UNIVERSITÄT LEIPZIG

Mainframe Internet Integration

Prof. Dr. Martin Bogdan
Prof. Dr.-Ing. Wilhelm G. Spruth

SS2013

WebSphere Application Server Teil 3

Load Balancing

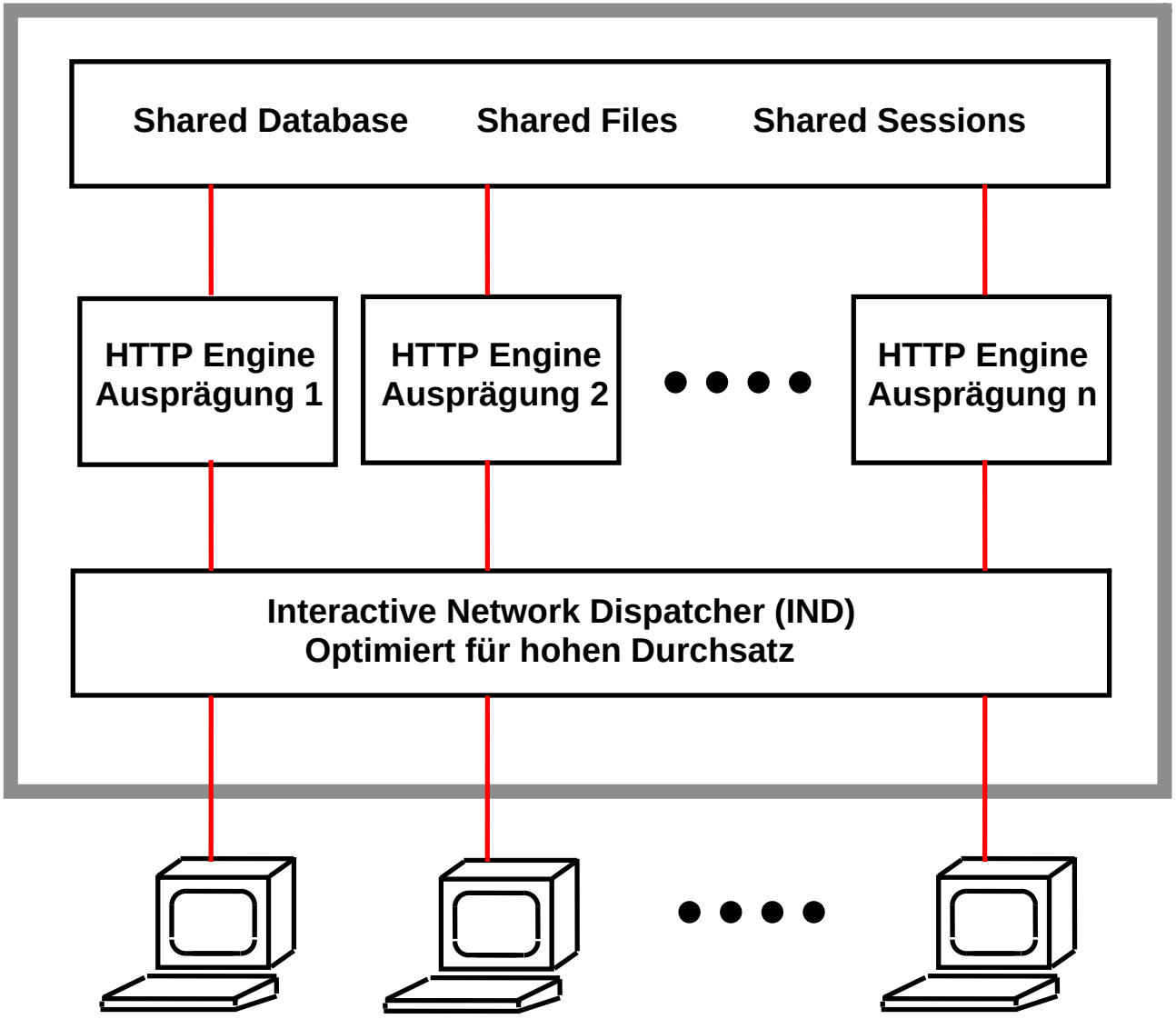
Skalierbarkeit

Die Anzahl der Web Browser Zugriffe wachsen von Jahr zu Jahr. Hilfreich ist, dass die große Mehrzahl aller Zugriffe sich auf statische Web Seiten bezieht, die einen nur relativ geringen Verarbeitungsaufwand erfordern. Faustformel: Rund 80 % aller Zugriffe sind statisch.

