

Internet Anwendungen unter OS/390

**Dr. rer. nat. Paul Herrmannn
Prof. Dr.-Ing. Udo Kebschull
Prof. Dr.-Ing. Wilhelm G. Spruth**

WS 2004/2005

Teil 6

Transaktionsverarbeitung mit Java

**J. Gray, A. Reuter: „Transaction Processing,
Morgan Kaufmann Publishers, 1993.**

J. Horswill: „Designing & Programming CICS Applications“. O’Reilly, 2000.

Transaktionen

Transaktionen sind Client-Server-Anwendungen, welche die auf einem Server gespeicherten Daten von einem definierten Zustand in einen anderen überführen.

Eine Transaktion ist eine atomare Operation. Die Transaktion wird entweder ganz oder gar nicht durchgeführt.

Eine Transaktion ist die Zusammenfassung von mehreren Datei- oder Datenbankoperationen, die entweder

erfolgreich abgeschlossen wird, oder die Datenbank unverändert läßt

Die Datei/Datenbank bleibt in einem konsistenten Zustand: Entweder vor Anfang oder nach Abschluß der Transaktion

Im Fehlerfall, oder bei einem Systemversagen werden alle in Arbeit befindlichen Transaktionen abgebrochen und alle evtl. bereits stattgefundenen Datenänderungen automatisch rückgängig gemacht.

Wird eine Transaktion abgebrochen, werden keine Daten abgeändert

ACID Eigenschaften

Atomizität (Atomicity)

Eine Transaktion wird entweder vollständig ausgeführt oder überhaupt nicht

Der Übergang vom Ursprungszustand zum Ergebniszustand erfolgt ohne erkennbare Zwischenzustände, unabhängig von Fehlern oder Crashes. Änderungen betreffen Datenbanken, Messages, Transducer und andere.

Konsistenzerhaltung (Consistency)

Eine Transaktion überführt das System von einem konsistenten Zustand in einen anderen konsistenten Zustand

Daten sind konsistent, wenn sie durch eine Transaktion erzeugt wurden. (Datenkonsistenz kann nicht zu Beginn einer Transaktion überprüft werden).

Isolation

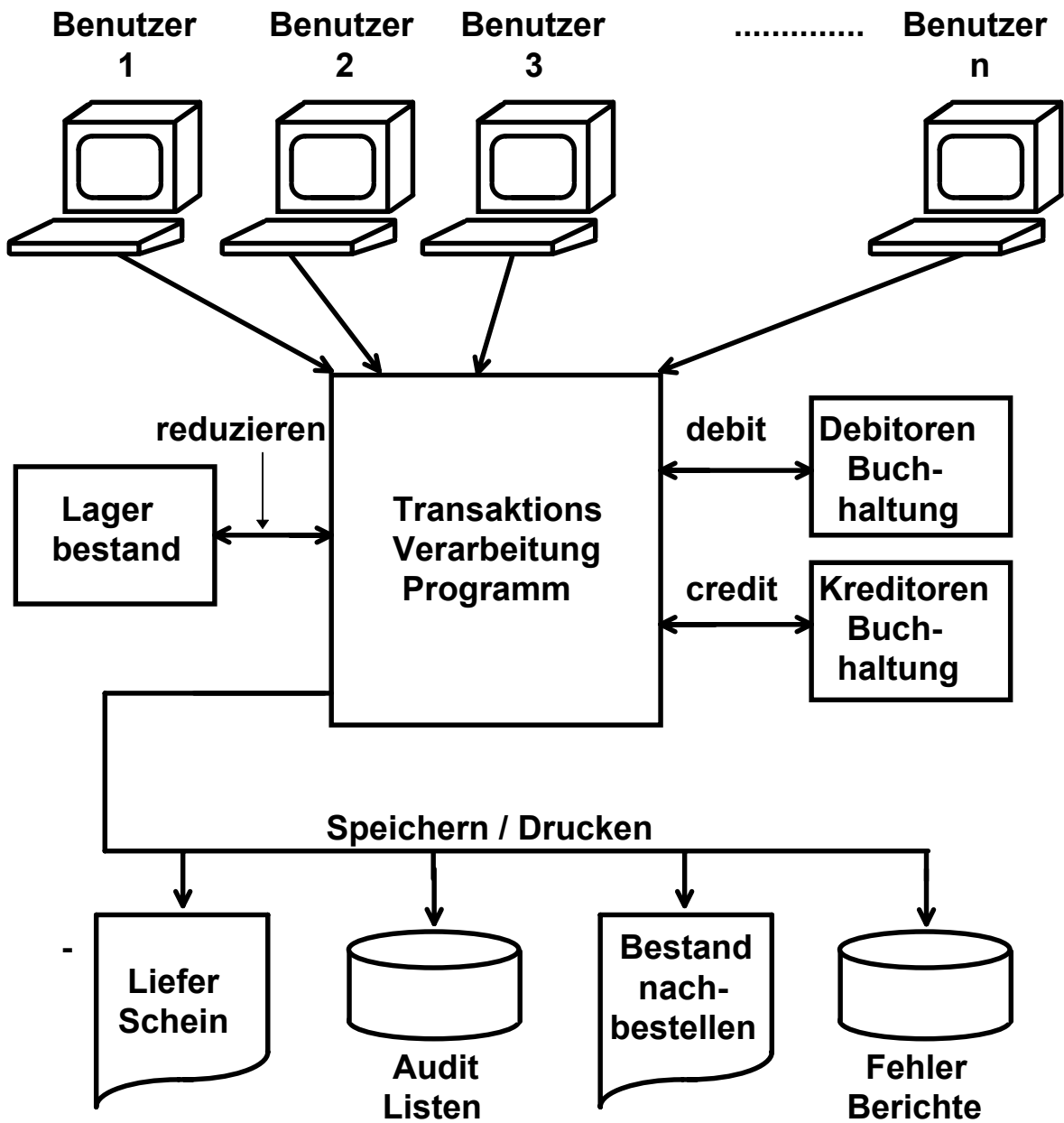
Die Auswirkungen einer Transaktion werden erst nach ihrer erfolgreichen Beendigung für andere Transaktionen sichtbar

Single User Mode Modell. Selbst wenn 2 Transaktionen gleichzeitig ausgeführt werden, wird der Schein einer seriellen Verarbeitung gewahrt.

Dauerhaftigkeit (Durability)

Die Auswirkungen einer erfolgreich beendeten Transaktion gehen nicht verloren

Das Ergebnis einer Transaktion ist real, mit allen Konsequenzen. Es kann nur mit einer neuen Transaktion rückgängig gemacht werden. Die Zustandsänderung überlebt nachfolgende Fehler oder Systemcrashes.



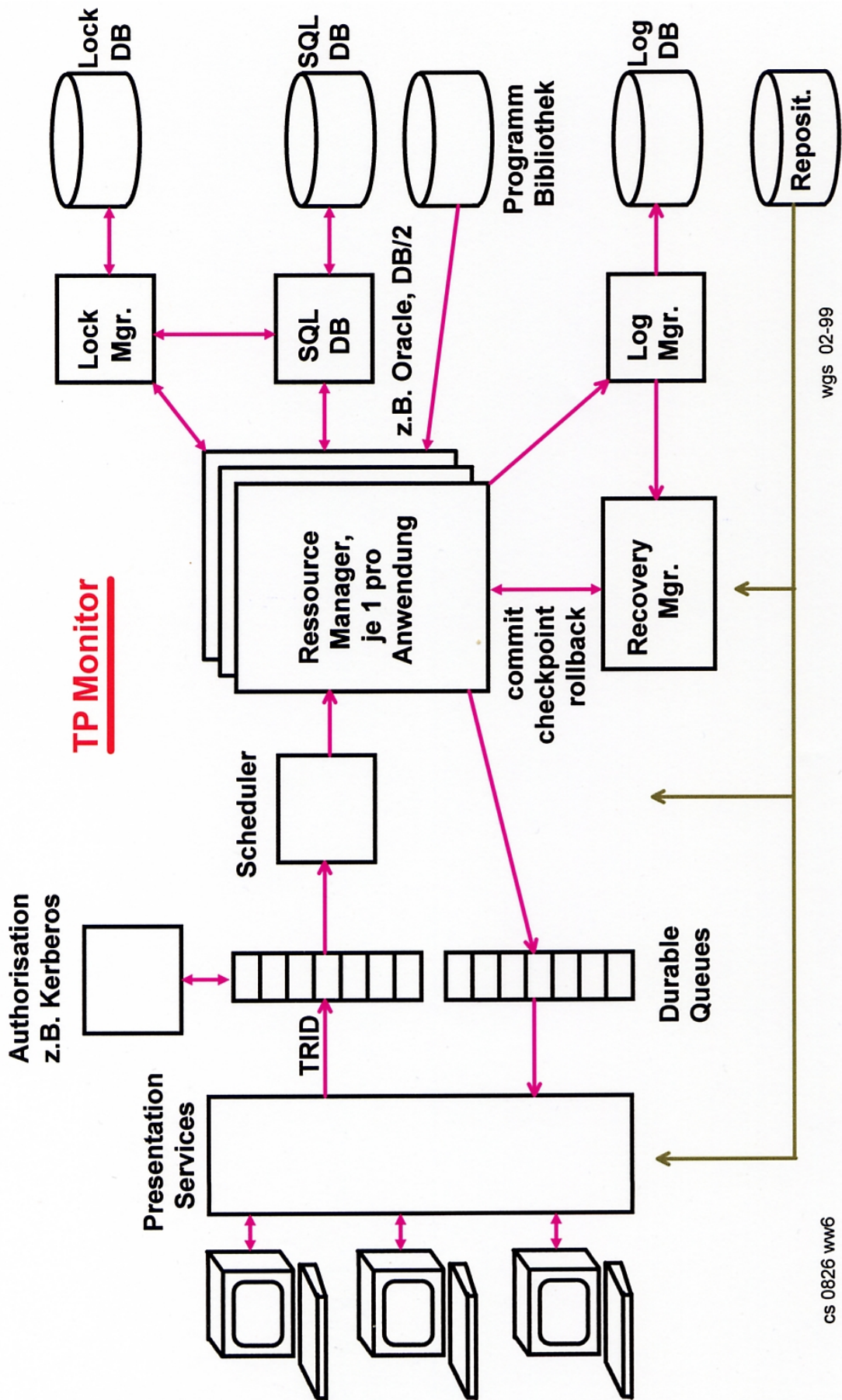
**Beispiel für eine
Transaktionsverarbeitungsanwendung:
Auftragseingang-Bearbeitung**

Eigenschaften eines Transaktionsmonitors:

Laufzeitumgebung für Transaktions-Anwendungen
hohe Verfügbarkeit
kurze Antwortzeit (< 0.3 Sek. erwünscht)
geeignet für hohes Verkehrsaufkommen
niedrige Kosten pro Transaktion
**Integrität beim Zugriff auf gemeinsam genutzte
Ressourcen**

Beispiel für Transaktionsmonitore

BEA Tuxedo (AT&T→Novell → BEA)
IBM CICS
IBM IMS-DB/DC
IBM TPF
Microsoft Transaction Server (MTS)
SAP R/3
Siemens UTM
Tandem Pathway
Transarc (OSF) Encina



CICS

Customer Information Control System

Der am weitesten verbreitete, IBM proprietäre Transaktionsmonitor.

Verfügbar unter den zSeries und S/390-Betriebssystemen, sowie in modifizierter Form (als Encina Erweiterung) unter OS/400, OS/2, Windows, AIX, HP-UX, Sinix, Solaris sowie Digital Unix.

Unter z/OS (und VSE) laufen alle Anwendungen und Dienste im Problemstatus, ungeschützt voneinander innerhalb eines einzigen virtuellen Adressenraums. Anwendungen und Ressource Manager laufen als Threads innerhalb dieses Adressenraums.

Spitzenposition bezüglich Durchsatz, Zuverlässigkeit und Verfügbarkeit.

Es wird angenommen, daß bei 490 der weltweit 500 größten Unternehmen CICS eingesetzt wird.

