

Don't be afraid of the mainframe

A demystifying guide



**The layman's introduction to
IBM mainframes**

Niek de Greef

Table of contents

Introduction	11
Reading Guide	13
1 What is a mainframe?	14
The mainframe: a definition	14
A very brief history	14
2 Understanding mainframe cost	16
An overview of the cost of a mainframe	16
The cost of managing a mainframe configuration.....	16
Who do you need to run a mainframe?	16
The benefits of centralized management.....	17
The cost of mainframe software	17
Understanding the cost of software on z/OS, MLC, and OTC	17
Clearing up MSU smoke and mirrors	19
The cost of hardware for the mainframe	21
Mainframe server costs	22
Mainframe storage costs	24
The cost of Mainframe connectivity devices.....	24
Special topics related to mainframe costs	24
New license models and their limitations.....	24
The bill for your current software usage cannot be changed unless.....	25
Super-expensive suppliers.....	26
Challenges with traditional license models.....	26
Why mainframe hardware is so expensive, and why not.....	27
3 The big server box and other hardware.....	30
The box, a big box	30
A box full of CPU and memory	30

What else is in the box?	31
Peripherals and other quirks	32
Big hardware, but partitioned in smaller parts	33
Specialty engines.....	34
4 Mainframe operating systems.....	37
z/OS	37
z/VM, the mother of all hypervisors.....	37
z/VSE.....	37
z/TPF.....	38
Linux on IBM Z	38
What's next in this book.....	38
5 z/OS – the mainframe flagship operating system	39
The MVS part of z/OS	39
Basic operation of the MVS part with batch and JCL.....	39
Address Spaces are processes	41
Side note: Virtual storage is memory.....	41
Datasets, files, and catalogs.....	42
Datasets are record-oriented and can have different structures.....	42
The EBCDIC character set.....	43
Catalogs are like directories	44
The Unix parts of z/OS.....	45
z/OS Unix.....	45
z/OS Container Extensions	46
All Unix variants	47
Parallel Sysplex.....	47
A cluster of z/OS instances.....	47
Special sysplex components: the Coupling Facility.....	48
Middleware exploits the sysplex functions	49
How is this different from other clustering technologies?	50

GDPS	50
The interface to z/OS and the green-screen myth	51
Green screens are for administrators and programmers, not end-users.....	51
TSO	52
ISPF	52
SDSF – or equivalent.....	53
Modern tools for development and operations.....	53
6 Middleware for z/OS.....	56
Application Servers.....	56
Applications Servers and Transaction Managers intermezzo.....	56
WebSphere Application Server	57
CICS.....	57
IMS.....	58
IDMS	58
ADABAS and NATURAL	58
IDEAL and Datacom	58
Database management systems.....	58
Db2	58
IDMS/DB.....	59
IMS/DB	59
Datacom/DB.....	59
ADABAS.....	59
Special topic: IBM Db2 Analytics Accelerator (IDAA).....	59
Integration software.....	60
WebSphere MQ	60
IBM Integration Bus.....	61
Batch processes and the Job Scheduler	61
7 Programming languages for z/OS.....	62
COBOL.....	62

PL/I.....	62
Assembler.....	62
The operational risk of assembler programs	62
Fortran.....	63
Java.....	63
C/C++	64
JCL.....	64
Rexx	65
Unix shell script	65
SAS.....	65
Easytrieve.....	66
Python.....	66
Other languages.....	66
8 Integration with the rest of the world	67
File interfaces	67
Network File System.....	67
FTP	67
Managed File Transfer	67
Message queueing.....	68
Web services (SOAP, REST).....	69
Enterprise Service Bus.....	69
Adapters.....	70
CICS Transaction Gateway.....	71
IMS Connect.....	71
z/OS Connect.....	71
Screen scraping tools.....	72
Legacy integration suites.....	73
Database access via JDBC, ODBC.....	73
9 DevOps tools for z/OS.....	75