Dennoch reicht ein einziger physischer Server häufig nicht aus, um die Anfragen mit einer vertretbaren Antwortzeit abzuarbeiten.

Erschwerend kommt hinzu, dass die Arbeitslast kurzfristig sehr stark schwanken kann. Wenn z.B. die Firma Otto Versand kurz vor den ARD Fernsehnachrichten um 20:00 eine Anzeige schaltet, geht die Anzahl der Zugriffe auf ihre Home Page sprunghaft um einen Faktor 10 oder 100 in die Höhe. Bricht der Server zusammen, war das Geld für die Fernsehwerbung umsonst ausgegeben.

Von Jahr zu Jahr müssen die Unternehmen sich auf ein starkes Kapazitäts-Wachstum einstellen. Zur Abhilfe ist es üblich, mehrere physische Server parallel zu schalten, und bei Wachstumsbedarf um zusätzliche Server zu erweitern.



Der HTTP Server behandelt Anforderungen für (meistens) statische Ressourcen: HTML Seiten, GIF Dateien, sowie für Servlet oder CGI Aufrufe .

Hohes Verkehrsaufkommen, kurzlebige Anforderungen. Vielfach reicht die Verarbeitungskapazität eines einzelnen Servers nicht aus. Die Skalierung wird durch mehrfache parallele HTTP Server Engines erreicht.

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

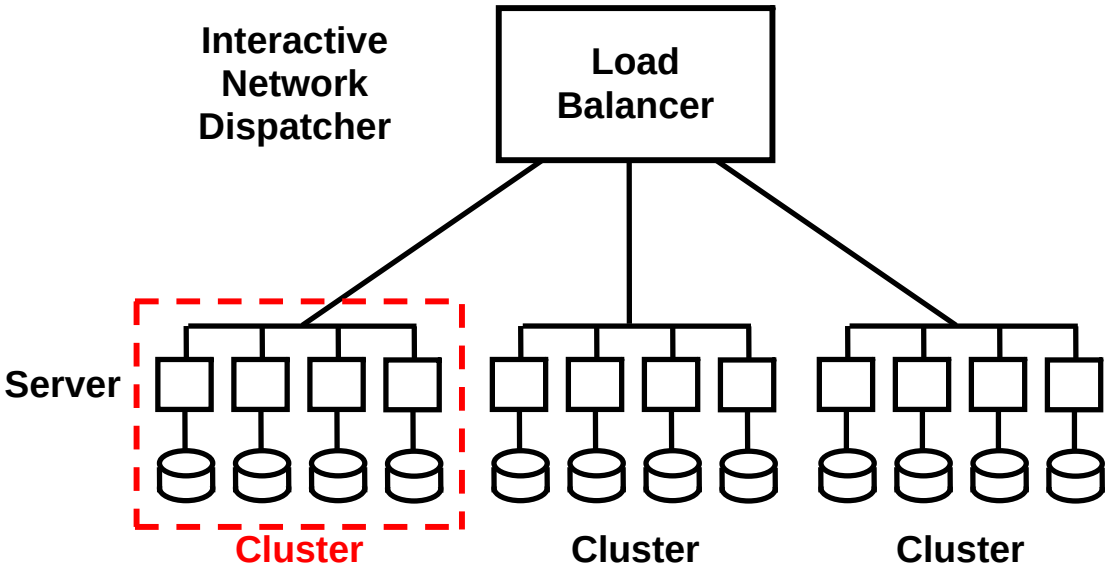
Meistens wird angenommen, dass die einzelnen Anforderungen keine großen Unterschiede bezüglich der erforderlichen Resourcen haben.

Ein extremes Beispiel: www.google.com

HTTP Server Farm

www.google.com

Google unterhält 2009 mehr als 30 Rechenzentren mit mehreren 10 000 Servern (Blades) in jedem Rechenzentrum. In einem Rechenzentrum stehen zahlreiche Cluster. Jeder Cluster dupliziert den ganzen Google Datenbestand. Das Update der Datenbestände in den einzelnen Clustern erfolgt nicht notwendigerweise synchron; es ist deshalb möglich, dass man bei zwei gleichzeitigen Anfragen unterschiedliche Antworten erhält.



