out successfully? Let's look at the evidence.

Suppose, for example, that I am a systems programmer working on a mainframe. I am making some changes to z/OS. And, while I enjoy a brief coffee break, I am visualizing my colleagues congratulating me on my work. I'm imagining the senior management at my organization coming over to my desk and shaking my hand because of my great work. Is that visualization likely to spur me on to doing great work? Or is it likely to reduce my motivation for doing great work? As strange as it seems, it's likely to reduce my drive to get the job done!

How do psychologists explain this? It seems that the brain can't tell the difference between real events and imagined events—after all, the information travels round the brain as electrical signals in neurons. So, what happens is this: you want to do something and set out to achieve your goal (ie update some SIT settings in CICS). When you've achieved the final result and everyone congratulates you, your drive to achieve the goal reduces to almost nothing—you've probably already begun planning your next project. However, when you visualize success, with its associated handshaking, that also reduces your drive to achieve your goal. Your brain thinks it has already achieved it, so you can stop now. And that can prevent you from actually achieving the goal in real life that you set yourself. The conclusion is to never visualize the success at the end, but only the steps on the journey.

Let's take another scenario. You have made some changes to the basic definitions of your z/OS system in parmlib. You're going to reboot on Sunday afternoon, and you hope that everything will work. Let's look at that key word, 'hope'. Again, we're looking on the bright side, being positive—isn't that what all the self-help gurus tell us?

Let's remind ourselves of the story of Pandora's box—the one that had all the curses of mankind (sickness, death, etc) in it. You remember that Pandora was curious about the box and let out all the troubles except hope. The problem is that the story we tell is quite different from the original told by the ancient Greeks. For example, the box was originally an urn or large jar. And 'hope', the Greek word 'elpis', is more often translated as 'expectation of evil' rather than 'expectation of good'. And that really doesn't make 'hope' a good thing, which is how we usually use the term.

But let's divide the world into two parts, the parts we can change (where we have agency—as they say) and the parts we can't change, like the weather (where we don't have agency). Hoping that it won't rain on Saturday is probably fine because there really is nothing we can do about it. Hoping that the mainframe will boot up and run perfectly is probably not OK, because we do have agency, if we are the systems programmer making the changes. We should be taking all the steps we can to make sure it works, and not just leaving it to chance that everything is going to be 'alright on the night'.

Let's take that example and go further with it. Let's suppose that we are all on a spaceship, miles out in the solar system. And, yes, I do like science fiction. I want the people at mission control to be considering the worse-case scenario of every action that we take on that spaceship. I also want them to consider what needs to be done now in order for that worst case not to occur. I want them to think whether an event will cause a fire, or a loss of oxygen, or life-support to fail, and make sure that appropriate steps are taken to prevent that outcome ever occurring.

In the mainframe world, this would be taking steps to ensure the operating system and subsystems were still running. To ensure that customers could still log in and buy product etc. It's like having a built-in health and safety officer making sure that whatever people do, no worst-case scenario will result. I guess the example in the news at the moment would be how we can prevent ransomware getting on our mainframe. Steps need to be taken now to ensure that it can't happen. And, in the event that it does, there need to be processes in place to prevent getting to the situation where backups are corrupted, files have been copied elsewhere and local copies encrypted, and a Bitcoin ransom message appears on the console.

In this case, not so much planning for the worse as planning the steps to avoid the worse happening is really a good idea. It's like working on the business continuity plan (BCP) to make sure that the company can stay in business should some kind of disaster occur.

The problem that many people find is that they have the opinion that thinking of bad things happening is their brain having negative thoughts. And, obviously, thinking that everything will turn out the way you want is having positive thoughts. So, it's not uncommon for mainframers and others to somehow feel guilty that they are not being positive. It's as if going against the wisdom of self-help gurus is wrong. But not all self-help or therapy techniques work like that. There's a thing called Acceptance and Commitment Therapy (ACT) that encourages people to embrace their thoughts and feelings rather than fighting them or feeling guilty about them. The approach was originated by Steven Hayes in 1982. The ACT model for people to follow is:

- Accept your thoughts and emotions
- Choose a valued direction
- Take action.