Eine detaillierte CICS Beschreibung ist zu finden unter

http://www.www-ti.informatik.uni-tuebingen.de/~spruth/index_de.html

im Unterverzeichnis

Transaction Processing / CICS / Einführung

**In 2000 war die Anzahl der weltweit
ausgeführten
CICS Transaktionen etwa so groß
wie die Anzahl der Hits auf Seiten des World-
Wide Web.**

**In 1999 setzen weltweit etwa 15 000 Unternehmen CICS ein.
Von den 2000 größten Unternehmen setzen > 90% CICS
ein.**

Sie generierten etwa 20 Milliarden Transaktionen pro Tag.

Es existieren etwa 30 Mill. CICS Terminals weltweit.

**Zum Vergleich existierten weltweit 379 Mill. Internet
Anschlüsse im März 2001, die meisten davon in Privat-
haushalten.**

**Durchschnittliche CICS Terminal Benutzungsdauer
4 - 6 Stunden / Tag.**

**Durchschnittliche Internet Benutzungsdauer
etwa 10 Stunden / Monat.**

<http://www.hursley.ibm.com/infopack/A33578.pdf>

J. Gray: How High is High Performance Transaction Processing?
<http://research.microsoft.com/~Gray/Talks/>

R. Fox: „Net Population Newest Numbers“. Comm. ACM, Vol. 44, No.7, July 2001, P.9 .



CHARGE ACCOUNT – CUSTOMER APPLICATION FORM

Customer's name: DAVID MOUNCE

Home Address: 79 WISTFUL VISTA

PLEASANTVILLE NEW YORK 10549

Telephone Number: 751 248 3960

Date: 03/27/84 Signature: David C. Mounce

Other Account Users

Name: CHRISTA MOUNCE (wife) Name: PETER MOUNCE (son)

Address: as above Address: as above

OFFICE USE ONLY

Account Number: 12345

No. of cards issued: 2

Reason: L Date: (MM/DD/YY) 04/01/84

(N - new L - lost S - stolen R - revised)

Special codes: AJ Approved by: CES

Kunden Kredit Antrag

Beispiel

KanDolt Großkaufhaus KundenKreditverwaltung

Kundendatei als index-sequentielle VSAM Datei

Field	Length	Occurs	Total
Account Number (Key)	5	1	5
Surname	18	1	18
First Name	12	1	12
Middle initial	1	1	1
Title (Jr, Sr, and so on)	4	1	4
Telephone number	10	1	10
Address line	24	3	72
Other charge name	32	4	128
Cards issued	1	1	1
Date issued	6	1	6
Reason issued	1	1	1
Card code	1	1	1
Approver (initials)	3	1	3
Special codes	1	3	3
Account status	2	1	2
Charge limit	8	1	8
Payment history:	(36)	3	108
-Balance	8		
-Bill date	6		
-Bill amount	8		
-Date paid	6		
-Amount paid	8		

ACCOUNTS MENU

TO SEARCH BY NAME, ENTER SURNAME AND IF REQUIRED, FIRST NAME

SURNAME : (1 TO 18 ALPHABETIC CHRS)
 FIRST NAME : (1 TO 12 ALPHABETIC CHRS OPTIONAL)

TO PROCESS AN ACCOUNT, ENTER REQUEST TYPE AND ACCOUNT NUMBER

REQUEST TYPE: **A** (D-DISPLAY, A-ADD, M-MODIFY, X-DELETE, P-PRINT)
 ACCOUNT : **26004** (10000 TO 79999)
 PRINTER ID : (1 TO 4 CHARACTERS (REQUIRED FOR PRINT REQUEST))

ACCT	SURNAME	FIRST	MI	TTL	ADDRESS	ST	LIMIT
26001	Meier	Rolf	A	MR	Ritterstr. 13	N	1000.00
26002	Meier	Steffie	G	MRS	Wilhelmstr. 24	N	1000.00
26003	Meier	Tobias	A	MR	Nikolaistr. 23	N	1000.00

ENTER DATA AND PRESS ENTER FOR SEARCH OR ACCOUNT REQUEST OR PRESS CLEAR TO EXIT

Enter

ACCOUNTS

ADD ACCOUNT NUMBER 26004

SURNAME : **Meier** (18 CHRS) TITLE : **DR** (4 CHRS OPTIONAL)
 FIRST NAME : **Walter** (12 CHRS) MIDDLE INIT: (1 CHR OPTIONAL)
 TELEPHONE : **733456** (10 DIGS)
 ADDRESS LINE1: **Heilbronnerstr. 91** (24 CHRS)
 LINE2: **70109 Stuttgart** (24 CHRS)
 LINE3: (24 CHRS OPTIONAL)

CARDS ISSUED : **1** (1 TO 9) CARD CODE : **A** (1 CHR)
 DATE ISSUED : **11 22 99** (MM DD YY) REASON CODE: **L** (N,L,S,R)
 APPROVED BY : **DEF** (3 CHRS)

UPTO 4 OTHERS WHO MAY CHARGE (EACH 32 CHRS OPTIONAL)
 O1: O2:
 O3: O4:
 SPECIAL CODE1: CODE2: CODE3: (EACH 1 CHR OPTIONAL)

NO HISTORY AVAILABLE AT THIS TIME CHARGE LIMIT STATUS

NOTE:- DETAILS IN BRACKETS SHOW MAXIMUM NO. CHARACTERS ALLOWED AND IF OPTIONAL

FILL IN AND PRESS "ENTER," OR "CLEAR" TO CANCEL

Enter



Enter account number:

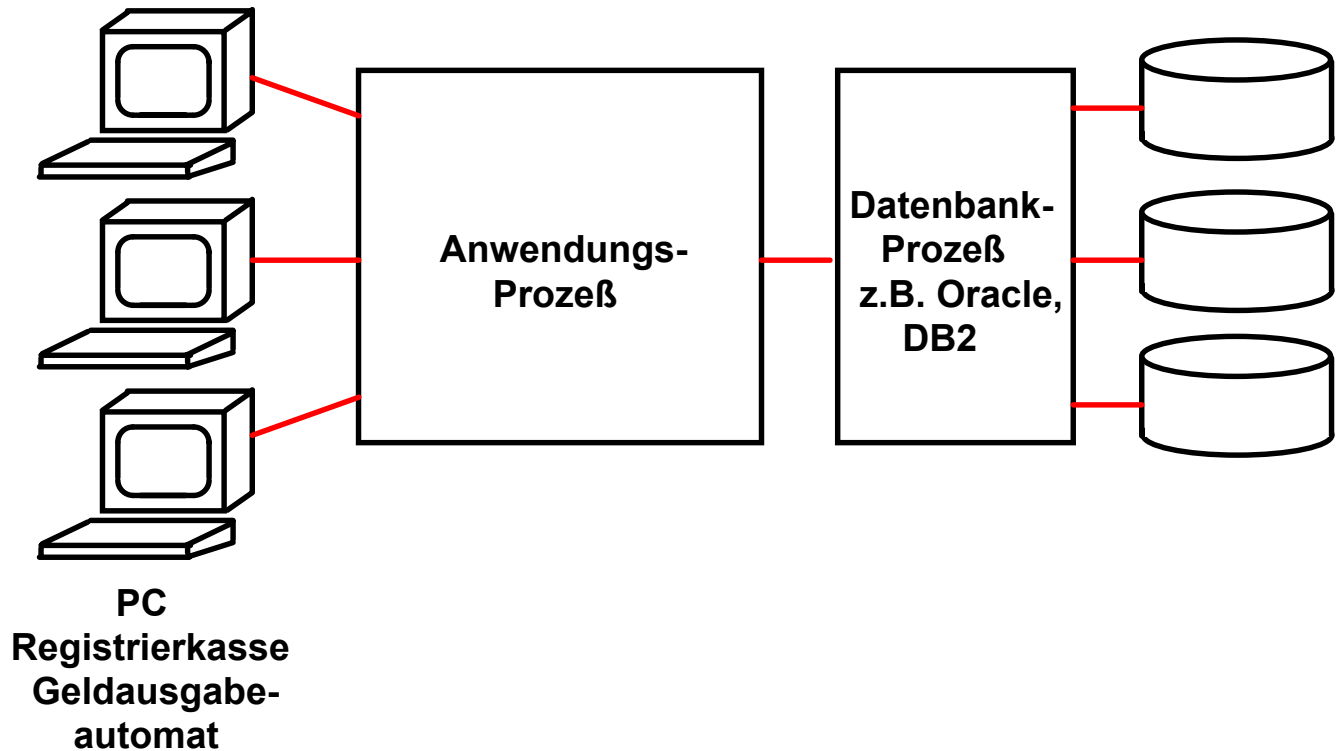
Title: DR
 Initial: R
 First name: Walter
 Surname: Meier
 Address: Heilbronnerstr. 91
 70109 Stuttgart
 Telephone: 0000733456

Others Who May Charge:

No. Cards Issued: 1
 Date Issued: 11-22-99
 Reason: L
 Card Code: A
 Approved By: DEF
 Special Codes:
 Account Status: N
 Charge Limit: 1000.00

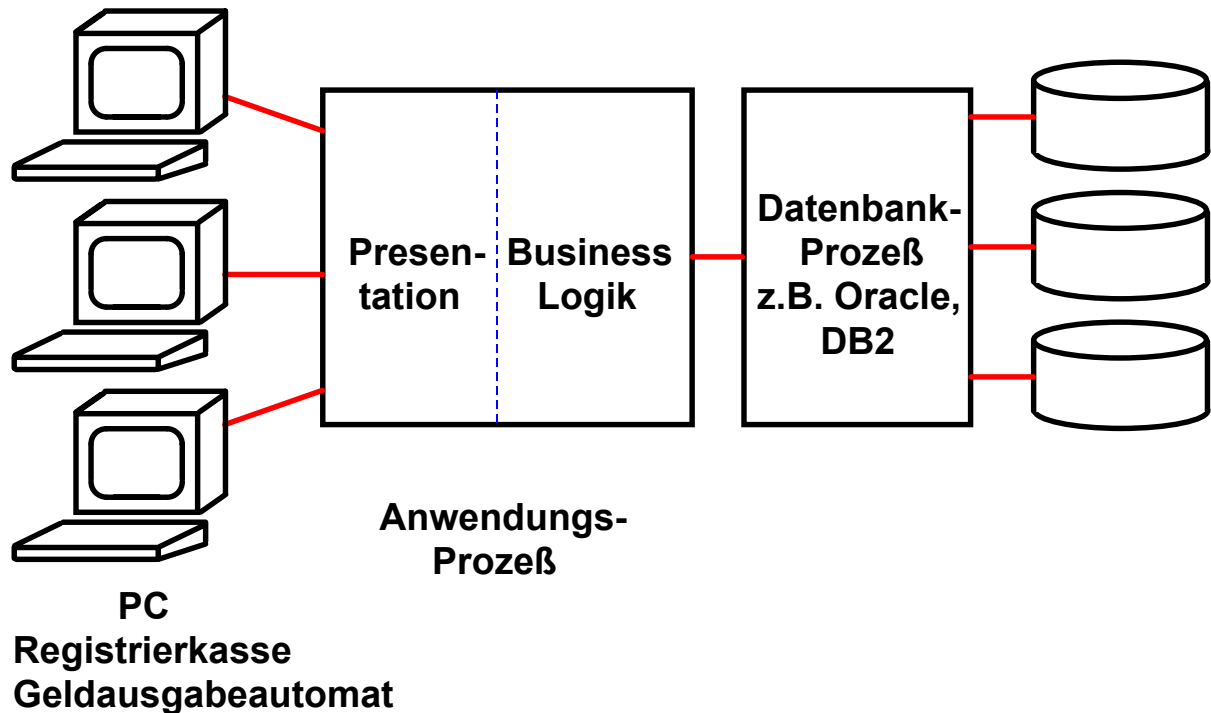
Account History

Balance	Billed	Amount	Paid	Amount
0.00	00-00-00	0.00	00-00-00	0.00
0.00	00-00-00	0.00	00-00-00	0.00
0.00	00-00-00	0.00	00-00-00	0.00



Typische Client/Server Anwendung

In den meisten Fällen, z.B. bei allen relationalen und nichtrelationalen Datenbanken, wird die Datenhaltung durch einen eigenen Prozess gesteuert (z.B. DB2 oder Oracle Datenbankprozess). Dieser verfügt über einen eigenen virtuellen Adressenraum, und kann deshalb leicht auf einem getrennten Rechner laufen.



