

1 Einführung

1.1 Eigenschaften eines Mainframes

Was ist Enterprise Computing? Das Zusammenspiel von Elementen die ab mittelgroßen Unternehmen in Bezug auf Server/Sicherung/Sicherheit auftreten.

Was ist ein Mainframe?

- Ein zentraler Server der in großen Wirtschaftsunternehmen häufig eingesetzt wird. I.d.R. von IBM hergestellt und mit einer System Z-Hardware und Architektur ausgestattet auf denen das z/OS-Betriebssystem (S/360, S/370, OS/390, MVS) läuft (Variante: zLinux).
- Reine Rechenleistung irrelevant (kein Supercomputer)
- **WICHTIG: Ein/Ausgabeleistung, Zuverlässigkeit/Verfügbarkeit, Sicherheit, Leistungsverhalten bei Transaktions- und Datenbankanwendungen**
- *Konkurrenz:* Siemens (BS2000, hardwarekompatibel), Bull und Unisys (nicht hardwarekompatibel)

Allgemeines

- System z und z/OS haben 64-Bit-Unterstützung und weitere Erweiterungen (z.B: Kryptographie)
- IBM garantiert, dass alle seit 1965 entwickelte S/360 Software ohne Recompilation läuft
- Gartner's platform positioning: IBM Mainframes stets Nr. 1.
- Mainframeeigenschaften sind umfangreicher als von Linux-/Windows-Server
- *System z hat technologische Führungsposition* (z.B. Sysplex, Clustering, JES, Hardware Protection, zLinux-Integration, CICS-Transaktionsmanager, MLC Multi-Chip-Module)
- **Verfügbarkeitsklassen:**
 - Sysplex mit 99,999% (10min pro Jahr)
 - Single Mainframe: 99,9% (8,7h pro Jahr)
- **Outage** (Ausfall):
 - geplante Gründe: Datenbank-Backup/Reorganisation, Release Wechsel, Netzwerk Rekonfig.
 - ungeplante: Permanente Hardware/Software Fehler, Datenprobleme (Inkonsistenzen)
- **Wartung:** automatische Fehlerkorrektur (automatic retry), Hamming Code Fehlerkorrektur + RAIM 5 Verfahren (ähnlich zu RAID5/6), über alle transienten Fehler werden geloggt und an IBM geschickt und bei Häufigkeit durch IBM korrigiert (mit Servicevertrag)

1.2 Total Cost of Ownership

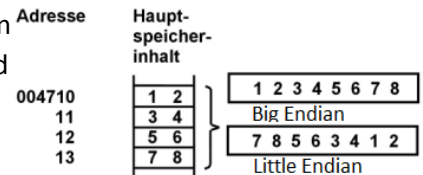
- SUN: Mainframes ab 1 Million, voll ausgerüstet auch 10 Mio. oder mehr
- Total Cost of Ownership (TCO): Die Kosten einen Arbeitsplatz mit einem Computer auszustatten. 1000\$ Hardware, 1000\$ Lizenzen, 8000\$ Administration/Support
- Durch Zentralisierung können die TCO gesenkt werden → Extremfall: Thin PCs (nur Browser)
Kosten: Windows < Linux < Mainframe ABER Administrationskosten bei Mainframes geringer
- TCO ist günstiger, je größer der Server und je mehr Benutzer er bedient
- TCO steigt bei großen Linux-Server (wegen schlechten Skalierungseigenschaften)
- Schwierigkeiten die Kostenvorteile von Mainframes umzusetzen
 - Fehlende Mainframe Spezialisten
 - Betroffene Fachkräfte müssen umgeschult werden
 - Distributed Computing Infrastruktur bietet Flexibilitätsvorteile
- Auslastung des Mainframes: 55% Legacy-Anwendungen (vor langer Zeit geschrieben), 33% neue Anwendungen > Anwendungen die von anderen Servern übernommen werden
- Aktuell hauptsächlich neue Mainframeinstallationen in Entwicklungsländern (China, Indien)

1.3 Mainframe Architektur

- Gängige Architekturen: x86, ARM, PowerPC, Sparc, Itanium, System z
- Viele Architekturen haben sich an die Mainframe Architektur angepasst (eine Art Standard) (z.B. 8Bit ein Byte, Einführung von Mehrzweckregistern, Verzicht auf direkte Hauptspeicher-Adressierung, Unterschied zwischen Supervisor- und Problem-Status)
- Ziel der S/360: Die Architektur so langlebig wie möglich zu gestalten. (nach wissenschaftlichen Grundsätzen entwickelt)
- Enterprise Software hat eine lange Lebensdauer. (alte Software wird legacy Software genannt)

Architektur

- System z ist eine 64-bit Architektur mit je 16 Mehrzweck-/Gleitkomma-/Steuerregistern mit 64 Bit und 16 Access Register mit 32 Bit. (Unterstützen auch noch einen 32-bit-Modus)
- Maschinenbefehle haben eine einheitliche Länge von 16, 32 oder 48 Bit. (mit 16Bit Befehlen kann um 20-30% der Speicherplatz im Hauptspeicher verringert werden (im Vergl. Zu x86)
➔ Performancevorteil (in Bezug auf die reduzierte Cache-Bandbreite)
- Unterstützt die Gleitkommaformate: Hexadezimal, IEEE 754, Dezimale Gleitkommaformat (DFP)
- Alphanumerische Zeichen in der Regel in **EBCDIC**-Format (selten ASCII, UTF8, UTF16)
- Verwendet **Big Endian** (wie Unix Rechner), x86 verwendet Little Endian
- **Hardware-Speicherschutz** teilt Hauptspeicher in 4096 Byte-Blöcke und weist diesen einen 4 Bit Storage Key zu. Bei Speicherzugriff wird dann der Key abgeglichen. Jeder Prozess hat einen anderen Schlüssel. (u.a. auch Schutz vor Bufferoverflow)
- **Hardware Management Console:**
 - **Firmware** (äquivalent zum BIOS) implementiert Maschinenbefehle. Firmware-Bereich ist außerhalb des Adressbereichs der Maschinenbefehle im Hauptspeicher. Wird durch IML (Initial Microcode Load) geladen.
 - Firmware ist auch während laufendem Betrieb vom Admin aufrufbar, um Subsysteme zurückzusetzen, Datum zu setzen, ...
- Da viele Platten angeschlossen sind wird I/O-Abarbeitung auf dezentrale Spezialrechner (**Control Units**) ausgelagert. ➔ ermöglicht hohen Datendurchsatz
- Auf Magnetbandspeicher werden selten benötigte Daten gespeichert. (großes Volumen)



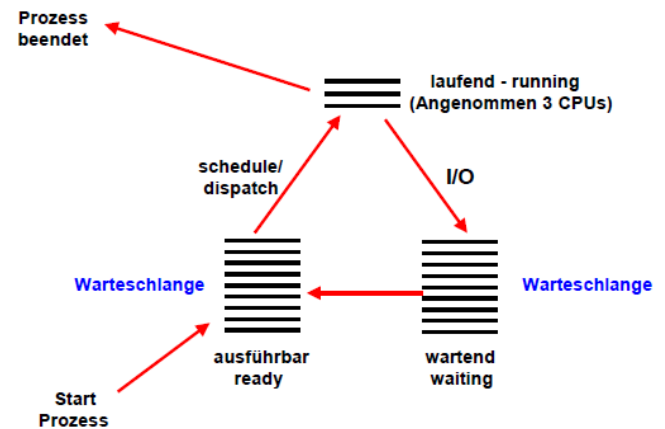
2 Verarbeitungsgrundlagen

2.1 Multiprogrammierung

- Compiler ➔ Linker ➔ Loader (kopiert Programm aus der Library in den Hauptspeicher)
- 32-bit Architektur: Der Adressenbus besteht 32
- 64-bit Architektur: Der Adressenbus besteht aus 64 Datenleitung (Im Endeffekt 8 Leitungen mit 1 Byte)
- Moderne CPU führt 10^9 Maschinenbefehle pro Sekunde aus
- Zugriffszeit zum Hauptspeicher beträgt 10^{-9} Sekunden (Piccosek.)
- Zugriffszeit zum Plattenspeicher beträgt 10 Millisekunden (Beschleunigt/Bremst mit 100g, schneller geht nicht, da sich sonst der stahl verbiegt)
- Beim Hochfahren wird ein Teil des Betriebssystems in den Hauptspeicher geladen
➔ Supervisor, Kernel, „Basic Control Program“ (BCP), Nucleus
- Menge aller Hauptspeicheradressen ist der Adressraum (Adress Space). Teilt sich (im einfachsten Fall) in User Space und Supervisor Space auf.

z/OS, OS/390	Windows/Unix
Problem State	User Mode
Supervisor State	Kernel Mode
Region	Virtueller Adressraum
Data Set	File
DASD (Direct Access Storage Device)	Plattenspeicher
Program Status Word	Status Register

- CPU läuft in jedem Augenblick entweder im Überwacherstatus oder im Benutzerstatus → bietet Schutz vor böswilligen Benutzerprogrammen die den Überwacheradressraum manipulieren
- Ein Prozess ist die Ausführung eines Programms auf dem Computer.
- Eine Transaktion ist eine spezielle Art eines Prozesses die garantiert, dass alle Daten auf dem Plattenspeicher entweder vollständig oder gar nicht erfolgen. (Siehe DB Transaktionen)
- Wenn ein Prozess eine I/O-Operation ausführt, wird er in den Wartezustand versetzt.
- Scheduler enthält Zeitschreiben (Time Slice) Komponente, damit ein Prozess nicht eine CPU endlos besetzen kann.
Ein Prozess erhält wenige ms Bearbeitungszeit und wird dann auf wartend gesetzt.
- Jeder Prozess erhält eine Region (teil des Benutzeradressraums) auf den er zugreifen darf
- Der Überwacher führt Steuerfunktionen der Benutzerprozesse aus. (z.B. teilt Ende einer I/O Operation mit)



2.2 Virtual Storage

- Der virtuelle Speicher wird in **Seiten** eingeteilt, diese sind gleich groß wie die **Rahmen** des realen Hauptspeichers.
- Es erfolgt eine willkürliche Zuordnung von Seiten- zu Rahmenadressen. Die Anordnung der Bytes innerhalb einer Seite wie Rahmen ist identisch (die untersten 12 Bit).
- Für jeden virtuellen Speicher gibt es eine Seitentabelle die sich im Überwacherteil des realen Hauptspeichers befindet. (genau genommen ist es eine 2-5 stufige Hierarchie von Seitentabellen)
- Virtueller Speicher kann größer als der reale sein mittels externem Seitenspeicher.
- Mehrere Prozesse haben in der Regel mehrere unabhängige virtuelle Speicher.
- Das ein- und auslagern von Seiten auf den realen Hauptspeicher vom externen Seitenspeicher wird als **Demand Paging** bezeichnet.
- Seiten des Überwachers werden (wegen der Performance) ständig im Hauptspeicher gehalten.