I mention this to show that mainframers don't need to worry about having negative thoughts. There is a talking therapy that helps people to accept those negative thoughts.

Almost lastly, I want to mention defensive pessimism. Defensive pessimism is a cognitive strategy that was first identified in the 1980s by Nancy Cantor, and it works like this. A person prepares for an event that causes them to be anxious (like installing new software or preparing against a security breach occurring) by using defensive pessimism. They do that by setting low expectations of their performance—ignoring how well they may have done in the past. They then think about all the setbacks etc that could happen to prevent things going successfully. Defensive pessimism allows them to avoid the pitfalls they imagined, or have strategies in place to deal with them. And the final outcome is usually success. Their anxiety has produced a positive outcome.

One final thing before we finish: don't think about a pink elephant! In numerous psychology experiments where people have been told not to think about something, they have failed. It's very hard not to think of something that you have just been told not to think of. Similarly with these self-help gurus: when they tell you not to think negatively, it's very hard not to.

When you're thinking about your mainframe and all the things that might happen, being negative is a good thing. It gives you the opportunity to prepare for worst-case scenarios, and be in a position to remediate them if they occur, and prevent them from occurring. Negativity keeps you focused and is probably what drives you to perform one more check that everything is OK before updating the system—rather than just hoping for the best! The truth is that mainframers can't be too negative.

The IBM Mainframe:

The most powerful and cost-effective computing platform for business

Allan Zander

Many of we mainframe pundits have written about the robustness, power, perseverance, capacity and more importantly, the cost-effectiveness of the mainframe (Allingham, Sun, Peleg), including myself. But what about showing the superiority of the mainframe using real numbers, comparing it to other platforms? That requires a lot more work. Schroder and Olders shows us some real-world numbers, but how about showing the ugly details? That's even more work, and fortunately, a couple of people have done that as well.

Michael Benson's *Enterprise Executive* article in 2016 did that—since then, distributed servers have come a long way (*AWS*, *Google* and a host of other cloud service providers), but so has the mainframe. In 2015, the top-of-the-line mainframe was the z13, an outstanding business machine; today the z16 outperforms it considerably on many levels —speed, transaction throughput, security, flexibility, and more. A main argument then, as now, is cost; and that's a losing argument right from the get-go.

Comparing Platform Costs

"Other platforms are cheaper..." This is the basic claim for most people interested in dumping mainframe systems in favor of commodity servers. Let's face it, *Google*, *Amazon* and *Microsoft* don't use mainframe systems at their back end, so why should anyone? That's a great point, but let's look at the premise first—are server farms less costly than the mainframe? Recently, Michael Benson did a study for *Enterprise Executive* magazine in an article called *CIOs: Are You Really Paying Less by Using x86 Platforms?* In it, he configured two similar performing platforms—one mainframe-based, using an IBM z13 mainframe system, and the other, a bank of HP servers. Table 1 shows the system specifications.

ATTRIBUTE	HP PROLIANT BL460 GEN9	IBM Z13 2964 N30
Total Servers	12	1
Processors	24	30
Cores/processor	12	1
Cores/server	24	30
Total cores	288	30
# VMs	1000	1000
Memory	2 TB	2 TB
Hypervisor	VMware vSphere 4	IBM z/VM
Cloud Mgmt	VMware vRealize	IBM Wave
OS	Red Hat Enterprise Linux	Red Hat Enterprise Linux
Web server	Apache HTTP	Apache HTTP
Application server	IBM Websphere	IBM Websphere
Messaging	MQ	MQ
Database	Oracle EE	Oracle EE

Table 1: Test Systems

He explains that running *Linux* on the mainframe is no different than running it on x86 servers. The only real difference is the cost, and the belief is that x86 platforms do it for less. But do they? The hardware costs for these configurations run in at $2,299,451 for the server farm solution, and $2,793,371 for the mainframe solution. However, due to licensing costs, the software cost for the server farms comes in at $1,807,406, with the mainframe running at only $416,883.