Business- und Präsentationslogik

Ein sauber strukturiertes CICS Programm besteht aus zwei Teilen: Business Logik und Präsentations-Logik.

Business Logik ist der Teil, in dem Berechnungen erfolgen und Daten in einer Datenbank gelesen/geschrieben werden.

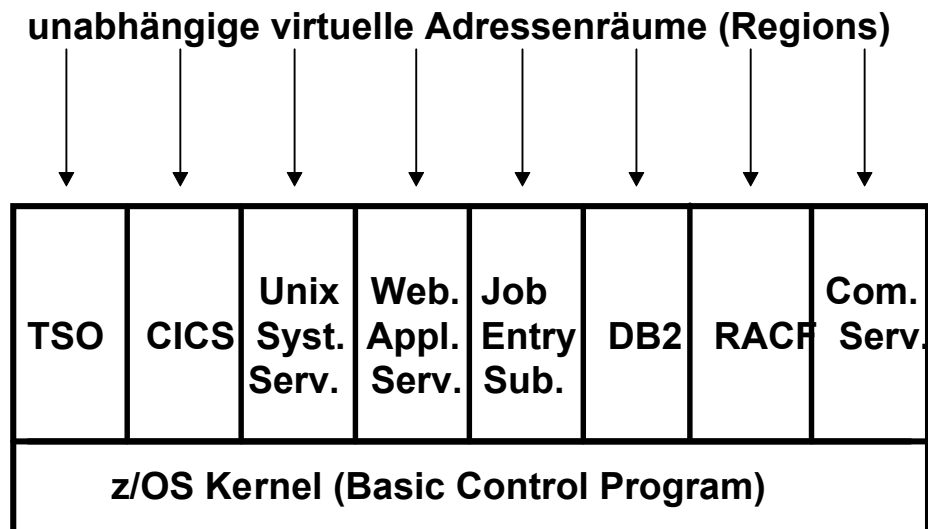
Präsentations- Logik ist der Teil, in dem die Ergebnisse der Berechnungen so aufgearbeitet werden, daß sie dem Benutzer in einer ansprechenden Art auf dem Bildschirm dargestellt werden können.

Business Logik wird in Sprachen wie C, C++, COBOL, PL/1, Java usw. geschrieben.

Für die Präsentations - Logik gibt es viele Möglichkeiten. Die modernste Alternative benutzt Java Server Pages und einen Web Application Server um den Bildschirminhalt innerhalb eines Web Browsers darzustellen.

Die älteste (und einfachste) Alternative verwendet das CICS BMS (Basic Mapping Support) Subsystem. BMS Programme werden in der BMS Sprache geschrieben.

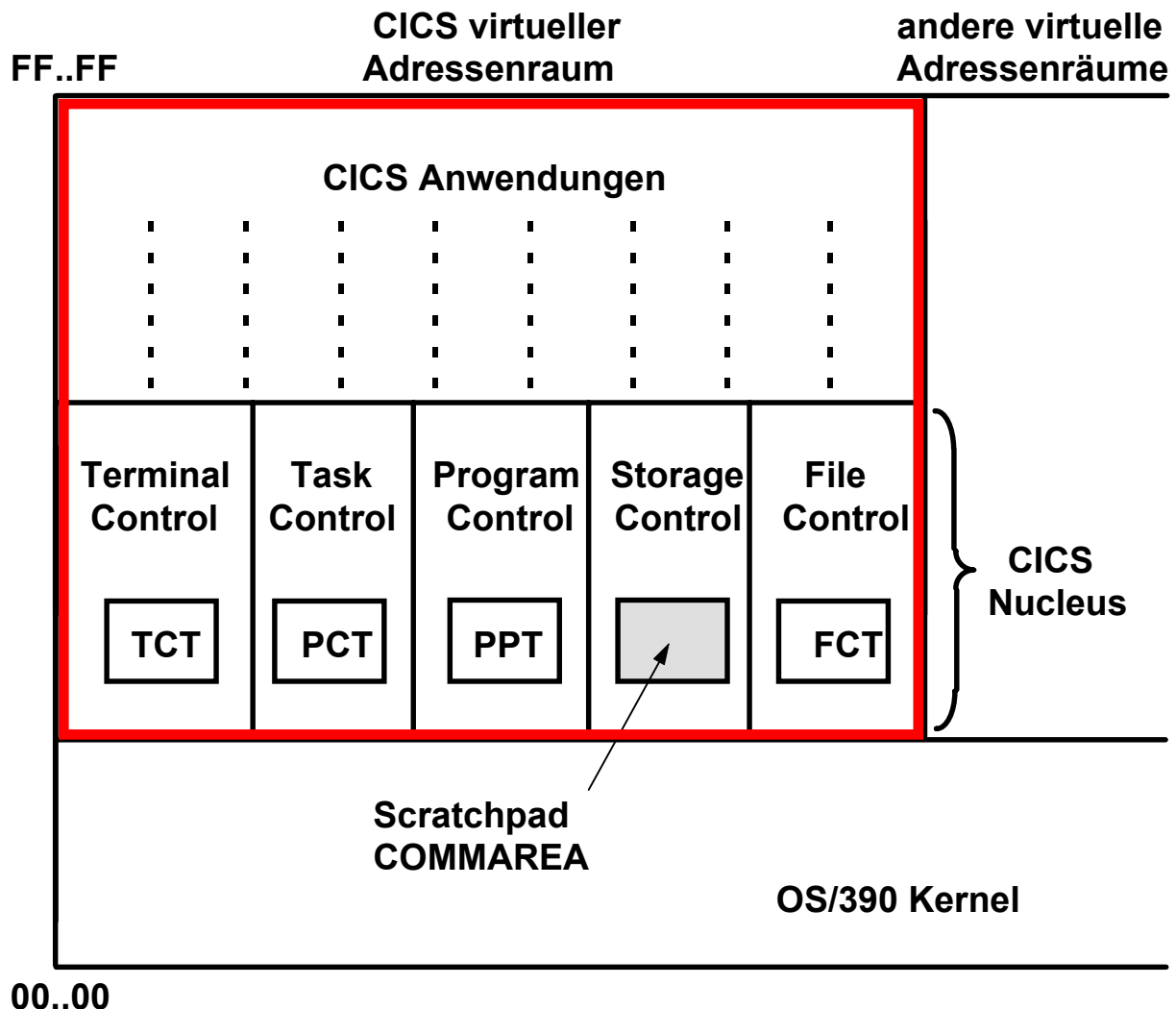
OS/390 Grundstruktur



Der z/OS Kernel unterstützt eine Vielzahl von virtuellen Adressenräumen, die im z/OS Jargon als „Regions“ bezeichnet werden.

Manche Regions beherbergen Subsysteme, die Teil des Betriebssystems sind, aber im Benutzerstatus laufen. Einige der (zahlreichen) Subsysteme sind:

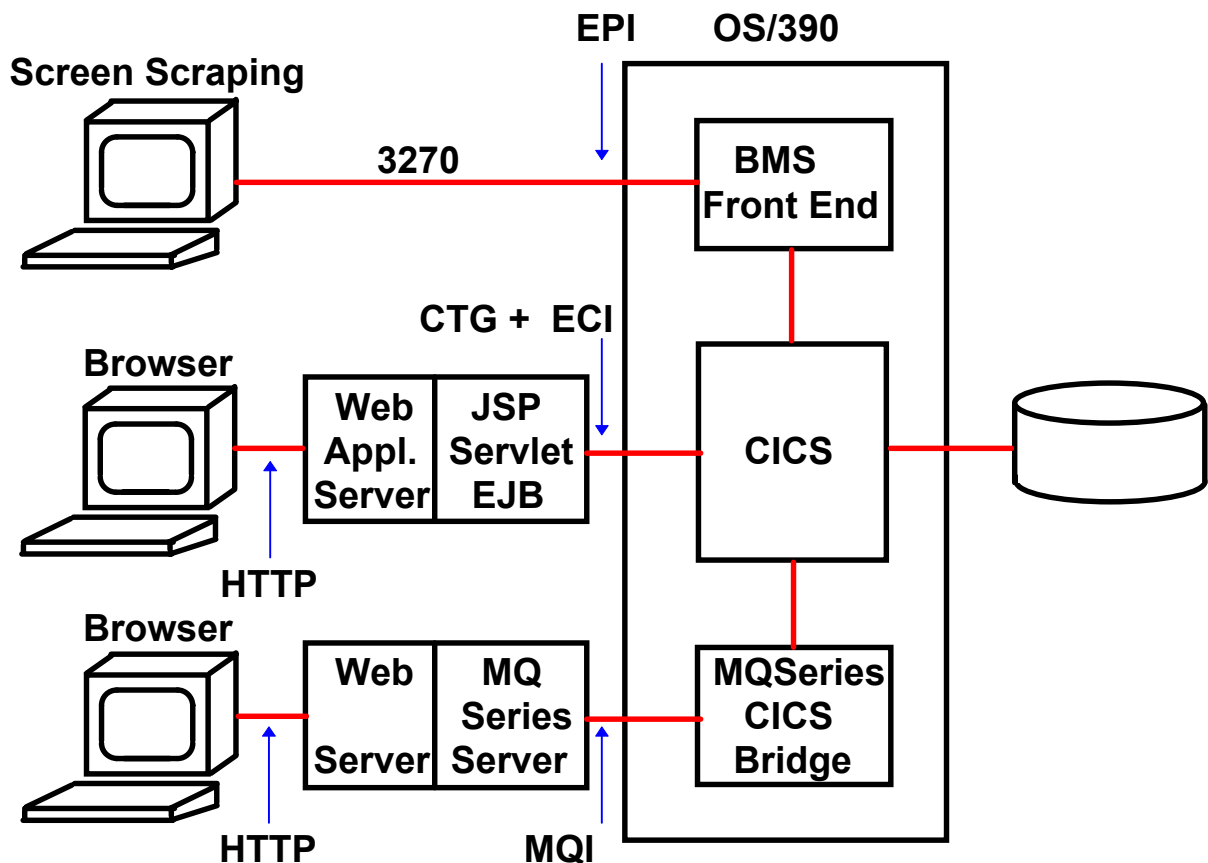
- **JES** Job entry Subsystem
- **CICS** Transaktionsverarbeitung
- **TSO** Shell, Entwicklungsumgebung
- **USS** Unix kompatible Shell, Entwicklungsumgebung
- **WAS** WebSphere Web Application Server
- **DB2** relationale Datenbank
- **RACF** Sicherheitssystem
- **Communications Server**