Traditional DevOps process for development	75
The traditional waterfall is a staged process.....	75
Deployments are incremental - the concept of a build does not exist	75
Problems with the waterfall model.....	76
Modern development processes for the mainframe	77
The modern Source Code Management tools for z/OS	78
Build automation.....	78
Deployment automation	78
Integration in other pipelines.....	79
Implications for mainframe development.....	79
Infrastructure provisioning	79
Full deployments versus delta deployments?.....	80
10 Security	81
Centralized security management.....	81
The SAF interface	81
Security products.....	81
IBM Enterprise Key Management Foundation	82
Cryptographic facilities on the mainframe.....	82
Traditional encryption.....	82
Pervasive encryption.....	82
Data Privacy Passports	83
Multifactor Authentication.....	83
11 System management	85
System operations	85
Operator interfaces and some history.....	85
Automated operations	87
Monitoring	88
Infrastructure monitoring.....	88
Application monitoring.....	89

	Business Monitoring.....	90
12	Modern application architectures	91
	Main characteristics and principles	91
	Architecture principles	91
	Position of mainframe applications in modern application landscapes	92
	The main characteristics of a z/OS application	93
13	Legacy.....	95
	A problem for the organization.....	95
	Technical debt	96
	Application technical debt.....	96
	Infrastructure technical debt	97
	Technical debt is neglect of maintenance.....	98
	What to do if you want to decommission the mainframe	98
	Outsourcing infrastructure management	98
	Rehost the platform	99
	Retire all applications on the mainframe platform.....	100
	Replace through repurchase	100
	Replace through refactoring.....	100
	Do nothing.....	100
	Investing in modernization.....	101
	Can't we move to the Cloud?	101
	A program to get a grip on your mainframe platform	102
	Cost of value of the mainframe platform and its applications.....	102
	Position of the mainframe landscape in your application portfolio	102
	Mainframe application portfolio lifecycle and strategic fit.....	103
	Technical vitality of your mainframe environment.....	103
	The operational effectiveness of DevOps and infra teams.....	103
	Cloud strategy and mainframe alignment	103
	The way forward	103

	Addressing the cultural aspect.....	104
14	Linux for the mainframe	105
	Linux on IBM Z = ZLinux	105
	What does a Linux for Z configuration look like	105
	What is Linux on Z for?	106
	Linux in z/OS Container Extensions.....	107
	What is z/OS Container Extensions.....	107
	What is it for?	108
	In summary: Linux on Z.....	108
15	Towards Cloud?	110
	Standardization and automation.....	110
	The problem with z/OS application environments: no (industry) standardization.....	110
	The approach to cloud for z/OS.....	111
	A stepwise approach to standardization.....	111
	A financial model for Cloud and mainframe	112
	Conclusion.....	112
16	“About the Author”	113
17	Bibliography	114

Introduction

This book is an introduction to IBM mainframes. Mainframe computing is different and often not well understood. This book aims to describe IBM mainframe technology in easily understandable terms. The reader is expected to have no more than a basic understanding of computers. My aim with this book is to clear the smoke and mirrors around mainframes and the use of mainframe technology.

In this book, I will not dive deep into technology. However, technical descriptions are necessary here and there to explain certain concepts. It is okay if you skip the parts that go too deep.

The book's title is Don't Be Afraid of The Mainframe because that is what I was aiming to achieve in writing this book: to make the reader more comfortable with IBM mainframes. In my experience, we, as mainframers, are often very good at scaring non-mainframers with weird technical jargon. Or, to put it more lightly, we are not very good at explaining mainframe things in layperson's terms. The mainframe is as simple (or complex) as any other computing tool. Perhaps it is even simpler.

I will discuss technical aspects around IBM mainframes and commercial and financial considerations. These aspects are related to each other, as I will explain. Some basic concepts must be understood even for a person only interested in the financial aspects of IBM mainframe technology. I will do my best in this book to address this challenge.

I hope you enjoy reading the book. All feedback is welcomed. Please send me an email at: ...

Niek

1 What is a mainframe?

Before anything else, let's first define what a mainframe is. After that, I will give a very brief historical perspective on mainframes.

The mainframe: a definition

I will make up my definition based on definitions from several other resources.

A mainframe is a large central computer designed to run many different computing tasks simultaneously in a secure manner.

In this definition, I use the term computer for all the hardware and software components needed to run a meaningful application software.

There are a few critical aspects in this definition of a mainframe.

Central. The design point of the mainframe is to provide a centralized function for many computing tasks. The mainframe shares computing resources, hardware, and software between applications. The other way around, many applications use the same mainframe computer.

Many tasks at the same time. From the ground up, the mainframe operating system design is meant to run many technical tasks and applications very efficiently. The operating system is the core software on a computer on which application software can be executed.