2.3 Betriebssystem Überwacher

- Neben der 16 Mehrzweck/Gleitkomma/Control/Access Register gibt es einen 64bit Befehlszähler und ein Flag Bit Register. Die Kombination von letzteren beiden ergibt das „Programm Status Wort“ (PSW, 128bit). Dieses definiert den Status eines CPU Cores.
- Speicherschutz: Benutzerprogramme können nur auf einen Teil des virtuellen/realen Hauptspeichers zugreifen. Privilegierte Maschinenbefehle können nur im Überwacherstatus ausgeführt werden.
- Unterbrechungen bewirken den Aufruf und die Ausführung von speziellen Programmen.
- **Unterbreckungsklassen:**
 - Maschinenfehler: Paritäts-Fehler (Bus, Hauptspeicher), fehlerhafte Addition
 - Reset: Setzt Zentraleinheit in jungfräulichen Zustand
 - I/O: Signalisiert den Abschluss einer E/A Operation
 - Extern: System externes Signal (Ablauf des Zeitgebers)
 - Programm: (Division durch 0, illegale Adresse, Fehlseitenunterbrechung)
 - Systemaufruf: (SVC) Aufruf des Überwachers im Benutzerprogramm

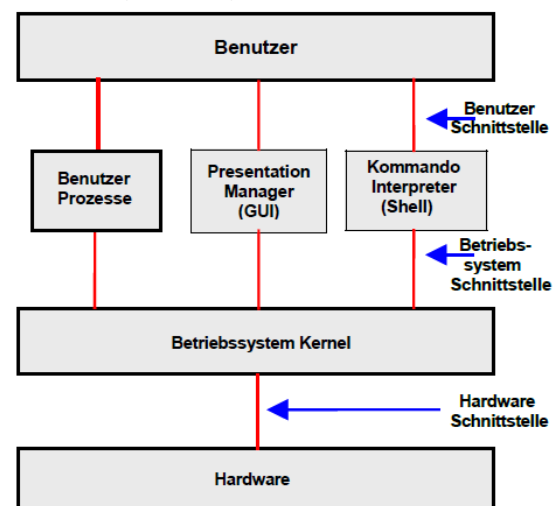
- **Interruption Process Flow:** abspeichern des PSW, laden des neuen PSW mit dem Befehlszähler auf erstem Befehl der Unterbrechungsroutine. (automatisch im Überwacherstatus)
- Das Betriebssystem besteht aus einer *zentralen Steuerungs- und Verwaltungseinheit* (Überwacher) und *weiteren Systemprogrammen oder Komponenten (Subsysteme)*.
Beispiel: TSO, Job Entry Subsystem (JES), DB2, CICS, ...
Subsysteme laufen parallel zu Benutzerprozessen.
- Der Überwacher ist in der Regel nicht strukturiert. ;)
- Die Funktionen des Überwachers werden von vielen Prozessen genutzt. In der Regel verbringt ein Prozess 50% der Ausführungszeit mit Überwacherefunktionen.
- Nur mittels **System Calls** (SVC) kann ein Benutzerprozess mit dem Überwacher kommunizieren. SVC wird im Benutzerstatus ausgeführt und leitet Überwacherstatus ein.
- Überwacher hat mehrere Komponenten/Funktionen: (z.B.):
 - Prozessverwaltungs-komponente: Prozesse aktivieren/deaktivieren
 - Scheduler/Dispatcher-komponente: Prozesse vom wartenden in den ausführbaren Zustand versetzen und entsprechend in die TCB Warteschlangen einreihen.
 - Virtuelle Speicherverwaltung (ordnet virtuellem dem realen Speicher zu)
 - Datenstrukturen (die Art der Formatierung)
 - Zeitscheibensteuerung
- **Task Control Block** (TCB): Enthält alle relevanten Informationen die der Prozess zum weiteren Ausführen benötigt (inklusive des PSW).

2.4 Hauptspeicher-Cache

- Static Random Access Memory (SRAM): schnellste und teuerste Form des Speichers
- Dynamic Random Access Memory (DRAM): langsamer aber billigere Speicherform
- eDRAM: Optimierte Form des DRAM, fast so schnell wie SRAM, braucht aber weniger Platz
- Mit der Kombination aus DRAM = Hauptspeicher und einem Cache aus SRAM-Speichern sind Zugriffszeiten von 100ps bis 10ns möglich.
- Existenz des Caches für den Benutzer nicht sichtbar (nicht mit Maschinenbefehlen manipulierbar)
- Cache wird in Cachelines mit je 256 Bytes aufgeteilt. Anordnung willkürlich → Cache Directory
- Ein Cache Miss zieht 8 Transferzyklen nach sich.
- Oft wird eine Cachehierarchie verwendet mit 2 Leveln: (heute oft noch komplizierter)
Level 1 ist in der Regel ein Split Cache mit schnellen & teuren SRAM Zellen.
Level 2 ist in der Regel ein Uniform Cache mit langsameren & billigeren SRAM Zellen.
- Split Cache: Teilt den Cache in Befehlscache (I-Cache) und Daten-cache (D-Cache) auf.

3 z/OS Betriebssystem

- Betriebssysteme die für die System z Plattform entstanden sind: z/OS, z/VSE, z/VM, z/TPF, zLinux, UTS 4 (Amdahl), Open Solaris (Sun), BS2000 (Siemens)
- **z/Transaction Processing Facility** (z/TPF): für spezialisierte Transaktionsverarbeitung (nur sehr einfache Funktionen), gilt als weltweit leistungsfähigste Transaktionsserver
- wichtigste Subsysteme: JES, TSO, Unix System Services
- Subsysteme sind in sich abgeschlossene Softwareeinheiten.
- System- und Anwendungsprozesse laufen in getrennten virtuellen Adressräumen (als Regions bezeichnet)



- z/OS wurde als reines Stapelverarbeitungssystem eingeführt
- Systemprogramme können klassische z/OS Hintergrundprogramme (Batch Programs) sein, Benutzeranwendungen, die unter der Kontrolle von Subsystemen laufen oder UNIX-Programme, die die UNIX System Services unter z/OS ausnutzen
- System Management Funktionen werden für die Steuerung und Überwachung des Ablaufes benötigt. (von IBM wie Drittanbietern verfügbar)

3.1 Job Control Language (JCL)

//	Label	Operation	Parameter	Kommentar
----	-------	-----------	-----------	-----------

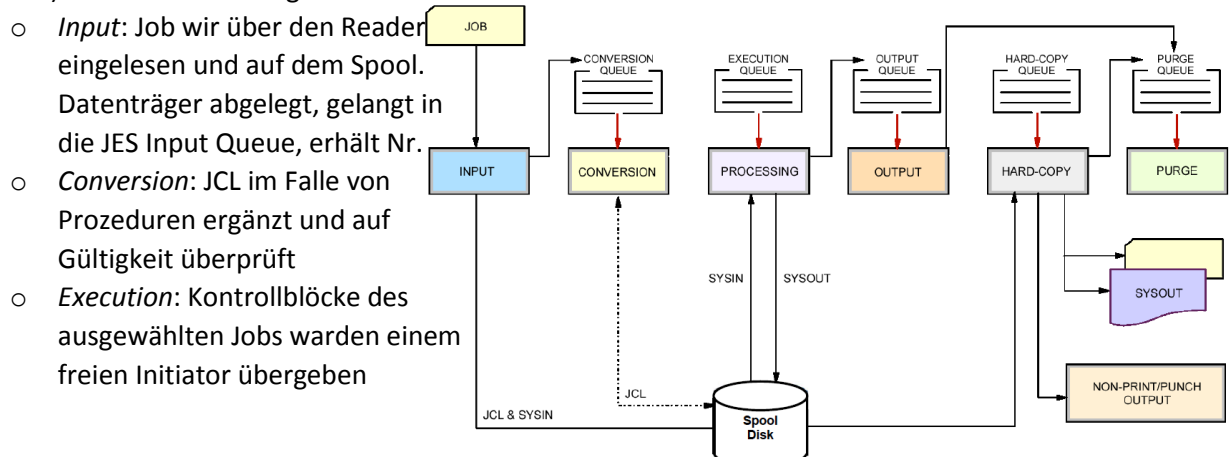
- Interaktive- vs. Stapelverarbeitung: Interaktiveverarbeitung blockiert den aktuellen Prozess
- Job Entry Subsystem (JES) ist ähnlich zu cron (in Linux)
- Für Jobsteuerung wird JCL verwendet. Ermöglicht „late Binding“ der verwendeten Dateien.
- Eigenschaften eines Stapelverarbeitungsprozesses:
 - Lange Laufzeit, periodisch, hohes Datenvolumen
 - Läuft asynchron zu irgendwelchen Benutzer-Aktionen
- JCL geht auf Lochkarten und hat damit 80 Spalten/Zeichen
- Einige Begriffe des Syntax:
 - JOB Statement markiert den Anfang eines Jobs
 - EXEC Statement bezeichnet Prozedur, die ausgeführt werden soll
 - PROC Statement gibt die Adresse einer Prozedur an
 - DD Statement bezeichnet die zu benutzenden Dateien (hier nicht verwendet)
- Wirkung von **sub**: JCL Script wird an JES übergeben. JES generiert einen Job mit Nr. und übergibt diesen an den JES Initiator. EXEC IGYWCL bewirkt ausführen des JCL Scripts IGYWCL aus der Programmbibliothek.