CICS Komponenten

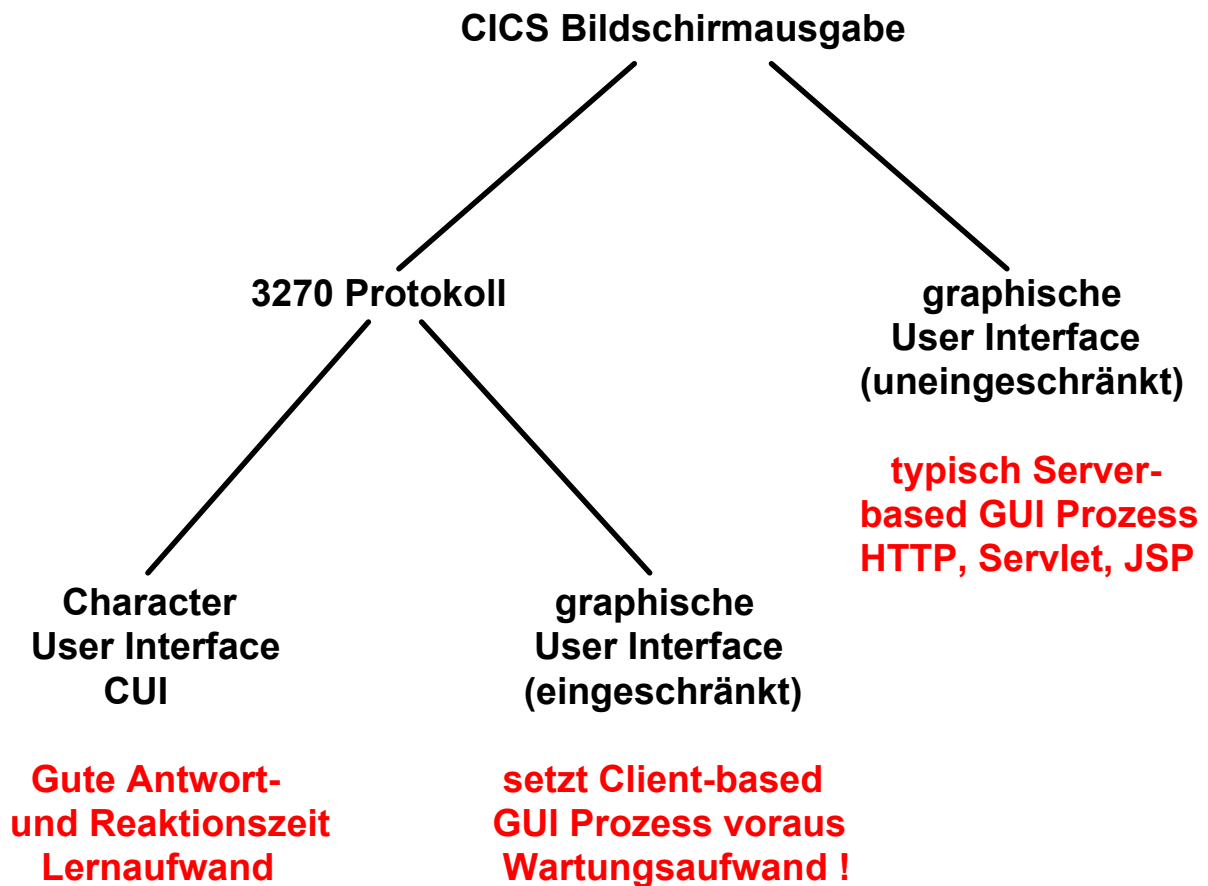
CICS läuft als Stapelverarbeitungsjob in einem einzigen virtuellen Adressenraum (Region in z/OS Terminologie). CICS Anwendungsprogramme laufen „run to completion“; Interaktivität wird programmtechnisch gewährleistet, indem ihre maximale Ausführungszeit eine vorgegebene Grenze nicht überschreitet.

Die CICS Nucleus Komponenten (Terminal Control, Task Control, Program Control, Storage Control and File Control) nutzen den gleichen virtuellen Adressenraum wie alle Anwendungen. Jede Nucleus Komponente hat eine zugeordnete Tabelle: TCT, PCT, PPT, FCT. Über COMMAREA werden Sessions eingerichtet: Der State einer Transaktion ist für die Folgetransaktion verfügbar.



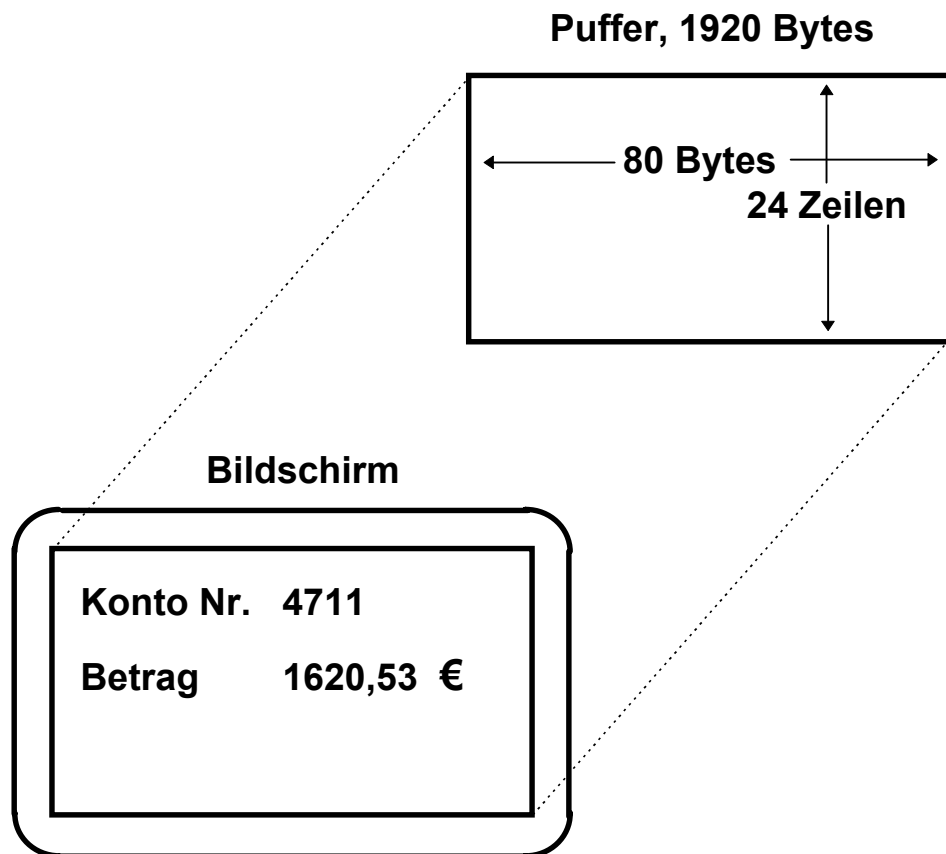
CICS Klienten Anbindung

- EPI** Die BMS Maps werden weiter verwendet. Keine Änderung der Information, die auf dem Bildschirm wiedergegeben wird. Die Darstellung der Information kann geändert werden.
- ECI** Die Presentation Service Komponente von CICS (BMS) wird nicht genutzt. Direkter Zugriff auf COMMAREA.
- MQSeries** Asynchrone Übertragung durch Message oriented Middleware



Alternativen der Bildschirmausgabe

CUI	Character User Interface
GUI	Graphical User Interface



3270 Bildschirmdarstellung

Es wird eine Nachricht übertragen, die einen $24 \times 80 = 1960$ Byte großen Bildschirm Puffer mit Character Daten füllt und vom CICS Terminal interpretiert wird.

Der Pufferinhalt wird als 24 Zeilen mit 80 Zeichen/Zeile wiedergegeben. Jede der 1960 Byte Positionen kann einzeln adressiert werden. In der Regel werden Gruppen von Bytes (Felder) adressiert.

Dargestellt sind 4 Felder: Konto Nr.
 Betrag
 4711
 1620,53 €

3270 Protokoll

CICS entstand 1968-1969, benutzte IBM 2740 Typewriter Terminals. Die IBM 3270 Display Terminal Familie wurde in 1972 eingeführt („nicht-intelligente Terminals“). Sie nutzt das 3270 Protokoll für die Datenübertragung Terminal - CICS Server.

Arbeitet mit einem zeichenorientierten Screen, bestehend aus 24 Zeilen mit je 80 α/n Zeichen.

Jede der $24 * 80 = 1920$ Positionen ist vom Anwendungsprogramm des CICS Servers individuell adressierbar (gewisse Ähnlichkeit mit dem X-Window-Protokoll).

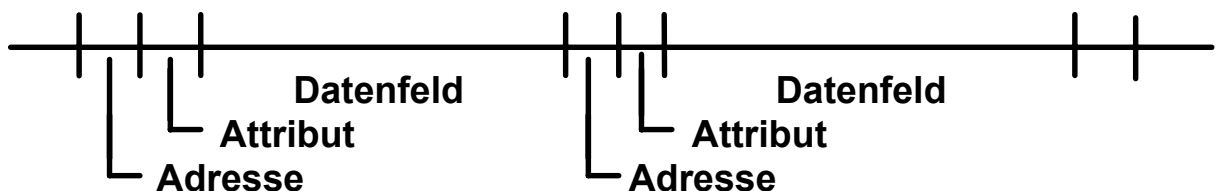
Normalerweise werden „Felder“ angesprochen. Ein Feld ist eine Folge von Zeichenpositionen. Felder können gelesen und geschrieben werden.

Eine CICS-Utility, Basic Mapping Support (BMS) erleichtert dem Anwendungsprogrammierer die Entwicklung von Screens.

Auf heutigen Arbeitsplatzrechnern mit Hilfe der 3270 Emulation implementiert.

3270 Bildschirm Datenausgabe

Das Anwendungsprogramm erzeugt die Datenausgabe an den Bildschirm im Format



Das Datenfeld enthält eine variable Anzahl von α/n Zeichen, welche auf dem Bildschirm in einem Feld wiedergegeben werden.

Das Attributfeld (3 Bytes bei BMS/CICS) enthält Steuerzeichen, welche Information über die Art der Wiedergabe des folgenden Datenfeldes enthalten, z.B. Darstellung in roter Farbe, blinkender Cursor, Font, andere...

Das 3270 Protokoll verwendet eine Untermenge der 256 Zeichen des ASCII oder EBCDIC Zeichensatzes zur Datenwiedergabe auf dem Bildschirm. Die restlichen Zeichen werden als Steuerzeichen für Steuerungszwecke eingesetzt.

Der 3270 Bildschirm besteht aus 24 α/n Zeilen mit je 80 fixed Font-Width Zeichenpositionen pro Zeile. Das Adressenfeld kennzeichnet eine der $24 \times 80 = 1920$ Zeichenpositionen auf dem Bildschirm.

ACCOUNTS MENU

TO SEARCH BY NAME, ENTER SURNAME AND IF REQUIRED, FIRST NAME

SURNAME : (1 TO 18 ALPHABETIC CHRS)
FIRST NAME : (1 TO 12 ALPHABETIC CHRS OPTIONAL)

TO PROCESS AN ACCOUNT, ENTER REQUEST TYPE AND ACCOUNT NUMBER

REQUEST TYPE: **A** (D-DISPLAY, A-ADD, M-MODIFY, X-DELETE, P-PRINT)
ACCOUNT : **26004** (10000 TO 79999)
PRINTER ID : (1 TO 4 CHARACTERS (REQUIRED FOR PRINT REQUEST))

ACCT	SURNAME	FIRST	MI	TTL	ADDRESS	ST	LIMIT
26001	Meier	Rolf	A	MR	Ritterstr. 13	N	1000.00
26002	Meier	Steffie	G	MRS	Wilhelmstr. 24	N	1000.00
26003	Meier	Tobias	A	MR	Nikolaistr. 23	N	1000.00

ENTER DATA AND PRESS ENTER FOR SEARCH OR ACCOUNT REQUEST OR PRESS CLEAR TO EXIT

Beispiel eines CICS Basic Mapping Support (BMS) α/n Bildschirms

BCRVM1 - A - 3270 - 24x80

File Edit Transfer Appearance Communication Assist Print Help

File Distribute Inquire Options Help

100 Search Request

To request a search, type the search data and press Enter.