So yes, the hardware is cheaper, but there is not quite as much difference as you might expect. The real surprise is in the difference in software cost. When you also consider maintenance costs, the pattern is maintained. Maintenance costs for the server farm would come in at $390,327, with the mainframe at $269,767. Labor costs are also part of the picture.

ANNUAL LABOR	HP PROLIANT BL460 GEN9 (QUANTITY=12)	IBM Z13 2964 N30 (QUANTITY=1)
Server admin	$580,160	$430,000
Net admin	$384,000	$76,800
Total	$964,160	$506,800

Table 2: Labor cost

At the end of the day, what really matters is the ongoing operational costs of the two platform solutions. Table 3 shows a considerable gap in favor of mainframe computing.

OPEX	HP PROLIANT BL460 GEN9	IBM Z13 2964 N30
Hardware mtce	$9,544	n/a
Software mtce	$390,327	$269,767
Admin	$964,160	$505,800
Other (power, etc.)	$31,505	$68,355
Total	$1,395,536	$844,922

Table 3: Operating costs

Over a five year period, operating costs compound, and the picture looks much worse for the server farm, $9,052,749 vs $6,979,693 in for the mainframe setup. The shocking conclusion therefore, is that it is cheaper to run the mainframe system than it is to run the server farm.

When doing cost comparisons, it is good practice to look at all contributing costs, and to look at long term cost of ownership. This comparison would have looked a lot different if we stuck to just the hardware acquisition cost, or even if we hid the personnel costs in a general employee pool rather than in the TCO calculations.

Technology Economics

Cost is one thing—often a very misunderstood thing, as Michael Benson pointed out. But acquisition and ongoing cost represent only one dimension in a complicated cost-comparison between platforms. What about environments that run a mix of mainframe and distributed systems? And what about comparing not just cost between platforms, but real costs in specific industries? Well, that's where Dr. Howard A. Rubin of *Rubin Worldwide*, a technology economics research firm, comes in.

In his paper, *The Surprising Technology Economics of Mainframe vs. Distributed Servers*, Dr. Rubin explains that understanding computing platforms and their economic relevance in the context of their contributions to business performance is critical. This context provides a transparency that goes far beyond the basic economics of the costs of hardware and software acquisition or a TCO calculation. This is especially important when we consider that technology costs are a rising part of ongoing business operations expense.

IT costs vs business revenue and cost

Technology costs relative to business revenue and operating costs vary considerably from one industry vertical to another. For example, in banking and finance, IT expense represents about 6% of revenue and just over 7% of business operating expense; compared to the retail sector, where IT expense represents just under 1.5% of revenue and just over 1.5% of business operating expense.

Cost of platform choice

Businesses have choices on how they will handle their processing needs—and this typically comes down to the mainframe and server farms. The cloud is part of the latter solution. The reality is that any business that runs mainframe systems also runs server farms, so it is fair to characterize them as running "mainframe-heavy" datacenters, while those without mainframe run "server-heavy" datacenters. It is also useful to consider new metrics for these datacenters—MIPS per $1M of revenue and physical servers per $1M of revenue. These aren't equivalent in any way, but they serve to represent the economics of their computing choices in measurable economic terms.

When comparing businesses within the same industry vertical, the "heaviness" of their IT deployment strategies result in a significant differences. For example, for financial services businesses:

Mainframe-heavy shops consume:

- 3.1 MIPS per $1M of revenue
- 0.22 servers per $1M of revenue

While the server-heavy shops consume:

- 1.75 MIPS per $1M of revenue
- 1.2 servers per $1M of revenue

When these figures are mapped to the total cost of mainframe and server farm costs within various industry verticals, the economic differences that can be attributed to their deployment strategies become apparent—see Table 4. The inescapable conclusion is that mainframe-heavy computational costs to support a $1B organization on average may be 30% lower than a server-heavy deployment.