3.2 Job Entry System (JES)

- Ist für die Steuerung und Ablaufkontrolle aller Jobs (inklusive TSO-Sitzungen) zuständig.
- Bei einem Verbund aus Mainframerechnern (**Sysplex**) erfolgt die Steuerung durch JES3.
- Stapelverarbeitung: Queuing von Jobs, Prioritäts- und Ablaufsteuerung, Startzeit, Wiederholfrequenz, Durchsatzoptimierung, Zuordnung oder Sperrung von Ressourcen
- **Returncode** gibt an, ob der Job erfolgreich beendet wurde
- JES arbeitet nach dem **Master/Agent-Konzept**

Master: initiiert Jobs, ordnet zu Anfang der Bearbeitung die Eingabe/Ausgabe Ressourcen zu, reiht sie nach Priorität ein, kontrolliert den Ablauf, übernimmt die Freigabe der Ressourcen

Agent: starten die Anwendungsprogramme des Jobs, senden Statusinformationen, Laufzeitberichte
- Über mehrere Initiatoren (jeder bearbeitet einen Job zum Zeitpunkt t, insgesamt aber mehrere parallel) wird die Auslastung verteilt. Unterteilt in 5 Schritte:



- *Output*: erstellte Listen werden in der JES Output Queue gehalten, bis ein freier (virtueller) Drucker bereit ist die Listen auszudrucken
- *Purge*: Belegte Bereiche des Batchjobs werden auf dem Spool. Datenträger gelöscht.
- Master Konsole/Scheduler: Ermöglicht aufrufen des „BIOS“ auch während dem Betrieb.
- z/OS wird i.d.R. selten neugestartet. Initialisierung wird als Initial Program Load (IPL) bezeichnet. Gründe für IPL: Änderungen, äußere Einflüsse (Stromversorgungs-System testen), fehlerhafte Software belegt Systemressourcen (Grund einer Analyse)

3.3 Time Sharing Option (TSO)

- Single-User-Betriebssysteme sind Windows und CMS
- Multi-User-Betriebssysteme sind Unix und z/OS. (Mehrere Benutzer sind typischerweise im interaktiven, Zeitscheiben-gesteuerten Modus)
- Run-to-Completion-Modus: Wegfall der Zeitscheibensteuerung (vorteilhafte Performance)
- **TSO** ist eine Zeilenorientierte Shell (Vergleichbar mit DOS), die für Softwareentwicklung/Test und System Administration angewendet wird.
- **ISPF** (Interactive System Productivity Facility) ist eine Erweiterung der einfachen TSO Shell.
- Für jeden TSO-Benutzer wird beim Login von **TCAS** (Terminal Control Access Space) ein eigener (virtueller) Adressraum eingerichtet.

3.4 Datasets

- z/OS hat Dateien mit Struktureigenschaften (Datasets), aber auch strukturlose. Viele Datasets bestehen aus Records.
- Unter z/OS gibt es unterschiedliche Formatierungen und Zugriffsmethoden:
 - Direkter Zugriff
 - Sequentieller Zugriff
 - Index-sequentieller Zugriff
- Storage Management System (SMS): Komponente bei der freier Speicher angefordert wird
- **DFSMS Klassen**:
 - *Data Class*: Fasst Datasets mit identischen File System Attributen zusammen
 - *Storage Class*: Fasst Plattenspeicher mit identischen Performance Eigenschaften zusammen
 - *Management Class*: Fasst Plattenspeicher zusammen, wenn diese identisch verwaltet werden
 - *Storage Group*: Plattenspeicher mit derselben Storage und Management Class

3.5 Subsysteme

- Prozesse laufen in getrennten virtuellen Adressräumen. Threads sind unabhängige Ausführungseinheiten, laufen im Selben. → Threadwechsel erfordert weniger Aufwand
- Jeder Prozess kann Subtasks erstellen (ATTACH). Dies bringt großen Overhead mit sich, zeitkritische Subsysteme haben ihr eigenes Subtasking (z.B. CICS).
- Systems Network Architecture (**SNA**): Ähnlich zu TCP/IP, früher in Mainframes Standard
- z/OS Security Server ist ein Subsystem, welches für sicherheitsrelevante Belange zuständig ist. Enthält die Komponenten:
 - LAPD Server: Ein Verzeichnisdienst ähnlich zu DNS.
 - Kerberos Network Authentication Service (Kryptographie, Firewall Unterstützung)
 - Resource Access Control Facility (**RACF**): Ist IBMs Implementierung der Sicherheitsschnittstelle SAF. (alternativ: ACF2, Top Secret)
Zuständig für Authentifizierung, Autorisierung, Auditing (Logt Zugriffe), Überwachung
- Sicherheit hängt davon ab, wie korrekt die Einträge der RACF Datenbank sind.

4 System z/Hardware

- Konkurrenz: Fujitsu/Sun mit Solaris, HP mit HP-UX, IBM mit AIX
- Nachteil, wenn man in betriebswirtschaftlichen Großrechnern Technologien verwendet die für Arbeitsplatzrechner bzw. kleiner Server optimiert wurden:
 - Zuverlässigkeit und Verfügbarkeit richtet sich nach ökonomischen Kriterien, die für den PC Bereich etabliert werden. Nachträgliche Verbesserungen gehen ins Geld.
 - Ein/Ausgabe Einrichtungen sind für den Anschluss von 5 und nicht 50.000 Platten optimiert

4.1 Mikroprozessor Technologie

- System z verwendet an Stelle von Prozessor Boards sogenannte Books. Das Äquivalent zur **Fanout Adapter Card** wird als Host Channel Adapter (**HCA**) Card bezeichnet
- Außer IBM setzen alle Hersteller auf abgeleitete Technologien der PC-Welt. Unter anderem wird für Prozessor Boards die **Printed Circuit Board (PCB)** Technologie verwendet.
- Im Vergleich zu den Multi Chip Modulen (**MCM**) haben diese wesentlich weniger Schichten.
- MCM benützt die Multilayer Ceramic (MLC) Technologie. → schnellere Signallaufzeiten
- Neuestes Modell zEC12: 2,75 Milliarden Transistoren, 4 stufiger Cache
- **eDRAM** ist nahezu so schnell wie SRAM, benötigt aber viel weniger Platz
- System z hat **6 Cores pro Chip**, um mehr Platz für eine komplexe Cache Hierarchie zu haben
- Jeder Core enthält 2 Instruction Units, 2 Fixed Point Execution Units (FXU) und 2 Floating Point Execution Units (FPU). Alle Berechnungen verlaufen parallel ab und werden dann verglichen auf ihre Richtigkeit. Ansonsten greifen automatische Fehlerbehebungsmaßnahmen (Befehlswiederholung, instruction retry)
- **Recovery Unit (RU)**: In jedem Augenblick hält die RU den vollständigen Status der CPU. Damit kann selbst nach fehlerhafter Befehlswiederholung die Berechnung auf einen anderen Core verlagert werden.
- Kompression und Krypto Unit beschleunigt die Ver- und Entschlüsselung von SSL Nachrichten
- Cache-Hierarchie des zEC12: L4 Cache (384MB) wird von allen geteilt (über alle max. 4 MCM), L3 (48MB) teilen sich 6 Cores, L1 & L2 hat jeder Core extra
Die 4 L4 Caches der MCMs bilden einen sogenannten NUMA (Non-Uniform Memory Architecture) Cache.

4.2 Multichip Module

- Der z9 und der z196 Multichip sind vom Aufbau praktisch identisch. Die Anzahl der Chips ist jedoch halbiert, womit mehr Platz für Cache frei wurde.
- Ein MLC Modul hat etwa 100 Verdrahtungslagen.
- Die Anzahl der Transistoren pro Chip wuchs in den letzten 30 Jahren um einen Faktor 10^6 .
- Es können etwa 1800 Watt an Wärme abgeführt werden mittels des **Thermal Conduction Module (TCM)**. TCM gilt als besonders Zuverlässig. Verwenden Helium, weil es die beste Wärmeleitfähigkeit besitzt.
- **Land Grid Array (LGA)** ist ein Verbindungssystem für integrierte Schaltungen. Es ist verwandt mit dem *Pin Grid Array* (PGA-System) und dem *Ball Grid Array* (BGA-System)
- Ein System z196 kann maximal 4 Books enthalten. (Pro Book ein MCM)
- Mainframes greifen die Fanout Adapter Karten nicht auf den Plattenspeicher sondern auf Cache und nicht zwischen Plattenspeicher und Hauptspeicher. Damit wird eine höhere Datenrate möglich.