Lines 1 to 14 of 32

Corporate and Personal Directories

Name _____
 Search Locs/Nodes/Dirs . RTP, NC
 Node _____
 User ID _____
 Extension _____
 Job Responsibilities _____
 Department _____
 any field (Name Value) . _____ ex: DIV 1
 *any field (Name Value) . _____ ex: MGR Y

Departments Directory

Department Title/Number . _____
 Search Locs/Nodes/Dirs . -BOCA

(C) Copyright IBM Corporation 1988, 1992. All rights reserved.
 Command ==> _____

F1=Help F2=Set 2 F3=Exit F4=Profile F5=Refresh F6=Fuzzy search
 F7=Backward F8=Forward F9=Command F10=Actions F11=Dist list F12=Cancel


MA* a 8/32

PF1	PF2	PF3	PF4	PF5	PF6	Enter	PA1	Attn	Insert	NewLine
PF7	PF8	PF9	PF10	PF11	PF12	Clear	PA2	SysReq	Delete	NextPad

AT* Host Session. Connected to IP=ddcntsv4.demopkg.ibm.com. Port=2342

File Host Connection Options Service View Help




IBM



Company Directory

Person's name (last, first)

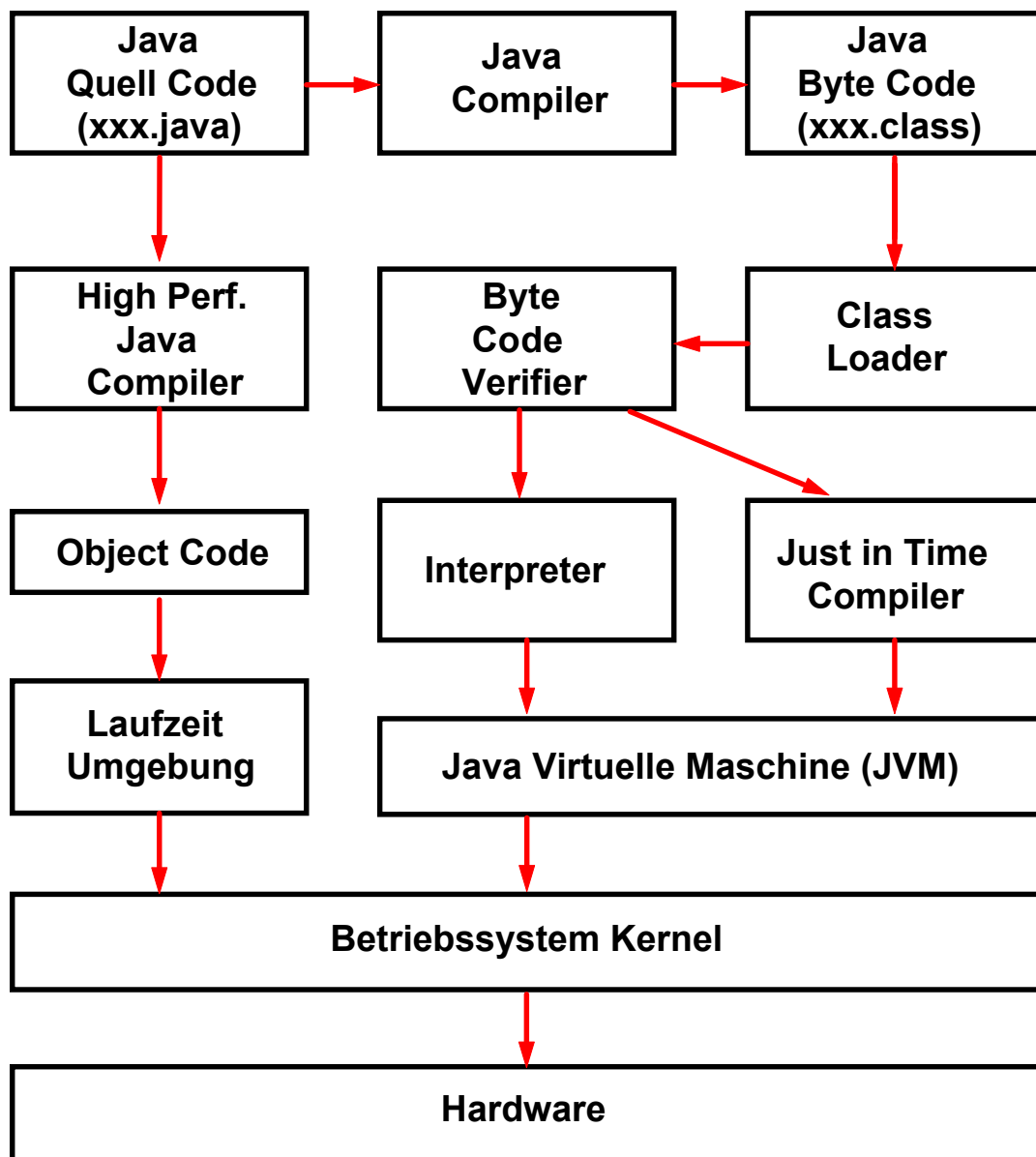
City location

 Enter  Quit  Help

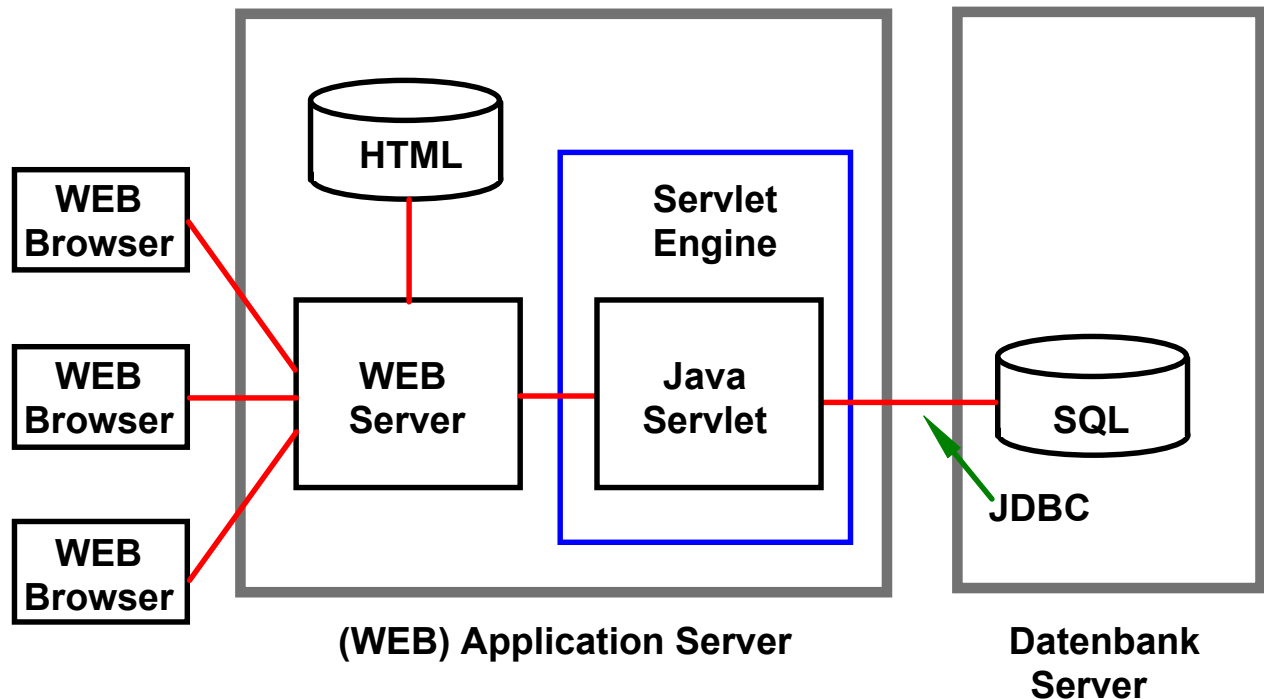
MA* a Ovr R8 C45 ID: call

PF1	PF2	PF3	PF4	PF5	PF6	Enter	Reset	Clear	Ins	More..
PF7	PF8	PF9	PF10	PF11	PF12	ErsInp	ErsEOF	PA1	PA2	PA3

Unsigned Java Applet Window



Java Virtuelle Maschine



Dynamischer WEB Seiten Inhalt (2)

Im Gegensatz zu CGI erfordert das Java Servlet nur light weight Context Switches. Daher deutlich besseres Leistungsverhalten.

Servlets sind Java Klassen und benötigen eine eigene Laufzeitumgebung (als Servlet Engine bezeichnet). Sie verfügen über alle Java API's, einschließlich JDBC (Java Data Base Connectivity).

Java Server Pages (JSP) sind eine Erweiterung der Servlet API. Verwenden in Java geschriebene XML- ähnliche Tags und Scriptlets.

Eine gute Einführung ist zu finden unter <http://www.apl.jhu.edu/~hall/java/Servlet-Tutorial/>

Servlet Container

Servlets laufen in speziellen Servlet-spezifischen Containern, die auch als Servlet Engines bezeichnet werden. Sie verbessern u.a. die Servlet-Ausführungszeit. Servlet Container haben keine Transactions-, Persistence- und Sicherheitseigenschaften. Ein Servlet Container ist ein Programm, das Requests für Servlets und Java Server Pages (JSP) behandelt. Der Servlet Container ist verantwortlich für:

Aufgaben

- **Erstellung von Servlet-Instanzen,**
- **Initialisierung von Servlets,**
- **Dispatching von Requests,**
- **Verwaltung des Servlet-Kontextes für die Nutzung durch die Web-Anwendungen.**

Servlet Beispiele

Auf unserem OS/390 Server an der Uni Leipzig sind mehrere Lehrbeispiele für Servlets installiert. Der Quellcode dazu kann eingesehen werden. Sie können auf dem OS/390 WebSphere Server <http://jedi.informatik.uni-leipzig.de> mit den folgenden URLs adressiert werden:

**<http://jedi.informatik.uni-leipzig.de/servlet/sm390.SMJDBCTestServlet>
<http://jedi.informatik.uni-leipzig.de/servlet/sm390.GuestBookServlet>
<http://jedi.informatik.uni-leipzig.de/IBMWebAS/samples>**

**Teilweise erfolgt ein Zugriff auf die OS/390 DB2 Datenbank
Für die Erstellung dieser Beispiele existieren ausführliche
Tutorials.**

Basic Servlets

Try our basic servlets for common tasks:

- [Echo a request \(RequestInfo\)](#)
- [Query a database \(JDBC\)](#)
- [Process form input \(FormProcessing\)](#)
- [Display form input \(FormDisplay\)](#)

Travel

Are you looking for an exciting vacation? Xtreme Adventures knows which site you were visiting and offers you trips you'll love. And while you're browsing, you can exchange messages with other potential travelers.

WOM Bank



Open accounts, deposit, transfer, or withdraw money. WOM Bank connects to a DB2 database to authenticate customers and maintain transaction information.

Ticket Server

TicketCentral searches pre-loaded DB2 database tables to find available seats at your favorite events. Order your tickets and purchase them with your credit card.

<http://jedi.informatik.uni-leipzig.de/BMWWebAS/samples/>

Einfache Java Server Page

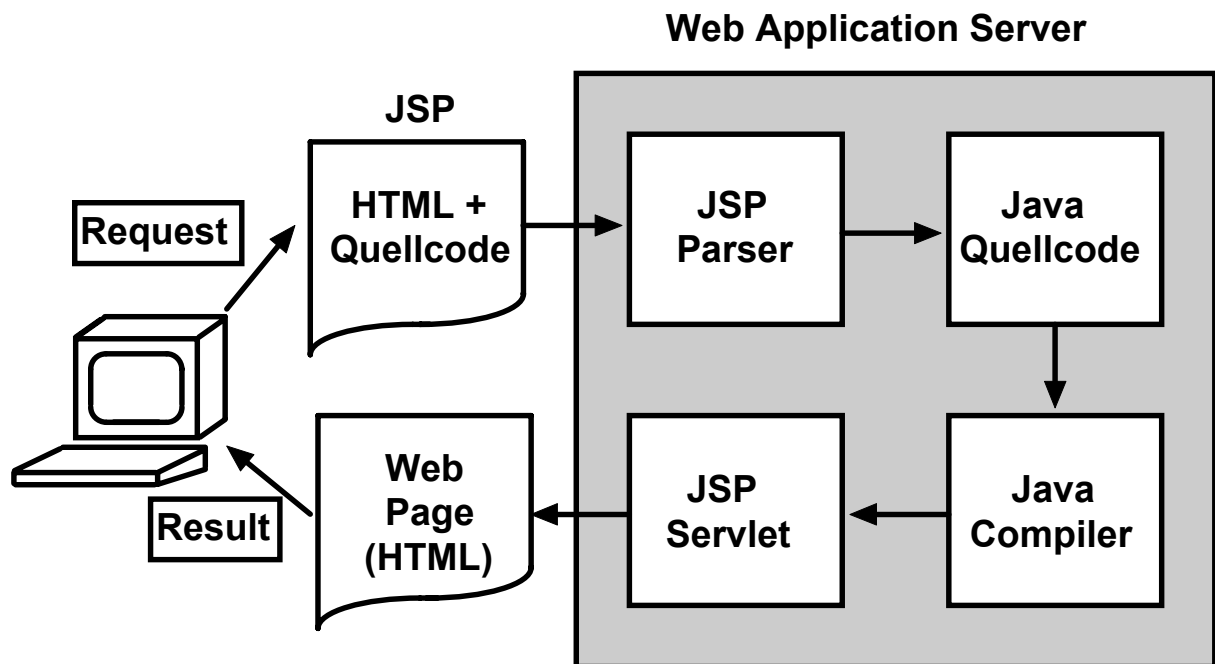
Put the following text in a file with .jsp extension (let us call it hello.jsp), place it in your JSP directory, and view it in a browser.

```
<HTML>
<BODY>
Hello! The time is now <%= new java.util.Date() %>
</BODY>
</HTML>
```

Notice that each time you reload the page in the browser, it comes up with the current time.