Secure. Because many applications run on the same computing platform, the mainframe's operating system has very sophisticated facilities to ensure that applications can not interfere with each other.

A very brief history

Mainframes stem from a time when computing equipment was costly, and a single central solution for computing was the only way to run computing tasks economically.

Many large organizations use mainframes. You typically find mainframes in banks, insurance companies, government institutions, large supermarkets, and many smaller organizations. These organizations have often used the mainframe since the 1960s or 1970s. Applications that were developed then are often still in use today.

Typical programming languages used to build business applications for the mainframe are COBOL and PL/I, although Java has also become a very popular language on the mainframe.

Other modern technologies and middleware are nowadays also available on the mainframe. You can run state-of-the-art application servers, integration tools, and data analytics solutions and even run Artificial Intelligence applications with special AI facilities on a mainframe.

A mainframe is broadly appreciated for its ability to process and manage large amounts of data very robustly and securely.

Many companies in the past 70 years have built mainframe computers. Examples are ICL, Bull, Sperry, Tandem, Siemens, Unisys, and NEC. However, IBM has been by far the most successful mainframe manufacturer, and IBM mainframes have become synonymous with the term 'mainframe.' In this book, I will focus on the IBM mainframe solely.

2 Understanding mainframe cost

Let's first talk about money. The main thing that scares mainframe users is the bill they get from their software and hardware suppliers. This chapter will describe the most important costs of running a mainframe. Then, I will explore the good, the bad, and the ugly of mainframe costs.

An overview of the cost of a mainframe

Mainframe costs fall into three categories: hardware, software, and management.

- Management costs are the costs to install, maintain, and operate the hardware and software in a mainframe configuration.
- Hardware costs include the acquisition and maintenance costs of the mainframe server machines(s), the storage devices, and all the other equipment you need.
- Software costs are the licenses paid for the required software components.

The most significant cost component of a mainframe setup is the software. I estimate unscientifically, but from experience, that two-thirds of the total mainframe costs are software costs. Hardware and management costs are half of the remaining costs, so each is about one-sixth of the entire sum.

The cost of managing a mainframe configuration

The costs of managing the mainframe are people costs. I will describe what kind of people you need and elaborate on the associated costs for these roles.

Who do you need to run a mainframe?

To run a mainframe computer requires a few roles. Roles will be combined into one or more persons for small mainframe shops. In larger mainframe setups, you will find that roles are even more specialized. The terminology I have used here is mine. Role names will differ across organizations, but the responsibilities are the same everywhere.

- Engineering. The engineering team installs and configures mainframe hardware, operating system, and middleware software components.
- Operations. A centralized operation team ensures the daily operation and functioning of the central components of the mainframe configuration.
- Application management. The application management team manages the daily functioning of one or more applications running on the mainframe. They perform this task as part of a business process that the application serves. In a DevOps organization, these teams perform the Ops side of a DevOps team.

5 z/OS – the mainframe flagship operating system

This chapter will introduce you to the main z/OS concepts and terminology. The goal is to give you a good idea about the main z/OS concepts. I will also explain how z/OS peculiarities relate to more commonly known concepts in Unix and Windows operating systems.

I will discuss z/OS from the traditional ‘MVS’ side and the more relatable Unix side. The traditional MVS side stems from the 1960s and deviates most from what we know from Windows and Unix. The second part is z/OS Unix, an extension added in the 1990s, and can run Unix applications, very much like other Unix flavors.

In the section following I will talk about the unique clustering facility that z/OS brings, called parallel sysplex.

Finally, I will look at the green screen user interface that the mainframe is often associated with. I will discuss where the tools with green screen are still necessary and describe the modern tools for z/OS, tools with more sophisticated graphical user interfaces.

The MVS part of z/OS

The part of z/OS that I refer to as the MVS side, is the traditional ‘old-school’ mainframe side. The MVS part is the subset of functions rooted in the 1960s. These functions were built in an era in which punch cards contained the program code. You fed a stack of punch cards into a punch card reader connected to the computer to get something done from the computer. In the MVS part of z/OS we still find the remains of these features. Today these look a bit awkward.

Basic operation of the MVS part with batch and JCL

First, I will look at the basic operation of a batch process in z/OS. Batch processing is the style of computing with which you prepare input data to be processed, start the compute process, wait a while, and receive the output data as a result. There is no interactivity in a batch process.