Der „Load Balancer“ (andere Bezeichnung „Interactive Network Dispatcher“) ist eine zentrale Componente, welcher einkommende Anfragen auf die einzelnen Rechenzentren und von dort auf die einzelnen Cluster verteilt. Jeder Cluster ist in der Lage, jede Art von Anfrage zu bearbeiten.

Ein einfacher Workload Algorithmus, z.B. Round Robin, verteilt Anfragen der Reihe nach auf die einzelnen Cluster und Server.

Die Google Datenbank ist zwischen den Rechenzentren und zwischen den Clustern dupliziert. Die Kopien sind nicht notwendigerweise auf dem gleichen Änderungsstand. Die allermeisten Abfragen sind read-only.

Google hat 19 Data Center Standorte in den USA, 12 in Europe, 1 in Russia, 1 in South America, und drei in Ostasien

Close-up of a Google server rack.

http://royal.pingdom.com/2008/04/11/map-of-all-google-data-center-locations/ , April 2008

Google Infrastructure

Google's Server Infrastructure besteht aus mehreren Server Typen, von denen jeder eine andere Aufgabe hat:

- **Google Load Balancer nehmen die Client Request (Anfrage) entgegen, und leiten sie an einen freien Google Web Server mittels eines Squid (Open Source) Proxy Servers weiter.**
- **Squid proxy Server erhalten die Client Request und geben das Ergebnis zurück, wenn es sich in einem lokalen Cache befindet. Andernfalls wird sie an einen Google Web Server weitergegeben. Squid ist ein Proxy Server und Web Cache Daemon. Es verbessert die Performance indem wiederholte Requests in dem Cache gehalten werden.**
- **Google Web Server koordinieren die Ausführung der Client Queries und formattieren das Ergebnis in eine HTML Seite. Zur Ausführung gehen die Anfragen an Index Server. Diese berechnen den Rank, holen Vorschlägen von den Spelling Servern ein, und besorgen eine Liste von Advertisements von den Ad Servern.**
- **Data-gathering Servers durchsuchen (spider) ständig das Web. Google's Web Crawler wird als GoogleBot bezeichnet; Sie updaten die Index und Document Database Server und benutzen Google's Algorithmen um den Rank zu berechnen.**
- **Jeder Index Server enthält einen Set von Indexes. Sie erstellen eine Liste von Document IDs ("docid"), dergestallt, dass Dokumente mit einer spezifischen docid das Query Wort enthalten.**
- **Dokument Server speichern Dokumente. Jedes Dokument ist auf Dutzenden von Dokumenten Servern gespeichert. Bei einer Search gibt der Dokumentenserver ein Summary des Dokuments auf der Basis der Query Worte zurück. Dokument Server können auch das vollständige Dokument laden wenn gewünscht.**
- **Ad Servers manage Anzeigen (Advertisements).**
- **Spelling Servers erstellen Vorschläge für das Spelling von Anfragen.**

Unterschied zwischen der Google Infrastruktur und einem Web Application Server

Die Google Infrastruktur verwendet Standard x86 und PC Bauteile, Linux als Betriebssystem, sowie proprietäre Software. Kennzeichnend sind die verteilten (distributed) Data Base Characteristics mit (fast ausschließlich) read-only Queries, sowie (fast) keinen Anforderungen an Datenintegrität.

Amazon und eBay verwenden einen ähnlichen Ansatz für ihre Query Front-Ends (Beispiel: Durchsuchen aller Romane vom Autor xyz). Für das Backend, (ein Buch kaufen oder ein Gebot abgeben), ist ein anderer Ansatz erforderlich. Für die Backend Verarbeitung verwendet Amazon ein großes Unix System, eBay ein z/OS System. Backend Verarbeitung erfordert transaktionale Integrität für alle verarbeiteten Daten. Bei Google existiert (fast) keine äquivalente Backend Verarbeitung und auch kaum Bedarf an Datenintegrität.

Anders als bei Goggle müssen die betriebswirtschaftlichen Installationen großer Unternehmen oder Organisationen eine sehr große Anzahl von stark unterschiedlichen Anwendungen unterstützen, mit unterschiedlichen runtime Anforderungen und sehr unterschiedlichen Ausführungszeiten. Transaktionale Integrität sowie Synchronisation bei verteilten Datenbanken sind Schlüsseleigenschaften.