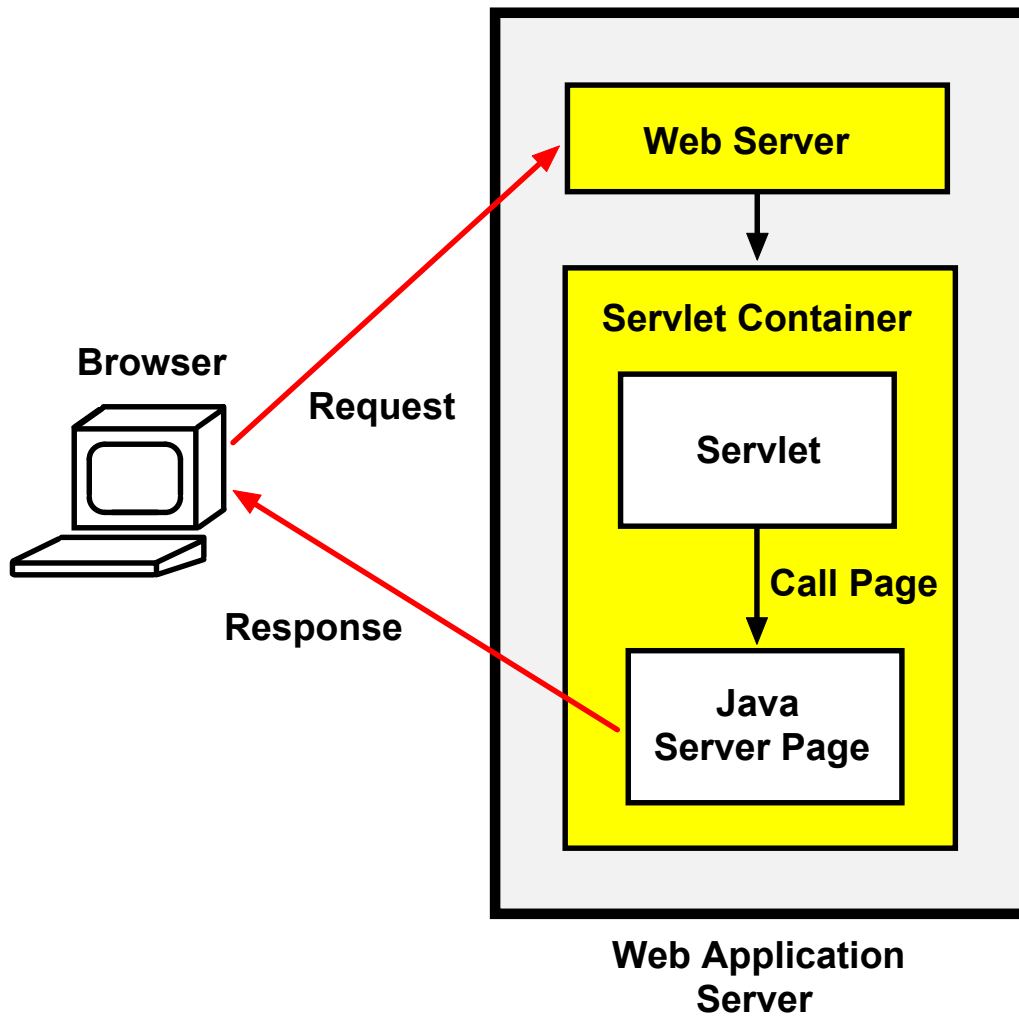
The character sequences `<%=` and `%>` enclose Java expressions, which are evaluated at run time.

This is what makes it possible to use JSP to generate dynamic HTML pages that change in response to user actions or vary from user to user.

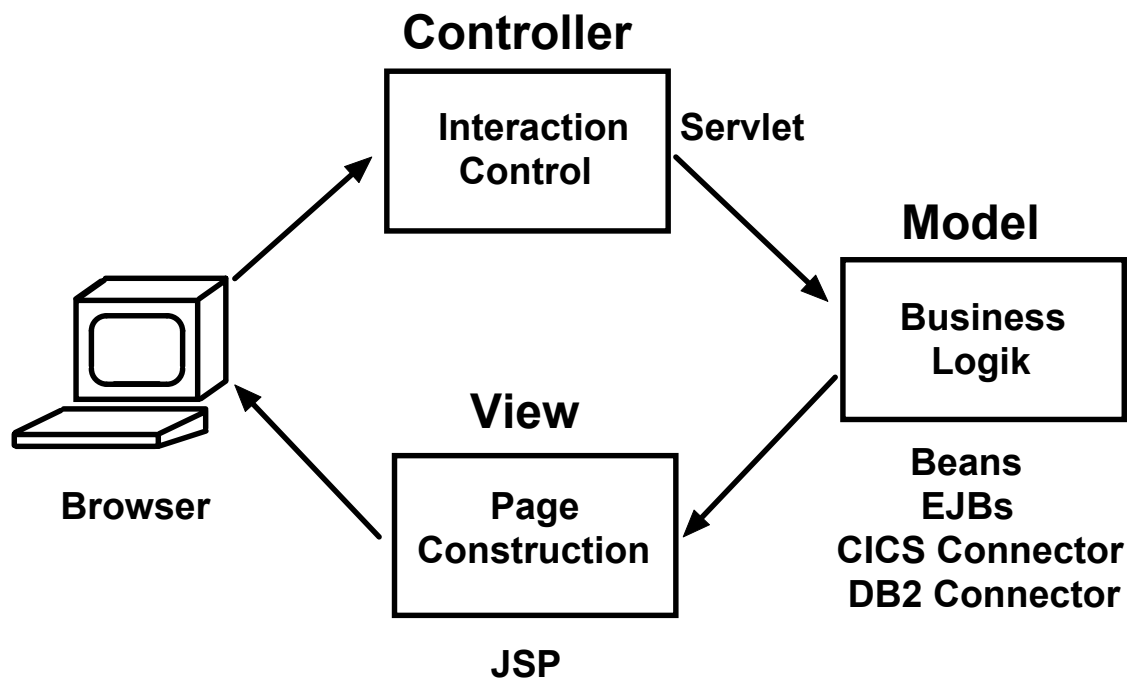


Java Server Page (JSP)

1. Der Web Browser sendet eine Request an die JSP Seite.
2. Die JSP Engine parses den Inhalt der JSP File. Sie erstellt temporären Servlet Quellcode basierend auf dem Inhalt der JSP.
3. Der Servlet Quellcode wird durch den Java Compiler in eine Servlet Class File übersetzt.
4. Das Servlet wird instantiated. Die init and service Methoden des Servlets werden aufgerufen; die Servlet Logic wird ausgeführt.
5. Die Kombination von statischem HTML, kombiniert mit den dynamischen Elementen spezifiziert in der ursprünglichen JSP Definition, geht an den Web Browser zurück durch den Output Stream des Servlet Response Objektes.



Interaktion Servlet - JSP



Model/View/Controller Triade (MVC)

Das „Modell“ ist ein Anwendungsobjekt und kapselt die Business Logik. Der „View“ ist die Screen Darstellung dieses Objektes. Der „Controller“ definiert, wie die Benutzerschnittstelle auf Benutzereingaben reagiert.

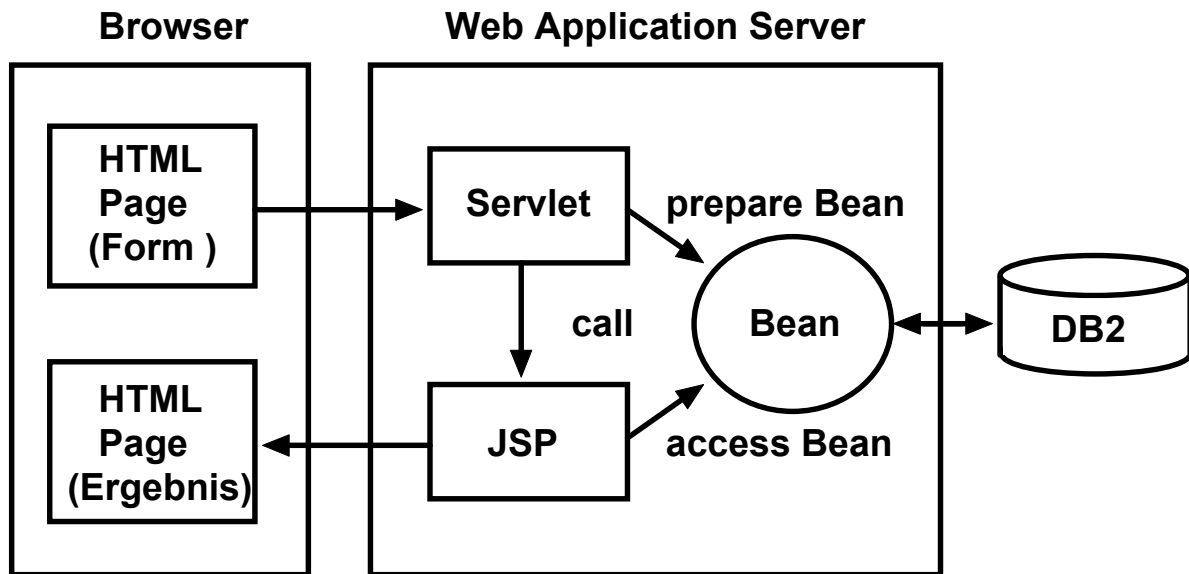
Beans oder Enterprise Java Beans, plus häufig CICS, IMS Programme, oder Stored Procedures, sind das „Modell“ (= Business Logik).

Java Server Pages und View Beans sind der „View“

Das Servlet ist der „Controller“

MVC entkoppelt Modell und View zur Verbesserung von Flexibilität und Re-Use. Der Entwickler der Browser Darstellung arbeitet nur mit der Java Server Page.

Zentrisches Programmier Modell. Die gesamte Anwendungslogik (EJB, Servlet, JSP) läuft auf dem Server. Der Klient (*thin client*) braucht nur einen Browser.



Java Server Pages (JSP)

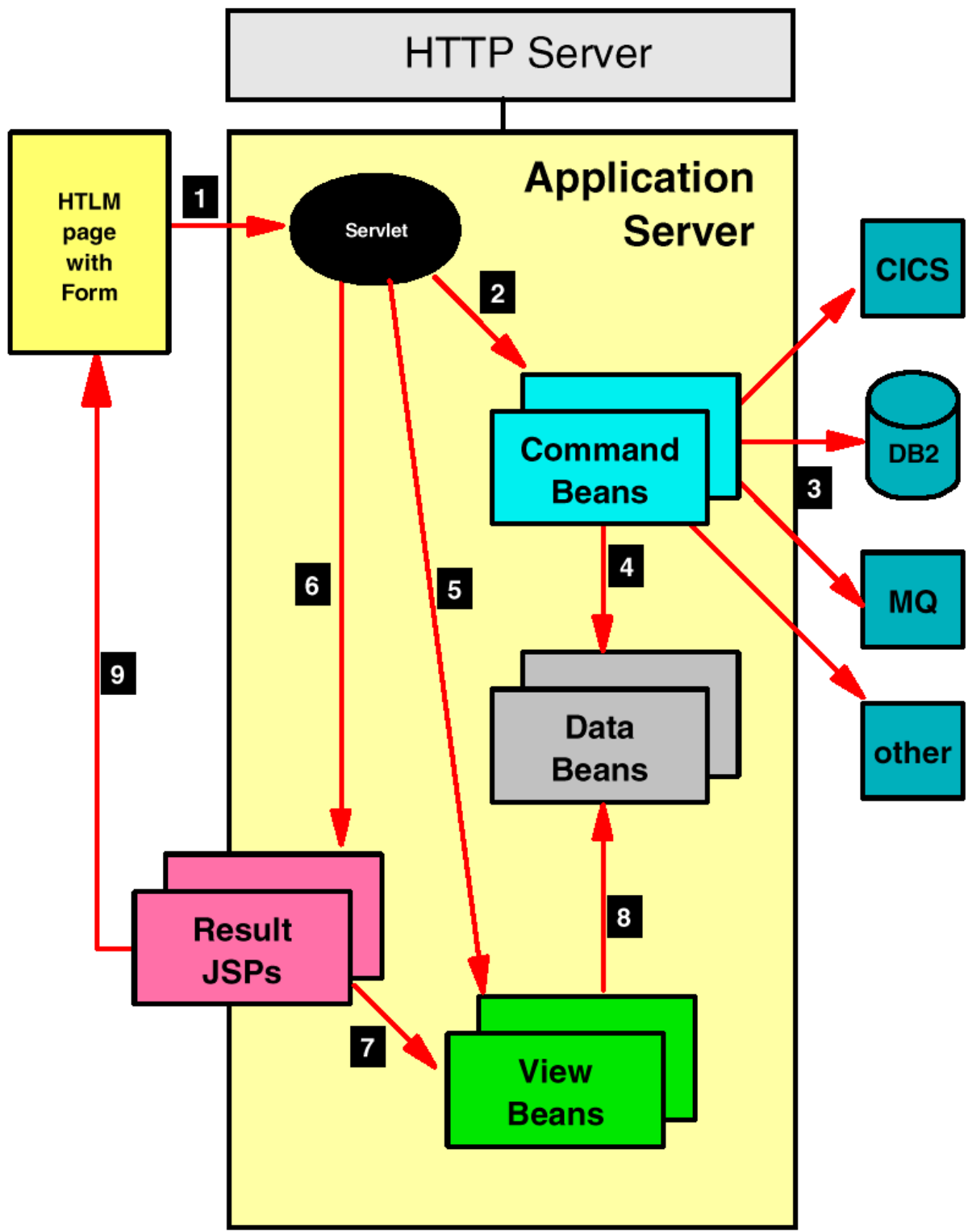
Ein Servlet ist ein Java Programm, das Bildschirm Output in der Form einer HTML Datei produziert.