BUSINESS	AVERAGE COST	MAINFRAME-HEAVY COST	SERVER-HEAVY COST
Distribution	$4,255,273	$3,936,728	$6,809,818
Communications	$4,979,371	$4,306,000	$8,295,000
General business	$4,832,000	$4,414,000	$7,846,000
Computer Services	$6.093,958	$5,644,350	$7,619,000
Industrial	$9,270,513	$9,082,000	$11,181,000
Financial Services	$12,627,002	$12,742,000	$16,445,000
Government	$15,161,129	$14,148,000	$15,981,703
Average	**$8,174,178**	**$7,753,297**	**$10,596,789**

Table 4: Mainframe & Server cost to support a Billion Dollar Business

Cost of Goods

While the cost of technology yields interesting conclusions, the actual costs of platform choice are also surprising, and support the former. The next step is to link the technology costs to business costs.

A good way to do that is to use a cost-of-goods metric. Ask the question, "what is the IT cost contribution to the business cost of goods?" And follow that up with, "how does technology deployment affect the measure of impact on the business?" Table 5 itemizes the cost of goods for five business types—finance, industrial, communications, general business and insurance.

PROCESSING COST PER:	AVERAGE COST	MAINFRAME-HEAVY COST	SERVER-HEAVY COST	RATIO, MF VS SERVER
ATM	$928.00	$572.00	$1,021.00	56%
SKU	$227.27	$184.09	$252.27	73%
Mobile subscriber	$23.26	$18.26	$26.12	70%
Patent	$390.83	$372.00	$401.00	93%
Claim	$78.00	$56.00	$92.00	61%
			Average	71%

Table 5: IT Cost of Goods: Processing Costs per Transaction

This data implies that where appropriate, effective use of mainframe resources results in a 29% cost advantage over distributed server-heavy deployments.

Looking closely at the insurance data, we see that the average IT cost of processing an insurance claim in a mainframe-heavy environment is approximately $56, which is $36 less than the processing cost in a server-heavy environment. What does that mean to an insurance business? For an insurer that processes 100,000 claims per year, the savings could be $3.6 million per year by leveraging mainframe technology.

Similarly, a bank with 4500 ATMs would be paying over $1000 per ATM using a server-heavy datacenter, as compared to less than $600 using the mainframe-heavy scenario. Such a bank could save more than $2 million per year by leveraging mainframe technology. Of course, ATM costs are only one small part of a bank's IT cost concerns.

Competitive advantage

Any large company interested in maximizing computing power AND controlling cost will clearly enjoy a competitive advantage over a similar company that just seeks to avoid mainframe technology in favor of server farms. This advantage translates directly to the bottom line, shareholders and investors. And for a company considering a mainframe migration project as a means for cutting costs, this information could be seen as "found money."

Conclusions

The facts support the notion that the mainframe is the most powerful and cost-effective computing platform for large businesses with a need for high-intensity transaction processing. Claims to the contrary are typically either as a result of simple lack of knowledge on the subject, or a biased unwillingness to look objectively at the facts.

But if the mainframe is so great, then why is it not being used by the newest and latest concerns (*Amazon*, *eBay*, etc.)? The reason is bias. Whether intentional or through ignorance, there is a great deal of bias against the mainframe. It's too expensive! (It clearly is not.) It's old and dusty! (Obviously not.) It's hopelessly outdated! (Not so.) I don't know very much about it! (Maybe that's it...)

The last part is the key to the puzzle of why the mainframe generally has a difficult time displacing server farms in environments where it could make a positive impact. The truth is, organizations that could benefit from the mainframe, but don't, are leaving money on the table.