Web Application Server wie WebLogic, Netweaver und WebSphere, sowie Transaction Server wie CICS und IMS/DC, MS Transaction Server und Tuxedo erfüllen diese Anforderungen. WebLogic, Netweaver und WebSphere WAS implementieren den JEE Standard.

Wir benutzen WebSphere in unseren praktischen Übungen.

IBM WebSphere Produkt Familie

Der Begriff WebSphere ist doppelt belegt.

Offiziell bezeichnet WebSphere eine Produktlinie der Firma IBM, die unterschiedliche Software für Anwendungsintegration, Infrastruktur (z. B. Transaktionen und Warteschlangen) und eine integrierte Entwicklungsumgebung umfasst.

Zu den WebSphere-Produkten gehört unter anderem:

- **WebSphere Application Server (WAS)**
- **WebSphere Rational Application Developer und dessen WDz Erweiterungen für z/OS**
- **WebSphere Process Server**
- **WebSphere Integration Developer**
- **WebSphere MQ (andere Bezeichnung für MQSeries)**
- **WebSphere Portal Server**

In den meisten Fällen wird der WebSphere Application Server (WAS) einfach als WebSphere bezeichnet. WAS ist eine der führenden Implementierungen des JEE standards.

WebSphere Produkt Familie

Der WebSphere Application Server (WAS) stellt eine Plattform für das Deployment und die Ausführung (execution) von Java Anwendungen und Web Services zur Verfügung, besonders Servlets, JSPs und EJBs.

WebSphere Rational Application Developer und dessen RDz Erweiterungen für z/OS sind die integrierte Entwicklungsumgebung (IDE) für WAS.

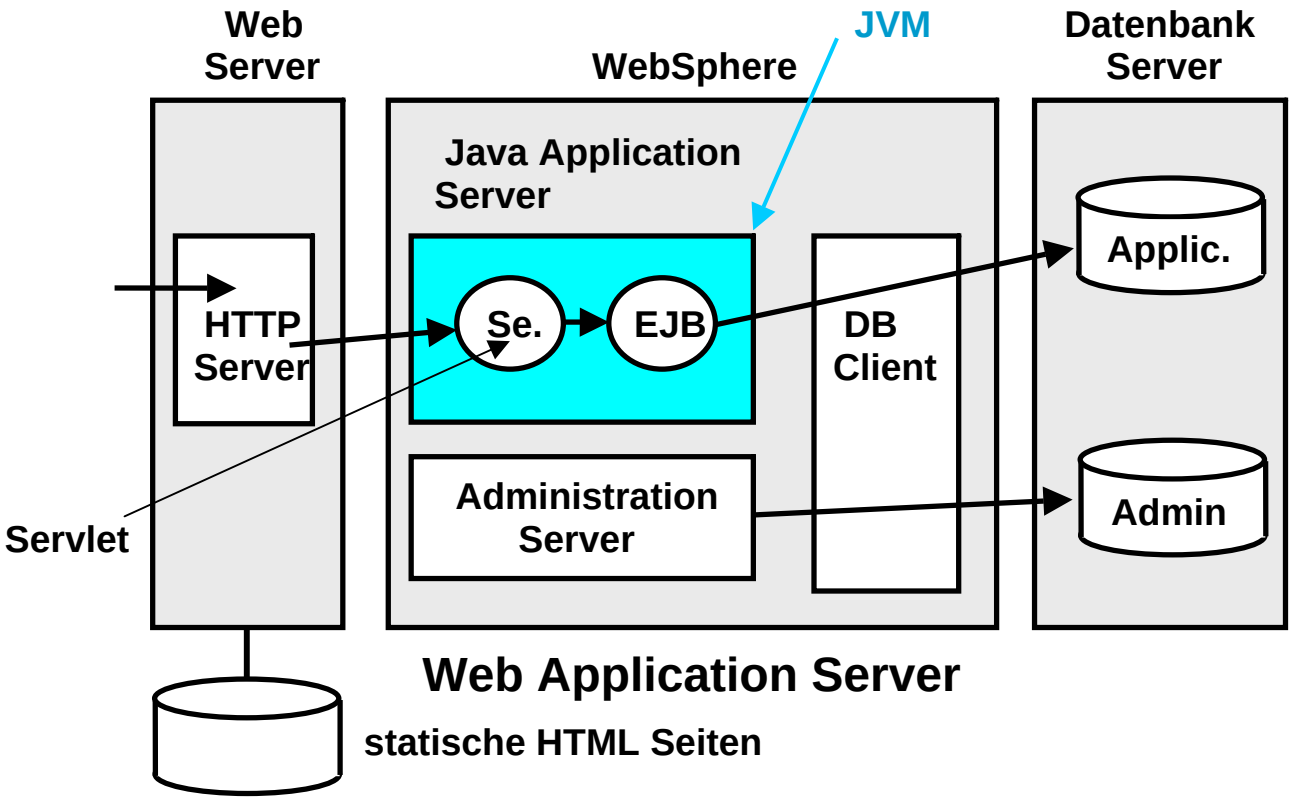
Für den WebSphere Application Server ist eine Erweiterung unter dem Namen WebSphere Process Server verfügbar. Dieser enthält Unterstützung für Java Business Processes und IBM JEE Programming Extensions. Eine Vorgänger Version hatte den Namen WebSphere Business Integration Server Foundation.