Eine JAVAServerPage ist eine HTML Seite mit zusätzlichen JSP Tags.

Wird eine JSP Seite aufgerufen, so kompiliert sie ein JSP Übersetzer in ein Servlet.

In der Praxis: Servlets und JSP werden von verschiedenen Leuten erstellt (Model-View-Controller Ansatz). Eine JSP ist zwar eine vollwertige Java Komponente, aber der Java Code Anteil innerhalb der JSP wird in der Regel auf ein Minimum reduziert.

Es existieren (wie für HTML Seiten) spezielle Werkzeuge für das Erstellen von JSP's, die das Hand-coding von HTML Statements automatisieren.

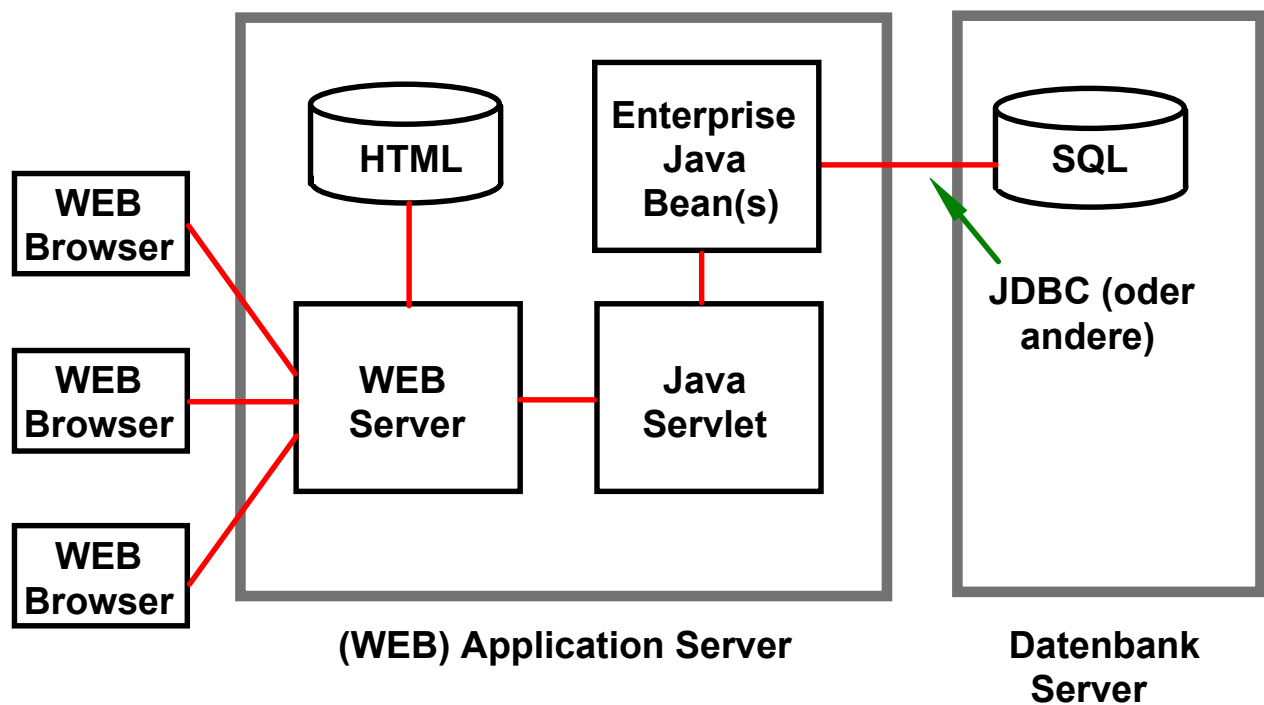


Architektur einer JSP Web Anwendung

Model - View - Controller Ansatz

- 1. HTML page:** static or dynamic HTML page, created from a previous step, contains one or multiple forms that invoke a servlet for processing of the next interaction.
- 2. Servlet** gets control from the Application Server to perform validation and control of flow; sets up and calls command beans that perform the business logic.
- 3. Command beans** control the processing of the business logic; logic may be imbedded in the command bean, or it can be delegated to back-end or enterprise systems, such as relational databases, transactions systems (CICS, MQSeries, IMS, and so forth); command bean may perform one specific function or it may contain many methods, each for a specific task (task wrappers). Command beans invoke database and transaction systems using „connectors“.
- 4. Results** of command beans (or back-end systems) processing are stored in data beans. Data beans could contain an SQL result or a CICS communication area.
- 5. View beans** provide the contract between the output producing JSPs and the data beans that contain the dynamic data to be displayed in the output; servlet initializes the view beans and registers them with the request block so that the JSPs can find them.
- 6. Servlet** calls a JSP for output processing and formatting depending on the results of the command beans; JSPs generate the output for the browser.
- 7. JSP** use tags to declare the view beans and get access to all the dynamic data that must be displayed in the output.
- 8. View beans** contain one or multiple data beans and provides tailored methods so that the JSP has access to the data stored in the data beans; data beans may not provide the necessary methods for a JSP to access the data.
- 9. JSP** assembles the output and sends it back to the browser as an HTML page with dynamic data; in many cases, that output again contains form(s) to enable the user to continue the dialog with the application.

Servlet is the controller
Command beans provide the model
JSP is the view



Dynamischer WEB Seiten Inhalt (3)

Im einfachsten Fall enthält das Java Servlet die Anwendungslogik. In komplexeren Fällen lohnt es sich, die Anwendung in Komponentenform zu implementieren. Java Beans implementieren das Java Komponentenmodell.

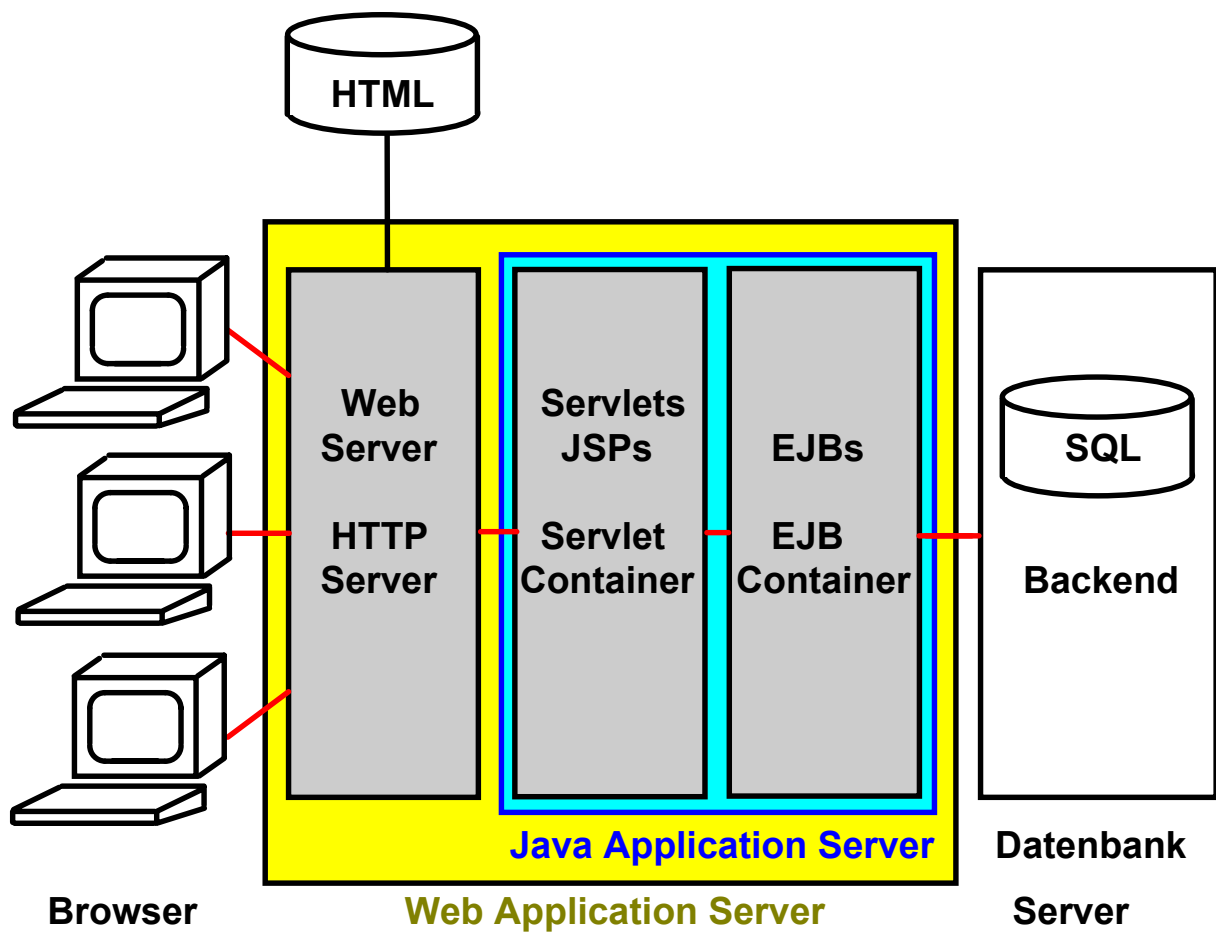
Enterprise Java Beans sind Java Beans mit zusätzlicher Funktionalität, besonders Transaktionseigenschaften (ACID), Persistenz und Sicherheit.

Enterprise Java Beans (EJB)

Java basiertes Server Komponentenmodell, März 1998. Final Release der Version 1.1 der EJB Spezifikation erfolgte im Dezember 1999

- EJB Komponenten sind serverseitige Komponenten, die ausschließlich in Java geschrieben sind. EJB Komponenten enthalten nur Business Logik, keine Präsentationslogik und keine Systemfunktionen.
- EJBs sind in einen „Container“ eingebettet (Laufzeitumgebung). Die EJB Architektur ist inhärent transaktionsorientiert, distributed, portierbar, multi-tier, skalierbar und sicher. Diese und weitere Systemfunktionen wie, Life-cycle management, threading und Persistenz werden automatisch für die EJB Komponente von dem EJB Container zur Verfügung gestellt.
- EJB Komponenten werden deklarativ (über einen *Deployment Descriptor*) zur Laufzeit angepaßt. Die Anpassung bezieht sich auf Transaktionsverhalten, Sicherheitseigenschaften, life-cycle und state management, Persistenz, usw.
- Die permanente Speicherung eines Objektes auf einem Plattenspeicher wird als Persistenz bezeichnet. Konzeptuell können Objekte in einer Objektdatenbank (z.B. POET) gespeichert werden. In der Praxis werden SQL (oder IMS oder VSAM) Daten als Objekte gekapselt; der Zugriff erfolgt z.B. über eine JDBC (Java Data Base Connectivity), SQLJ oder DB2Connect Schnittstelle.
- Interoperabilität von EJBs und CORBA konformen Objekten ist möglich.