Where to go from here

So, what about folks who are having trouble keeping up with growing workloads on their **"most powerful and cost-effective"** mainframes? Should they be upgrading? Shifting workloads off-platform? As you might guess, there are options. There are a couple of organizations that are helping mainframe shops to optimize what they have now—to increase workload throughput of the systems they're currently running. No upgrade needed; no changes to application logic, no changes to the Db/2 (or whatever) database being used. This is possible using high-performance in-memory technology. And both *IBM* and *DataKinetics* are offering these solutions right now.

Mainframe Outsourcing Transparency

Steven Thomas

Outsourced mainframe customers often lack a clear understanding of how well their outsourcer is managing the mainframe capacity and performance relative to what is optimal for the customer. The customer seldom has access to the data needed to really understand their capacity usage and often lack the skills to communicate clearly with the outsourcer on capacity and performance questions. The outsourcer may have limited motivation to help the customer optimize capacity costs. In many cases, the customer doesn't even have the ability to validate whether the outsourcer is invoicing in a correct manner relative to the contract or in a fair manner relative to industry 'best practice'. Simple questions like 'how is the basis for invoicing (e.g. MIPS) calculated?' can lead to a 10% difference in the invoice.

Many companies outsource their mainframe operations. And billing models vary—from customers who pay for a fixed capacity based on capping, to customers who have more complex models such as paying for the 90th percentile peak MIPS usage between 8:00 am and 5:00 pm weekdays. Often the actual contracts are negotiated by people on both sides with limited insight into the technical nuances. Once the contract is signed, other people take over the actual measurement and billing, and we often see a disconnect between what was agreed and what is invoiced. This could be because the agreement leaves room for interpretation – for example how MIPS are calculated. It could also be because the outsourcer's configuration changes in ways that are not foreseen in the contract. E.g. if the outsourcer adds zIIP engines to the mainframe, this will add an overhead that increases the MIPS utilization of all users. Is that effect regulated in the contract?

Some of the important questions that a customer and outsourcer should have a common understanding of include:

- Does the capacity usage (for example in CPU seconds) on the invoice match the measured usage as seen in the SMF Data?
- Has the outsourcer used the agreed methods for calculating the capacity for which the customer is paying (for example MIPS calculated from CPU seconds)?
- Are the agreed methods for invoicing capacity in line with industry 'best practice'? For example, is the MP overhead of additional processors fairly reflected in the MIPS rate used to bill the customer?
- What are the advantages and disadvantages of other methods of determining capacity usage for the customer and the outsourcer?
- How can the billing model contribute to an alignment of interests between the outsourcer and the customer? For example, if the outsourcer's costs are driven by peak monthly MSU usage, then how can both parties be motivated to move workload away from that peak?
- What are the cost drivers, where is there optimization potential and what specific actions can the customer or outsourcer take to achieve savings?
- Are there capacity related performance issues such as poor response time, and what options are there for addressing these besides increasing capacity?
- What are the ongoing reporting requirements to ensure a transparency between the outsourcer and the customer?

Answering these questions requires access to data and knowledge of the technical aspects of the agreements between the customer and the outsourcer.

The most relevant data is typically SMF 70, which gives the resource usage at the LPAR level, and is what invoicing is typically based on. Other SMF data such as SMF 72, 30, 101, 110 will allow a deeper understanding of the actual cost drivers seen from workloads, jobs, Db2 and CICS respectively. This kind of data is essential for understanding how to reduce cost or optimize performance.

The prices or commercial terms from the contracts are not actually necessary to start a transparency discussion. What is interesting is the technical definitions. E.g. Is the number of CPU seconds on the SMF 70 record used as a basis for the billing? How is the usage normalized across different machine configurations? E.g. how are MIPS calculated from CPU seconds? How are peaks determined? Is there a difference in the way CP and zIIP MIPS are billed?