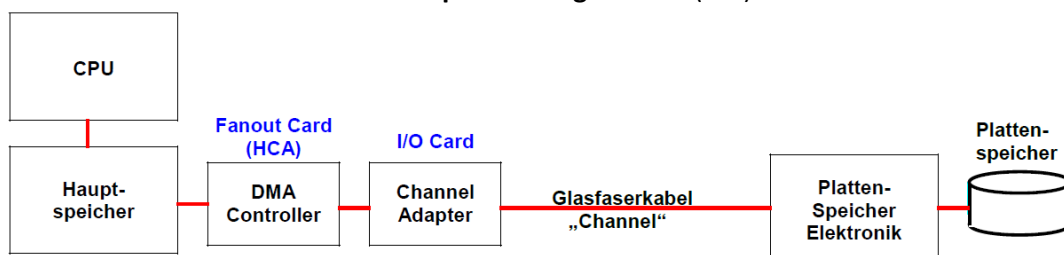
4.3 Book und System Frame

- Ein Book enthält Steckplätze für 30 Hauptspeicher DIMMs (Dual Inline Memory Module).
- Enthält weiter 8 Fan Out Adapter Karten (auch als Host Connector Adaptor (HCA) bezeichnet)
- Maximal kann ein Book 960 GB im Hauptspeicher haben.
- Es existiert eine doppelte Stromversorgung (Bulk Supply), um die Zuverlässigkeit zu erhöhen.
- Durch Voltage Transformation Modules (VTM) wird die Power Conversion bewirkt, welche Triple Redundancy verwendet.
- Drei verteilte Wandler-Baugruppen (DCAs) versorgen das Book mit Strom. Können während dem Betrieb ausgetauscht werden.
- Hauptspeicher enthält 4 Channels, jeder Channel verwendet Reed Solomon Fehlerkorrektur. Zusätzlich wird noch ein 5. RAIM Channel verwendet, ähnlich zu RAID. RAIM-DIMMs nehmen 20% des Speichers in Anspruch. → 768 GB realer Hauptspeicher
- Ein z196 Rechner enthält bis zu 4Books mit je 4 * 6 CPU Chips → max. 96 Prozessoren

5 Input/Output

5.1 Plattenspeicher Technologie

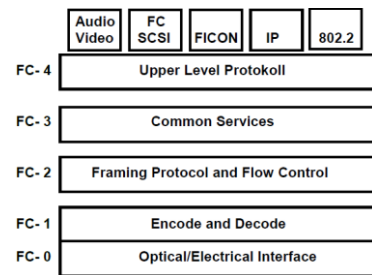
- Besteht aus mehreren Platten, von denen beide Oberflächen für die Aufzeichnung von Informationen genutzt wird. Information ist in kreisförmigen Spuren auf der Oberfläche angeordnet. Spuren sind in Sektoren aufgeteilt.
- Mindestens einen Lese/Schreibkopf. Mit 100g Beschleunigt/Gebremst, schneller ist technisch nicht möglich. Platte dreht sich mit 2/3 Schallgeschwindigkeit. 50nm Abstand zur Platte.
- Sind mit Helium gefüllt statt mit Luft für höhere Umdrehungszahlen.
- **Solid State Drive (SSD)**: Verwendet „Nand“ Flash-Speicher Chips. Teuer!
- Für performance-kritische Vorgänge sinnvoll. In Mainframe daher meist eine Mischung aus herkömmlichen Plattenspeichern und SSDs.
- Mittels Glasfaserkabel werden **Enterprise Storage Server (ESS)** mit dem Rechner verbunden



- ESS wird über Spezialrechner angesteuert. Auf Firmware von außen nicht zugreifbar.
- Der **DMA Controller** liest automatisch/unabhängig von der CPU die Daten aus dem I/O Puffer (im Hauptspeicher bezeichneter Bereich) und überträgt sie zum Plattenspeicher. Pro Byte wird die Länge im **Byte Count Register (BCR)** runtergezählt & SADR Register erhöht. Ist BCR = 0, so wird dies der CPU mittels Unterbrechung mitgeteilt, dass die I/O fertig ist.

5.2 SCSI und FICON

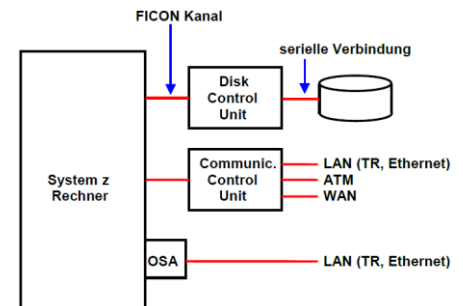
- Wichtige Protokolle: SATA (serial ATA), SAS (serial attache SCSI), iSCSI, FC-SCSI (Fibre Channel SCSI)
- Arbeitsplatzrechner verwenden SATA, Mainframes SCSI (Zuverlässiger, schneller, teurer)
- **ESCON** Channel erlaubt Datenraten von 17 Mbyte/s (veraltet)
- **FICON** Channel erlaubt Datenraten von 800Mbyte/s
- Fibre Channel FICON Protokoll verwendet Glasfasern, hat höheren Funktionsumfang (besser als SCSI)



- **Fibre Channel (FC)** ist für serielle & kontinuierliche Hochgeschwindigkeitsübertragung großen Datenmengen konzipiert. (in der Regel mit Glasfaser), ähnlich zu TCP/IP in 5 Schichten aufgeteilt →
- **FATA**-Platten (SATA mit Fibre Channel Anschluss) werden in Großrechnern eingesetzt, wenn Zugriffszeit & Zuverlässigkeit nicht höchste Priorität hat (z.B. bei Bilder, Videos, etc.). → als „**Nearline**-Platten“ bezeichnet
- **SCSI Adapter** verbindet mehrere Platten, Control Units oder Storage Server
- **Control Unit** führt Befehle aus, die vom Kanal-Adapter aus dem Hauptspeicher ausgelesen wurden, übergibt die Daten an die Platte/Kanal, Fehlerkorrektur, Kanalsbefehlswiederholung
- **Firmware**: Code den normale Benutzer nicht ändern/erweitern können, (früher hart verdrahtete Transistorlogik, heute dedizierter Mikroprozessor)
- z196-Rechner einige Cores (als System Assist Prozessoren (SAP) bezeichnet) die ausschließlich Firmware Code ausführen. (Benötigt für Fehlerbehandlung, Channel Subsystem, virtuelle Maschinen)
- Plattenspeicher typischerweise an 2 Control Units angeschlossen.
- Weg zwischen CPU und Plattenspeicher kann sich während der Ausführung einer I/O Operation dynamisch ändern.

5.3 Mainframe I/O

- Ein Volume bezeichnet einen Festplattenspeicher/Magnetbandkassette/etc.
- Jedes Volume ist (in Mainframes) durch eindeutige „**Volume Serial Number**“ (volser) gekennzeichnet.
- Spezifisch haben heutige Festplatten eine größere Speicherkapazität als die virtuellen 3390 Plattenspeicher. Das ESS emuliert daher mehrere virtuelle 3390 Platten (**Logical Volumes (LV)**) auf einer physischen Festplatte.
- Effektive Kapazität hängt von der Struktur der Data Sets und der Blocksize ab. VSAM Data Sets haben maximal 2.3 GB für Daten auf einer Modell 3 Festplatte (2,84 GB).
- Channel-to-Channel (CTC) Verbindung ermöglicht es 2 System z Rechner sich gegenseitig als I/O Einheit erscheinen zu lassen. Mit „Cross-System Coupling Facility“ (XCF) wird CTC für ein Cluster (Sysplex) verwendet.
- RAID-System (redundant array of independent disks) wird für Organisation mehrerer physischer Platten zu einem logischen Laufwerk verwendet. (i. d. R. mit Hot Swap)
 - RAID 0: Blöcke werden im Reißverschlussverfahren zu großer Platte zusammengeordnet, keine Redundanz, aber schnell
 - RAID 1: 2 Festplatten deren Inhalt stets gespiegelt wird.
 - RAID 3: Paritätsplatte, 80% Nutzbar, 1 Platte kann ausfallen ohne Verlust (min. 5 Platten)
 - RAID 5: wie RAID 4, Paritätsplatte wird auf alle Festplatten verteilt.
 - RAID 6: wie RAID 5, verkraftet aber den Ausfall von 2 Platten
 - RAID 10: Kombination aus RAID 0 und RAID 1, 2*n Festplatten
- Mainframes verwenden häufig RAID 6, es wird davon ausgegangen, dass Daten nie verloren gehen. → persistent



5.4 Enterprise Storage Server

- Es werden immer direkt größere Blocks gelesen, um sequentielle Zugriffe zu beschleunigen.
- Control Units werden in ESS implementiert.
- Cache grundsätzlich gefährlich falls Strom ausfällt (weil flüchtig) → Lösung: NVS

- I/O gilt als abgeschlossen, wenn Daten innerhalb der Non-Volatile Storage (NVS) sind. Speicherung erfolgt asynchron vom Betriebssystem.
- Ein Enterprise Storage Server besteht aus 4 Teilen:
 - Front End: Schnittstelle zu den Rechnern
 - Multiprozessoren inklusive to-read und to-write Cache. Kleiner Teil Davon NVS-Cache
 - Back End-Schnittstellen (Device Adapter) i.d.R. FC-AL (FC Arbitrated Loop)
 - Back End-Storage
- Mainframes werden über FICON und Unix über FC-SCSI an ESS angeschlossen.
- Magnetbandkassetten werden für Archivierung verwendet. (Lebensdauer von 30 Jahren)