J2EE (Java 2 Platform, Enterprise Edition) kombiniert Technologien wie Servlet, JSP, EJB, JMS, Konnektoren und den JDK.

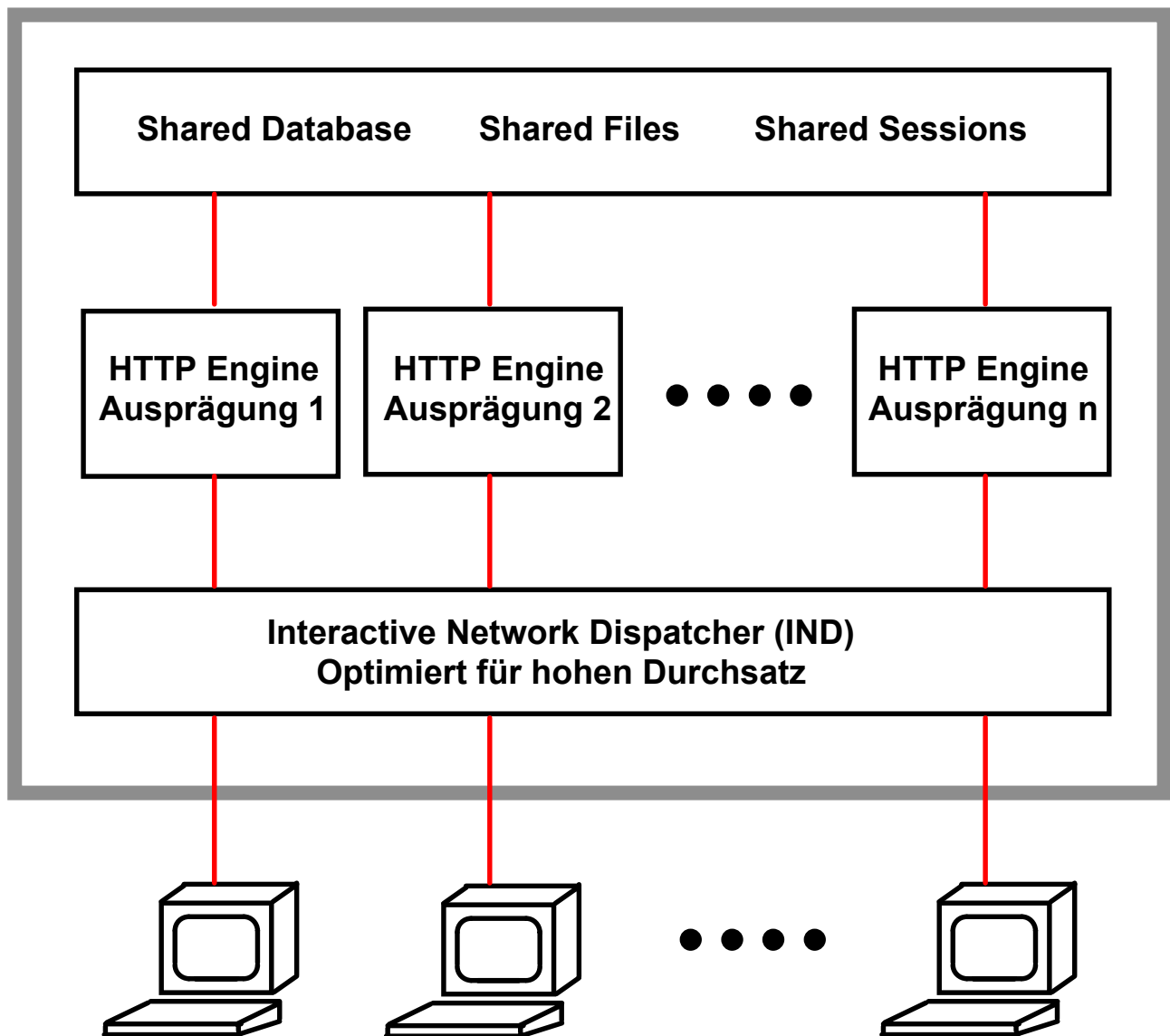


Führende Vertreter

BEA WebLogic
 IBM WebSphere
 SAP Netweaver

Weitere Implementierungen

Oracle Web Application Server
 Sun One Application Server (bisher iPlanet)
 Jboss (Open source)



HTTP Server

Der HTTP Server behandelt Anforderungen für (meistens) statische Ressourcen: HTML Seiten, GIF Dateien und CGI Aufrufe

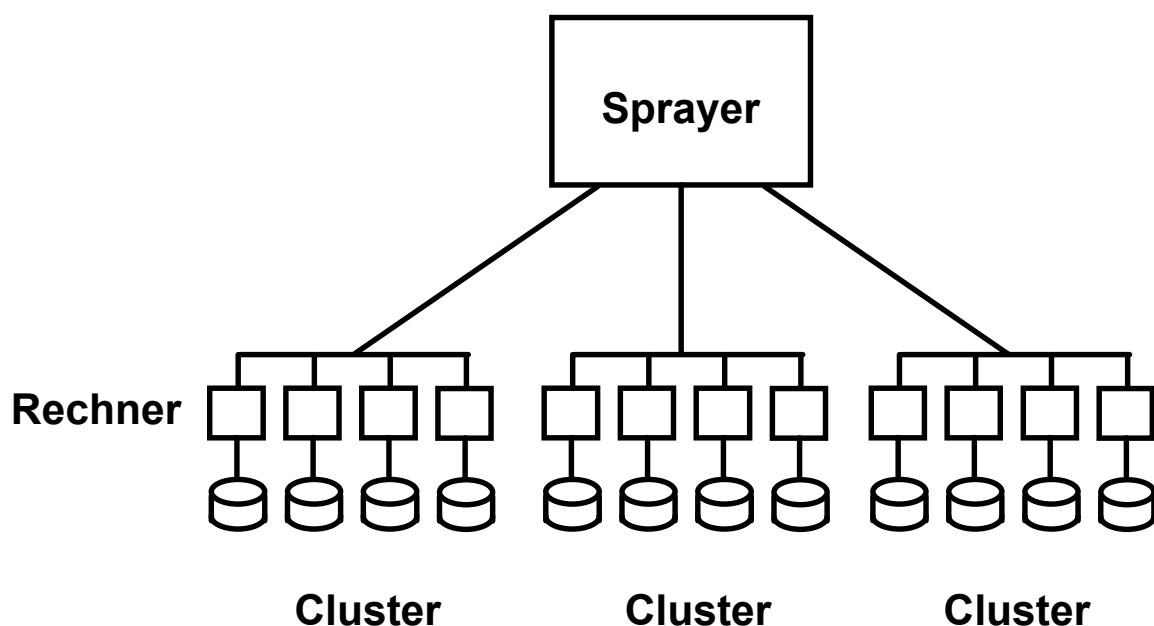
Hohes Verkehrsaufkommen, kurzlebige Anforderungen

Skalierung durch mehrfache Web Server Engines

Der Interactive Network Dispatcher (auch als „Sprayer“ oder Load Balancer bezeichnet) verteilt die Anforderungen auf die einzelnen Web Engines

www.google.com

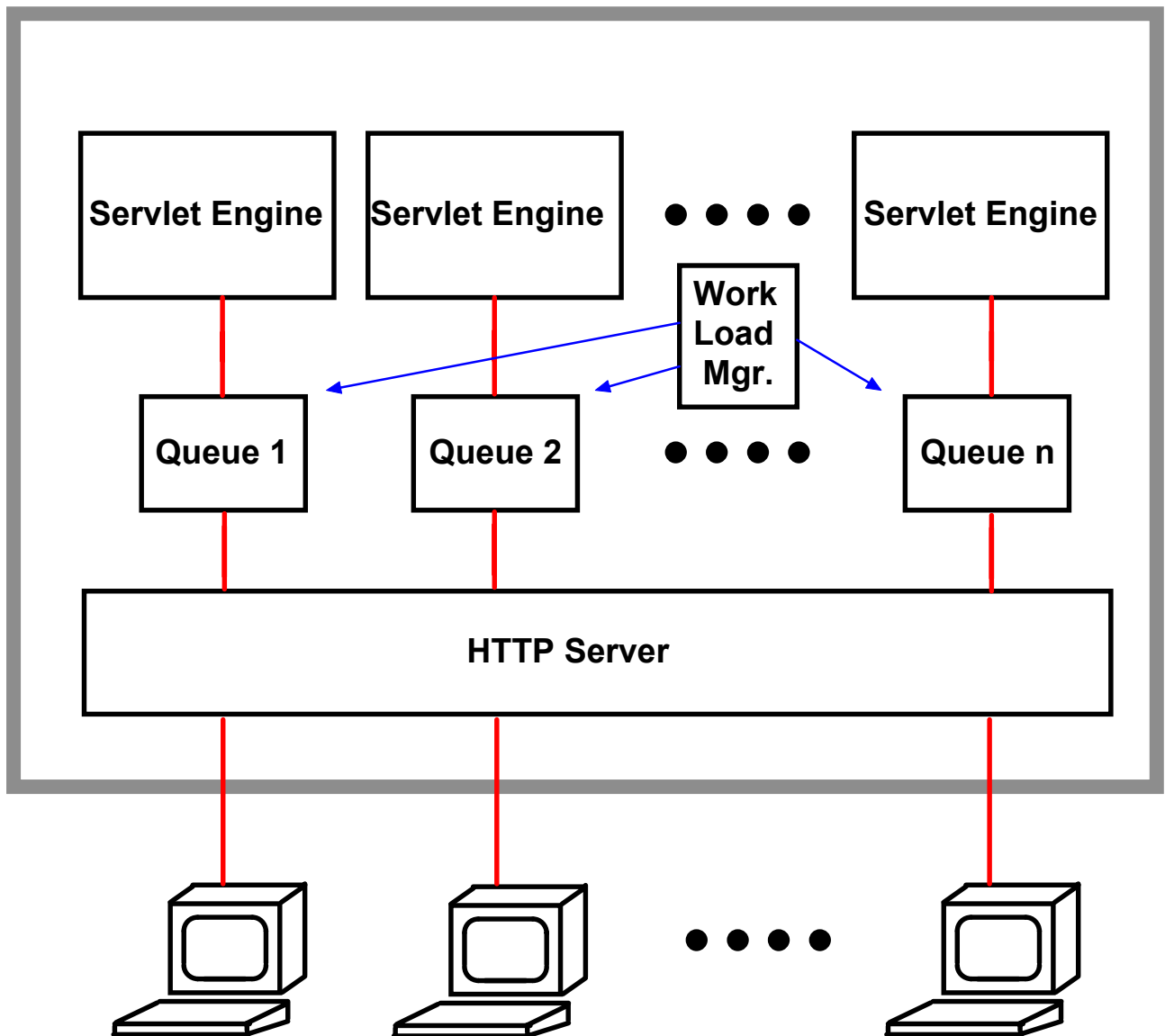
Google unterhält in 5 Rechenzentren ca. 10 000 Rechner. Mehrere Cluster in jedem Rechenzentrum. Jeder Cluster dupliziert den ganzen Google Datenbestand.



je 30 - 50 Rechner pro Cluster
je 3 - 5 Tbyte pro Cluster
Kopie aller Daten auf jedem Cluster

Sprayer verteilt Anfragen auf die einzelnen Cluster. Jeder Cluster ist in der Lage, jede Art von Anfrage zu bearbeiten.

Einfacher Workload Algorithmus.



Anwendungs-Queues und mehrfache Prozesse

Zur Verbesserung des Leistungsverhaltens laufen mehrere Servlet Prozesse auf dem Applikations-Server. Anforderungen von dem Web Server gehen (je nach Policy) zu einer von mehreren Queues. Jede Queue wird von mehreren Java Prozessen bedient.

Die Queue Policy bestimmt

- URLs, die von der Queue bedient werden
- Anzahl der Prozesse für diese Queue
- Sicherheitsumgebung

Der Administrator legt die Anzahl und die Policies jeder Queue fest