The batch process is a core concept in the MVS part of z/OS. In essence, today’s batch processing still works in the same way as it was when designed in the 1960.

To run a program on z/OS, you need a mechanism to tell z/OS what to do. You do this by creating a script. The language for this z/OS script is called JCL – Job Control Language.

With the JCL script you tell the computer which program to run, what the input is for that program and where to write the output. The JCL code looks like this:

```
//RUNPROG EXEC PGM=PROGRAM1
//INPUT DD DISP=SHR,DSN=MY.INPUT.DATA
//OUTPUT DD DISP=NEW,MY.OUTPUT.DATA
```

This code looks awful of course, despite its simplicity. The JCL language uses strange forward slashes, fixed positions of key fields, etcetera. Many of these characters exist because the JCL

language was designed for punch cards. Punch cards could carry maximum 80 characters per line, and every line also needed some particular positions to control the operation of the punch card reader. Everything needed to be coded in uppercase to make things as simple as possible for the punch card reader. JCL is easily readable for a punch card reader device, but from an aesthetical and ergonomic perspective, it is a quite horrendous phenomenon. Punch cards have disappeared, but the JCL language still exists.

Returning to the above JCL snippet, you tell the computer to run PROGRAM1 with this script, use label INPUT as input file, referring to file MY.INPUT.DATA, and label OUTPUT as output file, referring to file MY.OUTPUT.DATA. If you feed this into the mainframe running z/OS it will try to execute this as such. Figure 8 illustrates this process.

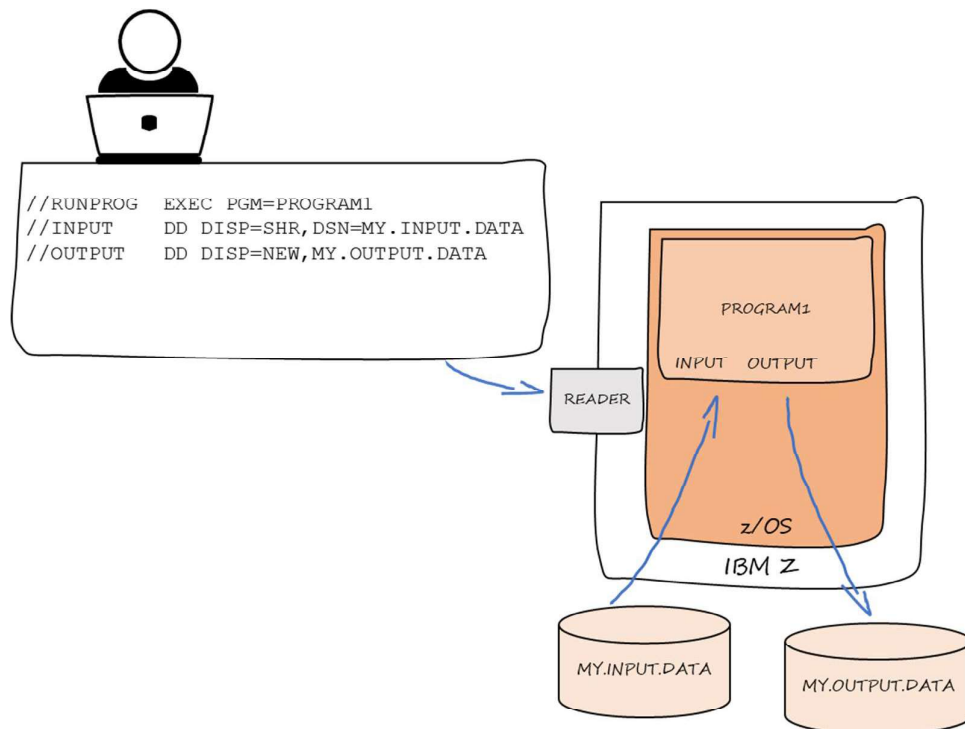


Figure 8 JCL to describe work to be done

In the old days the programmer punched the JCL on paper punch cards with a special device, a punch card machine, creating one card for every line of JCL. She then carried the stack of cards to the building where the computer was installed, and inserted into a punch card reader attached to the mainframe. The punch card reader read the stack of cards and executed the program.

Nowadays card reader devices do not exist anymore. They are replaced by a piece of software in z/OS. This software function is called the internal reader.