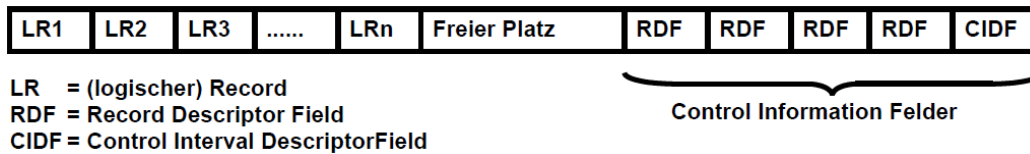
6 VSAM

6.1 Arten von Datasets

- Ein Rechner speichert Daten auf drei unterschiedliche Arten
 - In Mehrzweck- bzw. Gleitkommaregistern
 - Im Hauptspeicher
 - In Dateien (Files und Datasets)
- In Unix/Windows werden Files oft sequentiell gelesen. Es ist die Aufgabe der Anwendung (Tool, Datenbank, etc.) die Byte-Sequenzen auf „Records“ vorzunehmen.
- z/OS verwendet zwar auch Files und Filesysteme, meistens werden jedoch Daten in Datasets gespeichert. Datasets sind eine strukturierte Menge von Records
- Datenbanken verwenden Datasets.
- Datasets teilen sich in 4 grundsätzliche Arten auf
 - **Sequentiell:** Das System kann die Datensätze nur von Beginn an durchgehen.
 - **Indiziert:** Einzelne Records können per Schlüssel erreicht werden. (Access Method)
 - **Direkt – random:** Zugriff erfolgt direkt über Record Nummer, die das Anwendungsprogramm kennen muss.
 - **Partitioned:** Speichern Daten in untergliederten Komponenten (als Member bezeichnen)
- Beim Lesen von Records auf einer Platte wird die **„Queued“ Access Methode** verwendet. Es werden dabei benachbarte Records mitgelesen in der Hoffnung sie werden kurz darauf benötigt.
- Ein Datasetname besteht aus einem bis mehreren Qualifiern (je 1-8 Zeichen) mit bis zu 44 Zeichen. z.B. USER1.TEST.DATA (USER1 ← high level qualifier, DATA ← low level qualifier)
- VSAM ersetzt die älteren Dataset-Typen BSAM, QSAM, BDAM, ISAM etc. Sie werden in 4 Arten eingeteilt.
 - ESDS: ähnlich einem sequentiellen Dataset
 - KSDS: ähnlich einem indexsequentiellen Dataset
 - RRDS: ähnlich einem direkt organisierten Dataset
 - LDS: ein ESDS ohne jegliche Steuerinformationen
- Partitioned Dataset (PDS) ist eine Art Mini-Unix Filesystem. (PDSe optimierte Variante)
Es verfügt über ein Inhaltsverzeichnis und Platz für mehrere Dateien (Member). Die Dateien sind unstrukturiert (Byte-Stream). Einsatzbereich: Bibliotheken (ein Member ist ein Programm) (ermöglicht keine mehrstufige Hierarchie)
- Hierarchical File System (HFS) ist ein Unix kompatibles File System.
- Allgemein wird meist VSAM für Daten und PDSe für Programme verwendet
VSAM ist nicht für Source-Programme, JCL etc. vorgesehen
- Eine große Anzahl alter Programme verwenden non-VSAM Datasets

6.2 VSAM Struktur

- **Control Intervall (CI)**: mehrere logische Records, freier Platz, Control Interval Definition Field (CIDF), mehrere Record Definition Fields (RDFs)



- Aufgrund der Art wie VSAM Daten speichert und der Vielfalt der Verarbeitungsoptionen, können für einen einzigen logischen Record mehrere physische I/O Operation nötig sein.
- Control Intervalle werden gesplittet, wenn der freie Speicherplatz im CI nicht mehr groß genug ist. Es werden mind. 50% freier Speicherplatz bereitgestellt.
- CIs werden zu **Control Areas (CA)** zusammengefasst.
- Ein VSAM Buffer ist ein virtueller Speicherbereich in den ein CI während einer I/O Operation übertragen wird.
- Datenbanken nützen auch Bufferpools. DB2 und IMS transportieren Daten zwischen Festplatten und Hauptspeicher in Blöcken (Slots) von 4096 Byte.
- **Spanned Records** sind logische Records die mehrere Control Intervalle überspannen. Wird das letzte CI nicht ausgefüllt, so bleibt der Rest leer! (nie größer als eine CA)
- z/OS besitzt einen **Catalog** der den Speicherort aller Datasets (VSAM, non-VSAM, PDSE) kennt.
- **IDCAMS** ist ein Dienstprogramm zum Anlegen/Verwalten von VSAM Dateien.

6.3 VSAM Dataset Organisation – ESDS und KSDS

- Bei ESDS werden Records in der Reihenfolge des Eintreffens abgespeichert.
- ESDS unterstützt zwei READ Arten: sequentiell und direkt (mittels relativer Adresse (RBA))
- Die RBA hängt von der Länge der einzelnen Records ab!
- Bei KSDS ist der Schlüssel unabhängig von den Records, aber in jedem Record gilt für dieselbe Position derselbe Schlüssel. Schlüssel kann zwischen 1 und 255 Bytes lang sein.
- Neue Records werden bei KSDS entsprechend dem Schlüssel eingefügt und die anderen verschoben.
- KSDS kann sequentiell sowie direkt Records verarbeiten.
- Die Sortierreihenfolge von Alphanumerischen Zeichen ist bei ASCII, Unicode und EBCDIC verschieden. In VSAM werden Schlüssel nach EBCDIC sortiert.
- Bei KSDS Datasets gibt es zwei Komponenten. Dies sind die Index und die Datenkomponente.
- Ein VSAM-Index kann aus einer einzigen Ebene oder aus mehreren bestehen.
- **RRDS** (Relative Record Dataset) werden in Slots mit fester Länge geladen. Variable nur mit VRRDS.
- In **LDS** (Linear Dataset) existieren keine Records. Daher verläuft die Verarbeitung wie bei HFS.

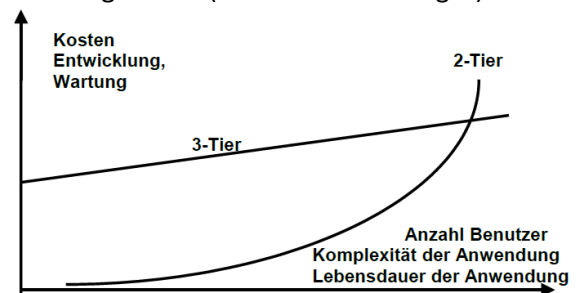
7 Transaktionsverarbeitung

7.1 Einführung

- Klient nützt einen Server Dienst. Server führt diesen aus. Dienst (Service) ist eine Software-Instanz, die auch auf mehreren Server Rechnern ausgeführt werden kann.
- Die meisten interaktiven Systeme wie Stapelverarbeitung verwenden Transaktionen
- Für einfache Interaktion mit Daten werden „Flat Files“ (z.B. VSAM) eingesetzt.
- Datenbankprozess läuft i.d.R. in eigenem virtuellen Adressraum. (oft auch mehrere)
- In 2-Tier Konfiguration befinden sich Anwendungsprozess auf dem gleichen Rechner wie der Klient. Bei 3-Tier sind Anwendung und Datenbank getrennte Prozesse.
- Schema legt fest, welche Daten gespeichert werden und wie deren Beziehung ist.

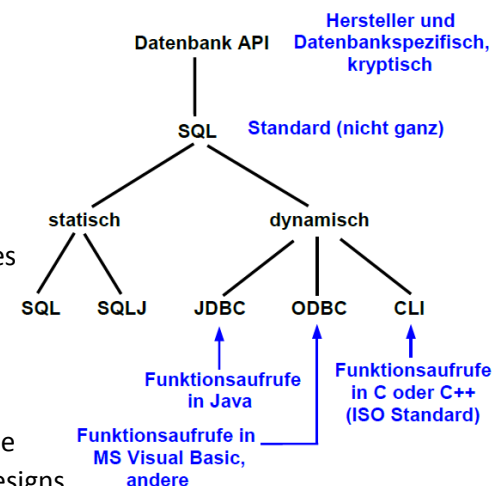
ZUSAMMENFASSUNG – ENTERPRISE COMPUTING – WS2013/2014

- Storage Group → DB → Table Space → Table
- Ein Table Space teilt sich in Slots (4096 Byte). Es wird stets ein ganzer Slot gelesen und in den realen Hauptspeicher transportiert.
- DBS optimiert Zugriffe auf die Daten durch Verwaltung von Buffer Pools. Eine Anzahl von derzeit benutzten Daten Items wird in I/O „Puffern“ im Hauptspeicher (Adressspace des DBS) gespeichert.
- Business Logic verarbeitet Eingabedaten und erzeugt Ausgabedaten (unaufbereitet)
- Präsentationslogik formt die rohen Ausgabedaten.
- Bei einer „Fat Client“ Konfiguration sind Business und Präsentationslogik auf dem Client. SQL-Anfragen werden über das Netz an den DB-Server gesendet. (2-Tier)
- 2 Varianten von 3-Tier: Anwendungs- und Datenbankprozesse getrennt!
 - Präsentationslogik auf dem Client. Business Logik auf dem Anwendungsserver. (z.B. SAPGUI)
 - Präsentationslogik und Business Logik auf dem Anwendungsserver (z.B. Java Server Pages)
- Transaktionen führen von einem definierten Zustand in einen anderen über. (ACID)
- **Atomicity Consistency Isolation Durability**
- 80% aller betrieblichen Anwendungen werden als Transaktionen verarbeitet
- **Ressource Manager** eines TP Monitors hat die Aufgabe die gleichzeitige Nutzung eines Anwendungsprogramms durch mehrere Transaktionen zu steuern.



7.2 Structured Query Language (SQL)

- SQL wurde von ISO und ANSI standardisiert. IMS verwendet DL/1 als alternative Programmiersprache.
- Relationale Datenbanken haben ein schlechteres Leistungsverhalten als nicht relationale (z.B. IMS). Trotzdem attraktiv!
- Statisches (Embedded) SQL arbeitet mit Precompiler. Dynamisches SQL ermöglicht Angabe von Parametern zur Laufzeit.



7.3 Stored Procedures

- Leistung: Stored Procedures > statisch > dynamisch
- **Nachteil Embedded SQL:** jedes SQL Statement erfordert getrennte Netzwerk-I/O Operation, detaillierte Kenntnisse des Datenbankdesigns notwendig, mehrere Kopien der Software erfordern hohen Administrationsaufwand
- Mit Hilfe einer Stored Procedure wird eine Gruppe von SQL Aufrufen auf dem Server ausgeführt. Die Gruppe hat ACID-Eigenschaften. Wählt mit connect die DB und ruft call auf. Erhält i.d.R. einen eigenen Adressspace. Benötigt wesentlich weniger CPU Zyklen!
- Stored Procedure Kann auch andere Anweisungen enthalten! (in Cobol, PL/I, C, REXX, Java, ...)
- **Nachteile:**
 - Stored Procedures sind komplex (schwieriges Debuggen)
 - Kann nicht auf mehrere DB (oder verschiedene DB-Systeme) gleichzeitig zugreifen

7.4 Transaction Processing Monitor (TP Monitor)

- Mit einem Transaktionsmonitor kann man ACID garantieren und gleichzeitig verschiedene DB-Systeme verwenden. (z.B. CICS, SAP R/3, MTS, Tuxedo)
- **Vorteile:** Lastverteilung, Skalierung, Prioritätssteuerung, Verfügbarkeit
- Ressource Manger kann – muss aber nicht – ein Transaktionsmonitor sein.