WebSphere Integration Developer ist die Entwicklungsumgebung für den WebSphere Process Server

WebSphere MQ ist die neuere Bezeichnung für MQSeries und ist lose in die WebSphere Familie integriert. WebSphere MQ unterstützt Java Message Driven Beans.

Der WebSphere Portal Server ist ein Web-basiertes User-Interface, das flexibel angepasst und personalisiert werden kann. Es ist dafür gedacht, ein einheitliches Front End mit flexibler Anbindung von verschiedensten Backend-Systemen schnell herzustellen. IBM hat maßgeblich die beiden im Portal-Umfeld relevanten Standards beeinflusst: Den Java Portlet Standard, JSR 286, sowie den WebServices for remote Portlets Standard, WSRP 2.0.

Von allen WebSphere Produkten existieren mehrere Implementierungen mit jeweils unterschiedlichem Funktionsumfang.



WebSphere WAS Grundstruktur

Dargestellt ist ein IBM WebSphere Application Server mit einem einzigen Web Container und EJB Container. Beide können in der gleichen JVM untergebracht werden, was einen direkten Aufruf von EJBs durch Servlets ermöglicht, und den RMI/IIOP Overhead vermeidet. Zusätzlich ist ein Data Base Client vorhanden, z.B. DB2connect. Der Administration Server stellt die Benutzer-Schnittstelle für die WebSphere Administration zur Verfügung.

Der http Server empfängt http Nachrichten und befriedigt diese, wenn es sich um Anforderungen für statische HTML Seiten handelt. Andernfalls werden die Nachrichten an ein Servlet weitergereicht.

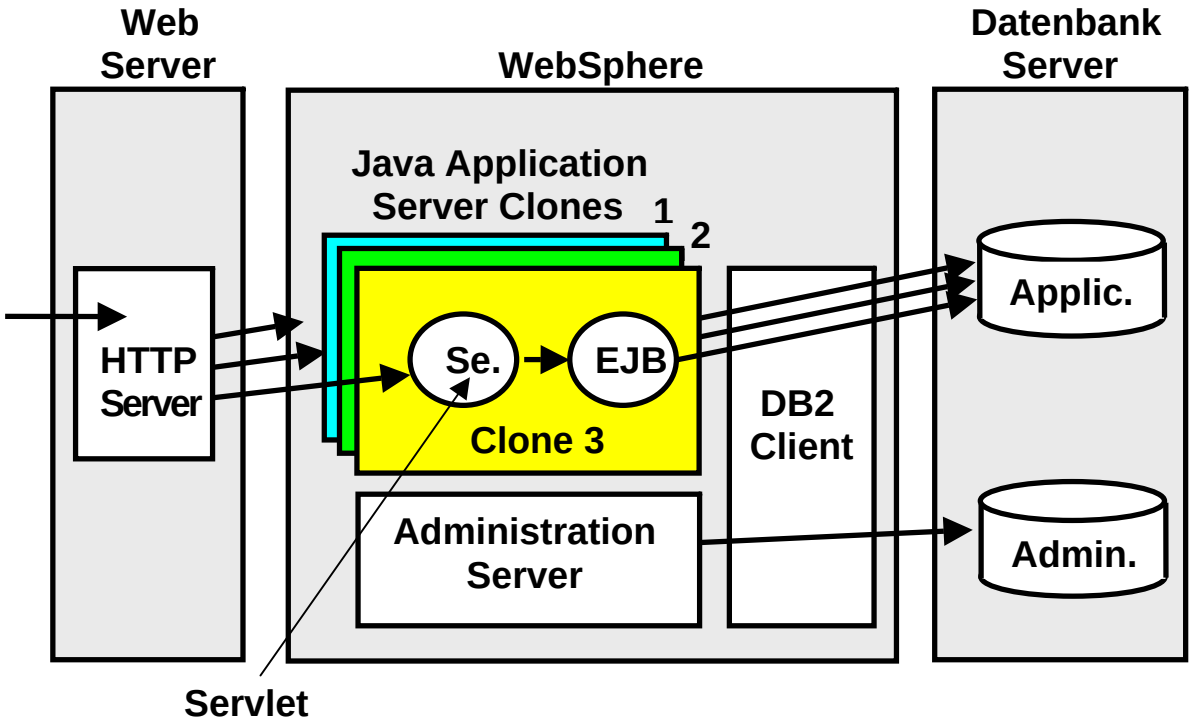
Innerhalb der JVM können mehrere Servlets und mehrere EJBs als parallele Threads laufen.