Opening these discussions can be a delicate exercise between the customer and the outsourcer. To avoid contention, it is important to be fact based. What is measured? How are the calculations done? What is billed? What can be done to reduce the bill? Clear and transparent answers to these questions are the basis for a positive working relationship between outsourcer and customer.

Has Mainframe Computing Inadvertently Started Its Own Culture War?

David Herlich

I have spent my career in the mainframe computing space. It's a strange and insular world. And it's shrinking a little each day, as the global mainframe workforce sees skilled workers retiring, transitioning to other roles, sometimes even dying, without a sustainable pipeline of mainframe-knowledgeable employees to take their places.

Before starting my current position at a private mainframe software vendor, I worked as a developer for one of the world's largest mainframe software providers, and later at the IT headquarters of one of the world's largest mainframe software users. Throughout my time at each, I kept hearing a familiar refrain.

The mainframe is dead! Long live the mainframe!

The story always begins the same way. Big users of Big Iron desperately want to get off the mainframe platform, and once they do, it's bye-bye Mainframe, and all remnants, real or imagined, of giant rooms of whirring tape reels, blinking lights, and teams of men with 1950s haircuts will disappear forever.

Obviously, that hasn't happened. The plot is always foiled, playing out something like this.

A mainframe shop undergoes some senior management changes. The new boss, invariably a too-young, MBA-wielding bean counter with no clue what he's doing, takes a look at the monthly mainframe bill. He recoils into mid-air like a cartoon character, red-faced and suspended in position for a few seconds while smoke comes out of his ears. He then falls back into his executive desk chair and launches an initiative to migrate his company's operations off of the mainframe.

His underlings earnestly carry that message forward, where it is met with scorn and laughter by the mainframe technical team, a group which, as the story goes, is somehow always made up of older, mostly male employees, crusty and bawdy as they come in a professional workplace. They say they've been through all this before, and insist, with a common voice and hacking cough, that the bean-counter is a fool who will eventually come to his senses, just as everyone before him has.

Fast forward about two-and-a-half years, and the migration project has been scrapped, the boss with the MBA has moved on to become CFO at another company, and on the mainframe team, one member has retired, another was laid off, and a third (the youngest and most promising) has transferred to a "safer" area of the company.

Five years later, repeat.

If you are a mainframe-centric person, you know this story well, and can recite the talking points:

- The mainframe platform is not more expensive when all costs are considered.
- The transaction throughput of the mainframe cannot be matched.
- Reliability, availability, scalability, data integrity, and security on par with the mainframe is not available elsewhere at any price.

But mainframe folks also hear the jeers.

- It's prehistoric technology.
- It's uncool.
- It's impossible to learn and use, and no one wants to try.
- Nobody cares.

In response to these criticisms, the mainframe community closes ranks and demonizes those who dare challenge all that is holy. **"They don't understand,"** the old-timers lament, before branching into an unhelpful tirade about how an assembler programmer can code circles around some Johnny-come-lately, Python punk.

I don't believe we devote enough resources to front-ends, dashboards, or even documentation which might help someone who is willing to venture into our world. Once the pioneers, we now often act as members of an elite and entitled club, refusing entry to anyone without the secret handshake and a detailed understanding of first level interrupt handlers.

Yes, it's a strange and insular world, and even though we say we want and need new blood, people have taken up sides in what increasingly looks like a culture war between the mainframers and the anti-mainframers. Some stakeholders embrace this battle with the partisanship of present-day American politics, with equally unproductive results.

The mainframe has been written off many times, only to emerge time and again as the platform of choice for the world's largest and most consequential enterprises.

The mainframe has been written off many times, only to emerge time and again as the platform of choice for the world's largest and most consequential enterprises. In this regard, those crusty old men with their crusty old **"I hate Mondays"** coffee mugs are right. But the younger generation of programmers are also right. We are keeping them out of the club while simultaneously making it a place where nobody wants to come in anyway. What is the appeal? With all the threats to the mainframe's existence seen to date, I fear that what could ultimately lead to its permanent demise is that there simply won't be enough people who know how to work on it.