- Transaktion Monitor ⇔ Transaction Server ⇔ Transaction Service (grob gesehen Synonyme)
- Ein Transaktion Manager ist eine Komponente die den Transaktion Monitor steuert.
- Mittels Locks wird die Concurrency eingehalten. Es wird meist zwischen Read (**shared**) und Write (**exclusive**) Locks unterschieden. (Write Locks verhindern Lost Updates, Read Lock wird sobald ein Write Lock angefordert wird ungültig)
- „**Two-Phase Locking**“ fordert Locks in der ersten Phase ein und hebt in der zweiten Phase nur noch Locks auf.
- Ausfall des Klienten bevor Antwort vom Server eintrifft. → Server stellt Ergebnis der Verarbeitung in die Ausgabe Queue (vom Standpunkt des Servers abgeschlossen)
- Queue Manager des TP Monitors teilt in 3 Subtransaktionen auf:
 - Nimmt die Eingabenachricht entgegen und stellt sie in die Eingabe Queue. Commit
 - Dequeues, verarbeitet, stellt Ergebnis in die Ausgabe Queue, löscht Eingabeeintrag. Commit.
 - Sendet Ergebnis der Ausgabequeue an den Klienten. Löscht Eintrag. Commit.
- Dauerhaftigkeit wird durch **statisches Log** erreicht. Backup + Log = neuester Zustand
- **Dynamisches Log** enthält Einträge die nach erfolgreicher Durchführung der Transaktion gelöscht werden. Wird für Rollback/Backout/Abort/Abend benötigt.
- z/OS Resource Recovery Services (RRS) stellt einen Recovery Manager zur Verfügung der das „**Two-Phase Commit Protokoll**“ beinhaltet.
- Neue Resource Manager Produkte verwenden meist RRS. Existierende Transaction Manager wie CICS verfügen i.d.R. über Funktionen die auch in RRS enthalten wären. Ursprünglich besaß jede Komponente in z/OS, die Transaktionen ausführen konnte ihre eigenen Two-Phase Commit Recovery Manager.

8 CICS Transaktionsserver

8.1 CICS Übersicht

- **Transaction Processing Monitor** (TP Monitor) und DB-System laufen als unabhängige Anwendungsprozesse unter dem Betriebssystem-Kernel.
- TP-Monitor unterstützt sowohl Stapelverarbeitungs- als auch interaktive Transaktionen
- TP-Monitor vermeidet Nutzung der aufwendigen Betriebssystem-Funktionen, um Leistung und Durchsatz zu optimieren. Hat in der Regel eigene Queuing-Einrichtungen etc.
- **Transaction Processing Facility** (TPF) ist ein eigenständiges Mainframe Betriebssystem. Alle Anwendungen laufen im Überwacher-Status. Verwendet Run-to-Completion Scheduler/Dispatcher → hohe Effizienz bei relativ einfachen Transaktionen (z.B. Flugreservierungssystem, ...)
- CICS (**Customer Information Control System**) ist der am weitesten verbreitete Transaktionsmonitor → 30 Mia. Transaktionen pro Tag
- Auf etwa 10 Mio. Mannjahre werden die Weltweiten Investitionen geschätzt.
- Die Neuentwicklung alter CICS Anwendungen wird als „High-Risk Low-Reward“ betrachtet
- Mögliche Ansätze für die Transaktionsverarbeitung:
 - **Prozess-Ansatz:** Jede Transaktion läuft als selbstständiger Prozess
 - **Thread-Ansatz:** Jede Transaktion erhält einen Thread. Alle laufen im selben Prozess. Durch CICS Struktur-Eigenschaften sind diese trotzdem isoliert.
- Operationen über Adressraumgrenzen hinweg benötigen viele Mikrosekunden. → zu teuer
- Bei Transaktionskritischen Aktivitäten werden nie direkt Funktionen des Betriebssystems aufgerufen. Spezielle EXEC CICS Kommandos die der CICS Nucleus bearbeitet.

8.2 Ausführungsbeispiel einer CICS Transaktion

- Der Programm Code einer CICS Transaktion wird ausschließlich mit Hilfe seiner TIRD lokalisiert.
- CICS TRID's sind grundsätzlich immer 4 Zeichen lang.
- CICS selbst unterstützt über BMS (Basic Mapping Support) nur zeilenorientierte Ausgabe.
- Es gibt Softwarepakete die auch eine GUI ermöglichen (getrennte Prozesse).

8.3 CICS Nucleus

- CICS Nucleus ist ein Teil des CICS Prozesses (selber Adressraum)
- Durch den Zwischenspieler CICS Nucleus werden die ACID-Eigenschaften sichergestellt.
- Scheduling erfolgt durch den CICS Nucleus. (ähnlich wie z/OS Prozesse scheduled)
- Der CICS Nucleus ist die Schnittstelle zwischen Anwendungsprogrammen, Datenbanken und Netzwerkkomponenten. Weiter wird mittels „Storage Protection Key“-Mechanismus (wie in z/OS) die Isolation einzelner Transaktionen sichergestellt.
- „**quasi reentrant**“: Eine CICS-Anwendung wird nur zur Ausführung von EXEC CICS Statements unterbrochen. Vor und nach jedem EXEC CICS Befehl muss das Anwendungsprogramm in einem konsistenten Zustand sein.
- Übersetzung läuft in 4 Schritten: SQL Precompiler → **CICS Precompiler** → Compiler → Linker
- Der Code einer Transaktion wird nie doppelt in den Hauptspeicher geladen.
- CICS Nucleus Komponenten laufen in Domains.
- CICS Domänen: Jede Domäne enthält eine Gruppe von Objekten, die einen gemeinsamen Satz von Funktionen ausführen. (hier wird Manager und Control als synonym verwendet)
 - Terminal Manager
 - Transaction Manager
 - *Program Manager (PG)*: Laden/Ausführen von Anwendungsprogrammen, Tasks können Unterprogramme aufrufen, dessen Übergabe durch den PG gesteuert wird. Zum Finden eines Programms wird die Program Processing Table (PPT) verwendet.
 - *Storage Manager*: Für die Zuordnung von (virtuellem) Speicherplatz, der für die Transaktionsverarbeitung benötigt wird.
 - *File Manager*: Zugriff auf Daten in Data Sets mittels File Control Table (FCT)
 - *Task Manager*: Empfang von Transaktionsrequests, Erstellung/Steuerung von Tasks. Legt für jede TRID einen Control Block (Task Control Area) an.
 - *Domain Manager*: Unterhält Katalog mit Kenndaten der anderen Domains
- Neben Management Modulen und Transaktionen enthält die CICS Region noch weitere Arten:
 - Tabellen: Um Informationen über Terminals, Dateien, Anwendungsprogramme zu verwalten
 - Steuerblöcke: Mit ihnen wird der Status der Transaktionen festgehalten.
 - System Data Sets: Speichern von Ergebnissen, die von anderen Transaktionen verwendet werden. (z.B: Transaction Logging, System Recovery)
- CICS ist üblicherweise 24h am Tag im Betrieb und wird sehr selten runtergefahren.
- CICS Transaktionen haben eine maximale Ausführungszeit um Interaktivität zu gewährleisten.
- **COMMAREA** (Communication Area) ist eine Art Ein/Ausgabepuffer, über den eine Transaktion Daten mit dem Klienten (Terminal) austauschen kann. (früher häufig umgangen)
- **Execute Interface Block** (EIB) ist ein Bereich auf den Anwendungsprogramme mit EXEC CICS zugreifen.
- CICS Anwendungsprogramme können auf zwei Arten speichern:
 - **Temporary Storage** (TS) ist die Datei, die mit DFHTEMP angesprochen wird (VSAM-ESDS). DFHTEMP wird als Queue benutzt. (häufig gibt es für jeden Benutzer eine eigene Queue)
 - **Transient Data** (TD) wird mit DFHINTRA angesprochen (VSAM-ESDS)

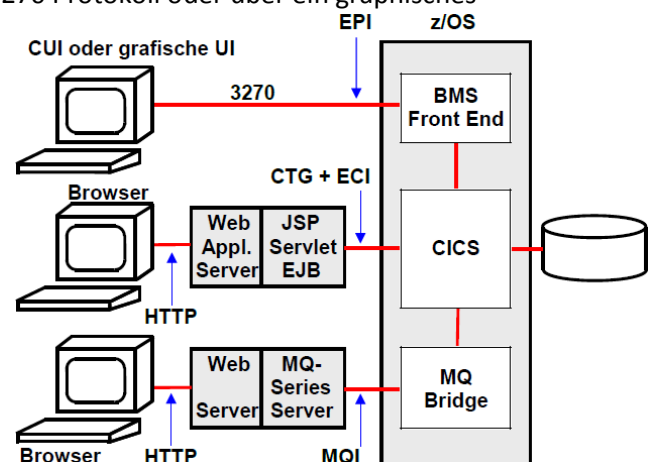
8.4 CICS Ablaufsteuerung

- Eine CICS Map enthält nur statische Daten. Mapinhalt wird während der Transaktionsausführung angereichert durch dynamische Informationen (farblich getrennt).
- Alle Maps die in einer Transaktion vorkommen werden als Mapset bezeichnet.
- Die Anwendungsentwicklung kann auf einem externen Rechner erfolgen. (z.B. in eclipse)
- Eine einfache Anwendung besteht aus 3 Teilen: Programm, Mapset, TRID → Group
- Bei der Installation werden die Referenzen zueinander aufgebaut und der TRID eingetragen.
- CICS ist ein logischer Server, der neben anderen auf einem physischen z/OS Server läuft.
- Ablaufsteuerung einer CICS Transaktion:
 - TRID aufrufen
 - Weiterleitung der TRID an Taskmanager
 - Program Manager lädt Programm (falls nötig)
 - CICS Prozess (Task) wird erzeugt
 - Terminal Manager baut ein Bildschirm Menü auf (BMS, Java Präsentationslogik)
 - Eingaben werden vom Terminal Manager entgegengenommen
 - File Manager liest/schreibt gewünschte Daten
 - Weitere Eingaben werden vom Terminal Manager entgegengenommen, ...
- Die CICS Shell hat auch schon interne Transaktionen (z.B. CEDA, CESF (login), CESN (logoff))

9 CICS Communication

9.1 3270 Protokoll

- Ein sauber strukturiertes Programm besteht aus 2 Teilen: Business & Präsentations-Logik
- Business Logik wird in C, C++, Java, ... geschrieben
- Präsentations-Logik mit BMS oder Java Servlets/JSP/Web Applikation Server umgesetzt
- Die Präsentationslogik bereitet die Informationen für die Endgeräte auf.
- CICS Bildschirmausgabe erfolgt entweder über das 3270 Protokoll oder über ein graphisches User Interface (http, Servlet, JSP)
- Das 3270 Protokoll unterstellt im Terminal einen 1920 Byte (24 Zeilen*80 Spalten) großen Puffer. Zusätzlich gibt es noch eine Statuszeile.
- Auf heutigen Arbeitsplatzrechnern wird die 3270 Bildschirmdarstellung mit Hilfe des 3270-Emulators simuliert. (verwendet Telnet Protokoll)
- 3270 Protokoll verwendet Untermenge der 256 Zeichen des ASCII oder EBCDIC Zeichensatzes.
- Die TCP/IP Version des 3270 Protokolls (das in SNA integriert ist) wird **TN3270** genannt.
- Mittels **EXEC CICS SEND MAP** wird eine entsprechende Map gesendet.



9.2 3270 Bildschirmausgabe Alternativen

- Die Struktur, in der die Daten zur Ausgabe in den COMMAREA I/O Puffers gespeichert werden, wird **Unit Record** bezeichnet.
- BMS ist keine Sprache sondern eine Sammlung an Assemblermakros Bestehend aus 3 Befehlen: DFHMSD, DFHMDI, DFHMDF

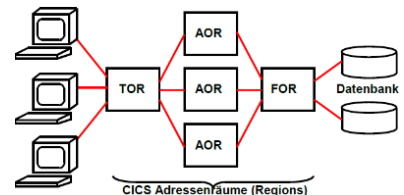
- **Screen Scraping** wird eine GUI für den 3270 Datenstrom bezeichnet. Der Datenstrom wird über die EPI Schnittstelle empfangen und erzeugt eine graphische Darstellung der Ausgabe. (Clientseitig). Problem: 3270 Protokoll lässt einige Funktionen nicht zu: Scroll Bar, mehrere Fenster
- Eine Server-seitige Lösung umgeht das Administrationsproblem. Dabei wird die 3270 Nachricht in ein Browser-fähiges HTML Format umgewandelt. → zentral steuerbar
- Im Optimalfall werden dann noch „Thin Clients“ verwendet, das heißt der Arbeitsplatzrechner wird nur noch auf das nötigste reduziert. (kein Plattenspeicher, etc.)

9.3 Multiregion and Intersystem Communication

- Um hohe Transaktionsraten (> 5000) zu erreichen, werden mehrere CPUs in einer „Symmetrischen Multiprozessor“ (SMP) Konfiguration und mehrere Mainframe Rechner benötigt.
- Mittels **MRO** oder **ISC** können zwei CICS Transaktionen miteinander kommunizieren.
- Multiregion Operation (MRO) wird für die Kommunikation zweier CICS Regions auf demselben Mainframe Rechner verwendet. MRO-Standards:

Hardware	klein/einfach	mittel	groß/komplex
Endbenutzer	100	10 000	> 100 000
CPU's	1 - 3	5 - 20	> 100
Plattenspeicher TeraByte	1 - 10	10 - 100	100 - 5 000
Magnetband Terabyte	100	100 - 1000	> 10 000

Software	x) einschl. Berichte, Maps xx) einschl. alte Versionen
Transaktionsprogramme X)	400 4 000 40 000
Quellprogramme XX)	1 000 10 000 100 000

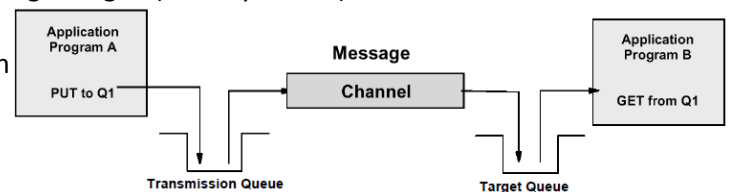
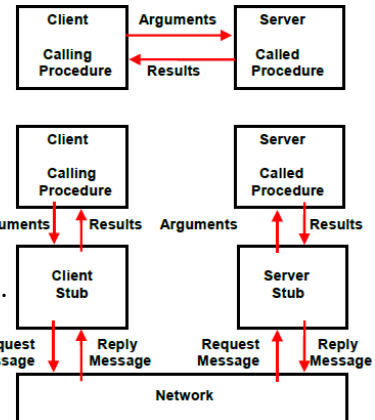


- *Transaction Routing*: Übergabe einer kompletten Transaktion an ein anderes System.
 - *Function Shipping*: Anwendung kann auf Daten eines anderen CICS System zugreifen.
- Häufig wird CICS auf 3 Regionen eingeteilt:
 - *Terminal Owning Region* (TOR)
 - *File Owning Region* (FOR)
 - *Application Owning Region* (AOR): Für verschiedene Aufgaben optimiert (z.B. Datenbankverbindungen, spezielle Zugriffsarten)
- AOR's können auf dieselben DB2-Daten zugreifen, da DB2 selbst mittels Lock Management parallelen Zugriff ermöglicht und steuert.
- CICSplex: Wenn mehrere CICS Instanzen in getrennten Regions installiert werden, nennt man die resultierende Architektur einen CICSplex. Als CICSplex wird eine Gruppe von logisch verbundenen CICS Regions betrachtet. Üblicherweise im Sysplex-Verbund eingesetzt.
- CICS Transaktionen können alternativ unter dem „Open TCP“ an Stelle des QR TCBs laufen, damit werden diese vom Betriebssystem gescheduled und auf mehrere CPUs laufen. Wird häufig von Java Programmen verwendet.
- Intersystem Communication (ISC) wird für die Kommunikation zweier CICS Systeme verwendet, die lediglich per TCP/IP verbunden sind. Sie verwenden einen klassischen Remote Procedure Call (RPC, synchron). Gegensatz: MQSeries ist weit verbreitetes asynchrones RPC Protokoll.
- Mittels EXEC CICS LINK wird eine Verbindung mit einem Programm eines anderen CICS-System aufgenommen.
- CICS ist auch auf anderen Systemen verbreitet. Distributed CICS genannt. Es verfügt über dieselbe API und kann damit mit dem regulären CICS kommunizieren. Anwendungen können in C, C++, PLI/I, COBOL und Java geschrieben werden und sind Problemlos auf die Mainframe CICS Version portierbar.

10 WebSphere MQ

10.1 Übersicht

- Ein physischer Server kann gleichzeitig viele verschiedene Serverdienste (Services) anbieten und bearbeiten. Ein CICS Server bietet entsprechend der TRIDs unterschiedliche an.
- **Local Procedure Call:** Eine aufgerufene Prozedur (Server) wird innerhalb des gleichen Adressraums wie die aufgerufene Prozedur (Client) ausgeführt.
- **Remote Procedure Call (RPC):** Der RPC erweckt den Anschein, als ob ein Client eine Prozedur aufruft, die sich im gleichen Adressraum befindet. In Wirklichkeit ruft die Client-Anwendung eine lokale Stub Prozedur auf.
- Arten von RPCs: Sun RPC, DCE RPC, CPI-C; DPL (nur zwischen CICS); OO: Cobra, Dotnet, RMI
- Beispiel eines asynchronen RPC: Electronic Mail
- **Message Based Queuing (MBQ)** ist ein Verfahren zur asynchronen Programm zu Programm-Kommunikation. (z.B. MQSeries (IBM), MSMQ (Microsoft), BEA MessageQ (Oracle))
Bietet sich für die Integration von heterogene Hardware und Software an.
Bietet ACID Eigenschaften, die sicherstellen, dass eine Nachricht genau einmal ausgeliefert wird.
- Queuing bedeutet, dass Programme über Nachrichten kommunizieren, die in Queues gespeichert werden. Dies muss nicht gleichzeitig erfolgen (nicht synchron). Das Ziel muss zum Absendezeitpunkt nicht verfügbar sein. WebSphere MQ kümmert sich um den Transport zur Zielanwendung und startet diese falls nötig.
- WebSphere MQ unterscheidet zwischen persistenten (Übertragung wird gewährleistet) und nicht-persistenten Nachrichten. Schlägt die Übertragung fehl, so wird sie beliebig oft wiederholt
- **MQSeries** ist der alte Name von WebSphere MQ.
- Der Kern von WebSphere MQ ist der (Message) **Queue-Manager (MQM)**. Ehe ein Anwendungsprogramm WebSphere MQ verwendet muss ein Queue Manager gestartet werden. Dieser besitzt und verwaltet die Ressourcen die von WebSphere MQ verwendet werden.
- Anwendungen kommunizieren mittels des **Message Queuing Interfaces (MQI)**. Alternativ kann auch **Java Message Service (JMS)** verwendet werden.