A former colleague of mine, an excellent mainframe systems programmer, once proclaimed in a meeting that "it takes a special kind of person to do this job." Was that a boast or a dare? Either way, it's not music to the ears of anyone hoping to widen the employee pipeline.

Efforts to recruit people to the field, or to insert mainframe courses back into universities, typically involve trying to make the case that mainframes are cool because of what they can do and how important they are.

This approach isn't fooling anyone. It's great messaging only for those already on board. Of course it's true. But once again invoking the analogy of politics, the truth hardly matters. To newly minted programmers with plenty of options, the mainframe path seems both unexciting and too short to sustain a career-long journey. One peek around a bend and they see surprisingly lackluster pay and old men with complaints, instead of stock options and young people with skateboards. Most importantly, they see themselves as having to declare a partisan choice. A binary decision with a big wall separating the two sides. When a new graduate's friends all go running to the beach party, it's hard to coax him or her over to the side of green screens and tales of punch cards.

As with politics, we must remember that we are on the same side and that we all want basically the same things. So, before I retire, get laid off, or die myself, here are a few suggestions to help replenish the mainframe workforce and save the world.

Let's try to portray the mainframe as within the same sphere as any other field of computing. Let's make the choice between C#, Javascript, and COBOL look like an equilateral triangle.

Let's show how the client-server side of a business works in conjunction with the mainframe side, not separately from it or against it. And show the synergy the web and social-media departments have with both. There is already a lot of common ground. Metal-C, Mainframe-based Linux, AI in all flavors. IoT needs both mobile devices and mainframes to operate.

Instead of apologizing for areas of overlap or highlighting them as potentially desirable exceptions, we should champion the wide open, undivided playing field that is modern computing, until such thinking becomes the status quo.

In all its forms, we must tear down this wall.

IBM Mainframes:

the enterprise computing platform that led to the evolution of the modern "Cloud Computing Technology"

Subhasish Sarkar

As strange as it may sound to many, but people need to get their fact straight—that IBM Mainframes were the first ever Cloud Computing platform built on planet earth. Cloud Computing is probably the most heard about '**Fancy Tech Term**' in the modern era of Digital Computing. However, most people probably are unaware of the fact that for decades, the IBM Mainframes have been providing organizations with all the benefits that the modern cloud does today. Believe me or not, IBM Mainframes are the first Cloud Computing Platform on earth!

Finding it hard to believe me? Let's go out and check the facts for ourselves then. Below I specify the various cloud computing benefits and how the IBM Mainframes have already been providing those benefits to enterprises for decades now.

Scalability

Scalability is a key feature of modern cloud computing. Scalability simply means that you have the flexibility and elasticity to quickly provision resources in the cloud as you need those, and then de-provision/remove the resources when you don't need them. This sounds to be such a cost-effective feature, right?

And, guess what? The IBM Mainframes were built to scale from their very inception. The IBM Mainframe provides massive scalability with it's astonishing computation and processing power. Just to provide you with an indication of the computing power of the IBM Mainframe, it can process around 2.5 to 3 billion transactions a day. IBM Mainframes were designed to support hundreds and thousands of users and millions of concurrent transactions. Batch Processing and Message Queue Processing technologies which have been a part of the IBM Mainframes for decades now, lend themselves to asynchronous computational activities and efficiently and fully utilize the CPU capacity. That the IBM Mainframe is a real beast when it comes to a seemingly unlimited computation and processing power speaks volumes about how scalable a platform the IBM Mainframe has been from the day it appeared on the commercial marketplace.

Virtualization

In very simple terms, virtualization denotes the capability of software simulating hardware and is another key feature of the modern-day cloud computing. With the advent of companies like VMWare, the virtualization technology started hogging the limelight in the mid-2000s. So, in that sense, virtualization is a recent technology in the history of computing when the technology enabled putting a layer of virtual machines on a commodity PC.