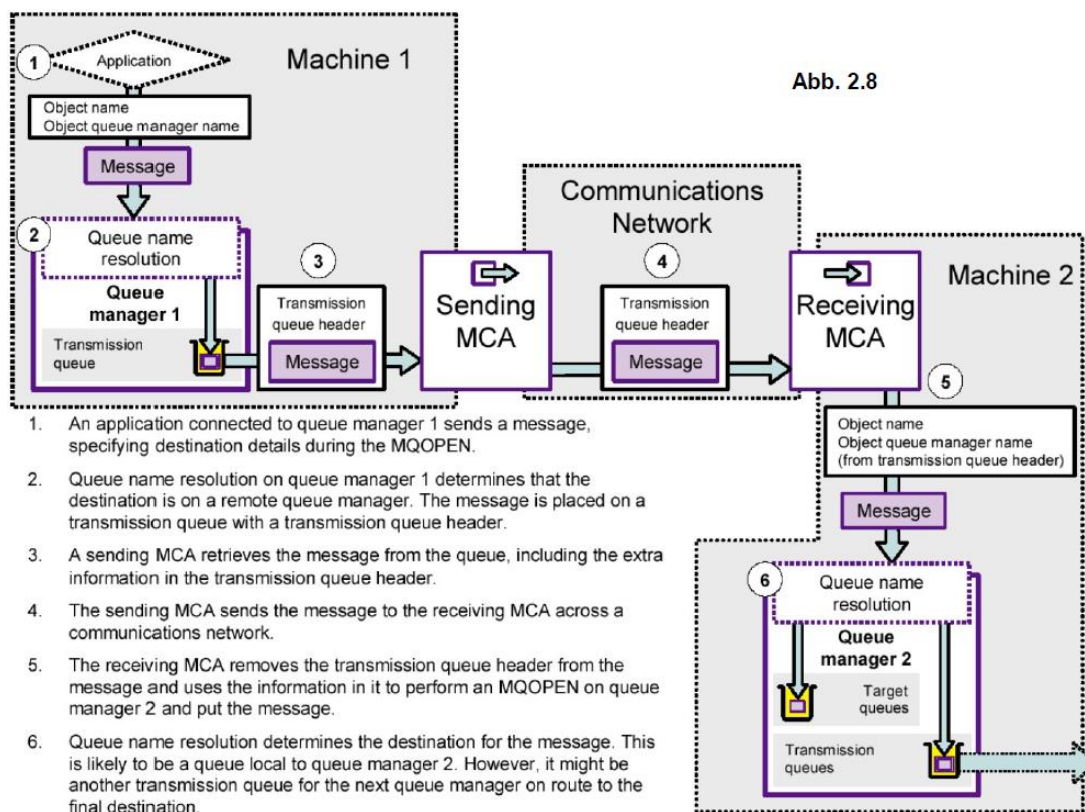
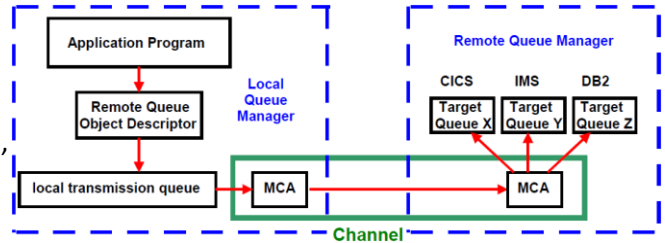


„Never underestimate the bandwidth of a station wagon full of tapes hurtling down the highway“

10.2 Queues und Channels

- Eine Message besteht aus 2 Teilen: Header und Daten
- Der Header beinhaltet einen **Message Descriptor (MQMD)**, der wiederum die Message-ID und verschiedene Steuerinformationen wie den Nachrichtentyp, Priorität, Expiration Time und den Namen für die lokale Queue für eine Antwort.
- In Bezug auf WebSphere MQ sind Queue und Message Queue austauschbare Begriffe.
- Eine Queue ist ein benanntes Objekt (bis zu 48 Zeichen lang, so lang wie ein Dataset Name), der von einem Queue Typ definiert wird. Der Queue Typ kann eine lokale Queue oder eine lokale Definition einer Target-Queue spezifizieren.
- Eine Transmission Queue ist immer eine lokale Queue und immer einem Channel und einem MCA fest zugeordnet.

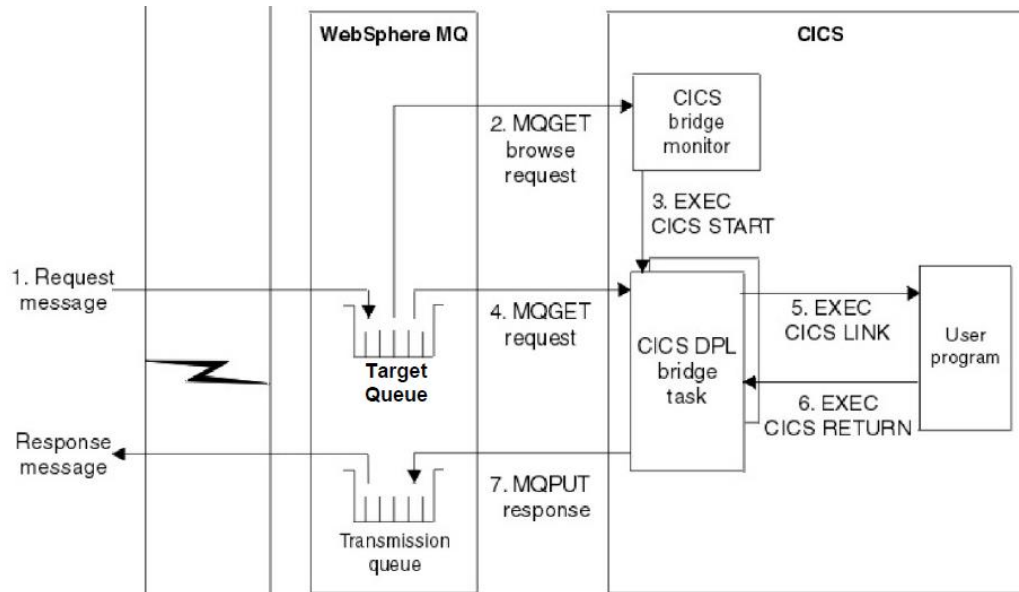
- Ein Channel ist mit einer einzigen **Transmission Queue** verbunden, kann aber mit mehreren Target Queues verbunden sein.
- Message Channel Agent (MCA) ist ein Programm, welches das Senden und Empfangen von Nachrichten steuert.
- MQ-Kanäle sind unidirektional. Für die bidirektionale Kommunikation müssen zwei Channel verwendet werden.
- Alle Queue Manager in einem Netzwerk müssen eindeutige Namen haben.
- Ein Channel Initiator ist ein Bestandteil eines Queue Managers. Dieser beherbergt alle MCAs.
- Listener Komponente wartet auf Nachrichten. Tritt eine Nachricht ein, wird ein passender MCA erstellt. MCA speichert Nachricht in die Target-Queue. (Standardport: 1414)



10.3 Trigger

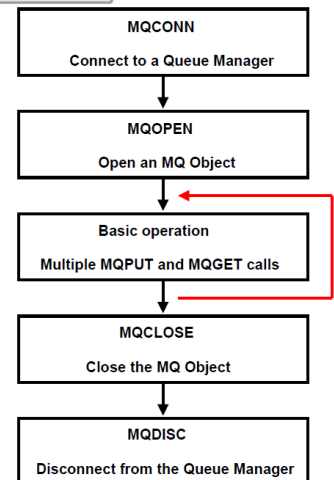
- Streng genommen stellt WebSphere MQ eine Peer-To-Peer-Kommunikation dar.
- Einfachste Variante von WebSphere MQ: „**Fire and forget**“
- WebSphere kann auch mittels zwei Channels ein Request/Response Modus simulieren.
- Die Idee der MQ Cloud ist es, dass der Client nicht weiß welches System in der Cloud die Nachricht erhält.
- WebSphere unterscheidet zwischen Slim und Fat Clients. Letztere haben einen Queue-Manager.
- **Multiple Target Queues:** Mehrere Nachrichten können die selbe Target Queue adressieren. Trotzdem kann eine einzige Initiation Queue und ihr Trigger Monitor alle Triggering Events bedienen.





10.4 MQI API

- **Handles:** Diese werden von den Queue-Manager MQ CONN und MQOPEN Kommandos zurückgegeben, und werden dann als Eingangsgrößen für die nachfolgenden MQPUT und MQGET Kommandos verwendet.
- **Return codes:** Es gibt ein Completion-Code und ein Reason-Code.
- Typischerweise wird eine Queue gelöscht, sobald das Programm, das sie geschaffen hat, ein MQCLOSE Kommando für diese Queue ausführt.
- Die meisten Unternehmen stehen vor dem Problem, bestehende Anwendungssoftware (in verschiedenen Sprachen) zu integrieren und auf verschiedenen Hardware-, Betriebssystem- und Middleware-Plattformen läuft. → WebSphere MQ gilt hier im Gegensatz als einfacher effizienter Weg



```

// 1 Connect application to a queue manager.
strcpy (QMName,"MYQMGR");
MQCONN (QMName, &HCon, &CompCode, &Reason);
// 2 Open a queue for output
strcpy (od1.ObjectName,"QUEUE1");
MQOPEN (HCon,&od1, MQOO_OUTPUT, &Hobj1, &CompCode, &Reason);
// 3 Put a message on the queue
MQPUT (HCon, Hobj1, &md, &pmo, 100, &buffer, &CompCode, &Reason);
// 4 Close the output queue
MQCLOSE (HCon, &Hobj1, MQCO_NONE, &CompCode, &Reason);
// 5 Open input queue
options = MQOO_INPUT_AS_Q_DEF;
strcpy (od2.ObjectName, "QUEUE2");
MQOPEN (HCon, &od2, options, &Hobj2, &CompCode, &Reason);
// 6 Get message
gmo.Options = MQGMO_NO_WAIT;
buflen = sizeof(buffer - 1);
memcpy (md.MsgId, MQMI_NONE, sizeof(md.MsgId));
memset (md.CorrelId, 0x00, sizeof(MQBYTE24));
MQGET (HCon, Hobj2, &md, &gmo, buflen, buffer, 100, &CompCode, &Reason);
// 7 Close the input queue
options = 0;
MQCLOSE (HCon, &Hobj2,options, &CompCode, &Reason);
// 8 Disconnect from queue manager
MQDISC (HCon, &CompCode, &Reason);
    
```