However, virtualization technology has been a part and parcel of the IBM Mainframe computing platform since as old as the mid-1960s. Creating logical hardware partitions called LPARs (logical partitions) has been in existence since decades. Virtual memory came into existence in 1965 and Machine virtualization in 1972.

The ability to take advantage of virtualization on the IBM Mainframe enables you to be able to run zLinux on your mainframe while also parallelly running z/OS.

Multi-Tenancy Model

What is a multi-tenancy model and how does it work? Let us consider a very simple example. Consider an apartment building where, say, 100 people, stay. Each one shares the common apartment building infrastructure, but each has his/her own apartment within that apartment building to live in. And, each share his/her own privacy in the respective apartment.
When you apply this analogy to cloud computing, multiple users share the same physical infrastructure and/or applications but also enjoy complete privacy and security over their own information. Cloud computing resources are designed to support the multi-tenancy model.

On the IBM Mainframe, LPARs have been doing just that for decades now. LPARs help create distinct computing environments such as separate environments for development, testing and production. Resource allocation and security access control can be set up for a given environment based on the need. As an example, you can build your production environment to be much more secure and get it an allocation of more resources (more memory allocation, for example) and probably, a higher execution priority as well.

Resource Pooling

Being able to service multiple users from the same physical pool of resources is what is called Resource Pooling. A cloud provider should have a very large and flexible pool of resources to be able to cater to servicing multiple client requirements, ensuring scalability and that resource allocation should, in no way, impact the performances of critical applications. The multiple users that are drawing resources from the same physical resource pool need not have any knowledge of the pool.

On an IBM Mainframe Computer, several different applications execute simultaneously, and each executing work expects consistent execution times and predictable access to databases. On z/OS, the Workload Manager (WLM) component fulfills all these needs by controlling the executing work's access to the different system resources, based on external specifications/goals set up by the System Administrator. The critical mainframe resources are thus mediated by the WLM and other parts of z/OS, ensuring high system performance. The end users don't even need to know about these mediations or how they are performed.

Measured Service

The cost model that the cloud service provider charges back to the user is a "pay as you use" model. The user will obviously pay a variable amount to the cloud service provider based on the user's actual consumption. So, that effectively means that the cloud service provider monitors, measures and reports back to the users their cloud resource usage.

On the IBM Mainframe computing platform, SMF (System Management Facility) can provide a metering capability, which enables a dynamic charge back model. SMF and other repositories provide the opportunity to identify resource utilization on the platform and cap usage or resources that is in line with the user's agreement.

Summary

Now that we have looked at the various cloud computing benefits and compared to see how the IBM Mainframe computing platform has already been, for decades, providing those benefits to the customers, are you pretty much convinced by the fact that the IBM Mainframe computing platform is, in fact, the earliest cloud computing platform to have existed? I am confident that you should be, by now!

My intention here is to not discredit the Cloud Computing technology in any form whatsoever. In fact, I have always believed that we should open our arms to embrace any new form of technology that proves to be beneficial to the entire ecosystem and so is the case with the Cloud Computing technology as well. However, the point that I have tried to establish through this article of mine is that we should not try to look at the IBM Mainframe computing platform to be the one that is trying to keep pace with what the world today calls the latest and greatest "Cloud Computing Technology" but that there always had been a computing platform in IBM Mainframe that has been effectively carrying out, for decades now, all the tasks that the modern "Cloud Computing Technology" is supposed to do.

I would rather like to call the modern "Cloud Computing Technology" to have evolved from its predecessor, the IBM Mainframe Computing Platform. The IBM Mainframe Computing Platform and the modern "Cloud Computing Technology" can and should coexist.

PLANET MAINFRAME

Your Mainframe Community

For Advertising Opportunities please contact us at:

info@planetmainframe.com
615.437.3070

2022