WebSphere Administration

Ein WebSphere Web Application Server (WAS) enthält neben einem Servlet Container (andere Bezeichnung Web Container) und einem EJB Container noch weitere Komponenten.

Bei einem CICS Server erfolgt die Administration durch eine eingebaute Kommando Shell, die durch die CEDA Transaktion (und weitere Transaktionen) implementiert wird, Bei WebSphere fasst man alle administrativen Funktionen in einem eigenen Server, dem WAS Administration Server zusammen. Dieser kommuniziert über eine grafische Oberfläche mit dem Benutzer.

Weiterhin gehört zur WAS Grundausstattung ein DB Client für Zugriffe auf eine DB2 Datenbank.



Aus Performance-Gründen muss der Application Server multiprogrammiert arbeiten. Im Prinzip ist das mit Hilfe von Threads innerhalb der gleichen JVM möglich. Hierbei gibt es jedoch Grenzen. Deshalb sind mehrfache Clones des Application Servers vorhanden, die Java Anwendungen parallel verarbeiten können. Die Clones des Java Application Servers verfügen jeder über eine eigenen Java Virtuellen Maschine.

Auf mehreren Clones kann die gleiche Anwendung laufen. Es ist auch möglich, dass auf den Clones unterschiedliche Anwendungen laufen, oder dass eine bestimmte Anwendung erst bei Bedarf geladen wird.

Distributed Server

Middleware Produkte wie WebSphere, aber auch CICS und DB2 sind grundsätzlich in zwei verschiedenen Versionen verfügbar:

- **Die z/OS Version läuft unter dem z/OS Betriebssystem**
- **Die „Distributed“ Version läuft auf einem getrennten Linux, Unix oder Windows Server, der über ein geeignetes Protokoll (TCP/IP, Corba, IIOP, http, andere) mit dem zentralen z/OS Server verbunden ist. Auch Betriebssysteme wie Solaris und HP-UX sind denkbar.**

Die Distributed Version ist in der Regel bezüglich Hardware- und Software Lizenzkosten günstiger, verzichtet aber auf z/OS Funktionen wie Reliability, Availability, Sicherheit, LPARs, Coupling Facility, WLM, usw. Es ist nicht unüblich, dass in einer Oranisation manche WAS Anwendungen auf distributed Servern, andere unter z/OS ausgeführt werden.

Für CICS und DB2 handelt es sich bei der distributed Version und der z/OS Version um jeweils zwei unterschiedliche Software Produkte. Wegen der ausgezeichneten Kompatibilität ist der Unterschied für den Benutzer kaum bemerkbar.

Für den WebSphere Application Server existiert nur eine einzige Quellcode Version für alle Plattformen. Durch eine Compilierung mit unterschiedlichen Compiler Optionen werden die Versionen für die unterschiedlichen Plattformen erzeugt. Dabei werden bei der distributed Version viele Funktionen disabled, die unter z/OS verfügbar sind.

Ein Beispiel ist die Benutzung des z/OS Work Load Managers an Stelle des eingebauten einfachen Work Load Managers. Sysplex und Coupling Facility Unterstützung sind weitere Beispiele. Insgesamt verhält sich die z/OS Version in Bezug auf Funktionsumfang und Leistungsverhalten deutlich anders als die distributed Version. Lediglich der eigentliche Java Quell Code ist bei beiden Versionen identisch.

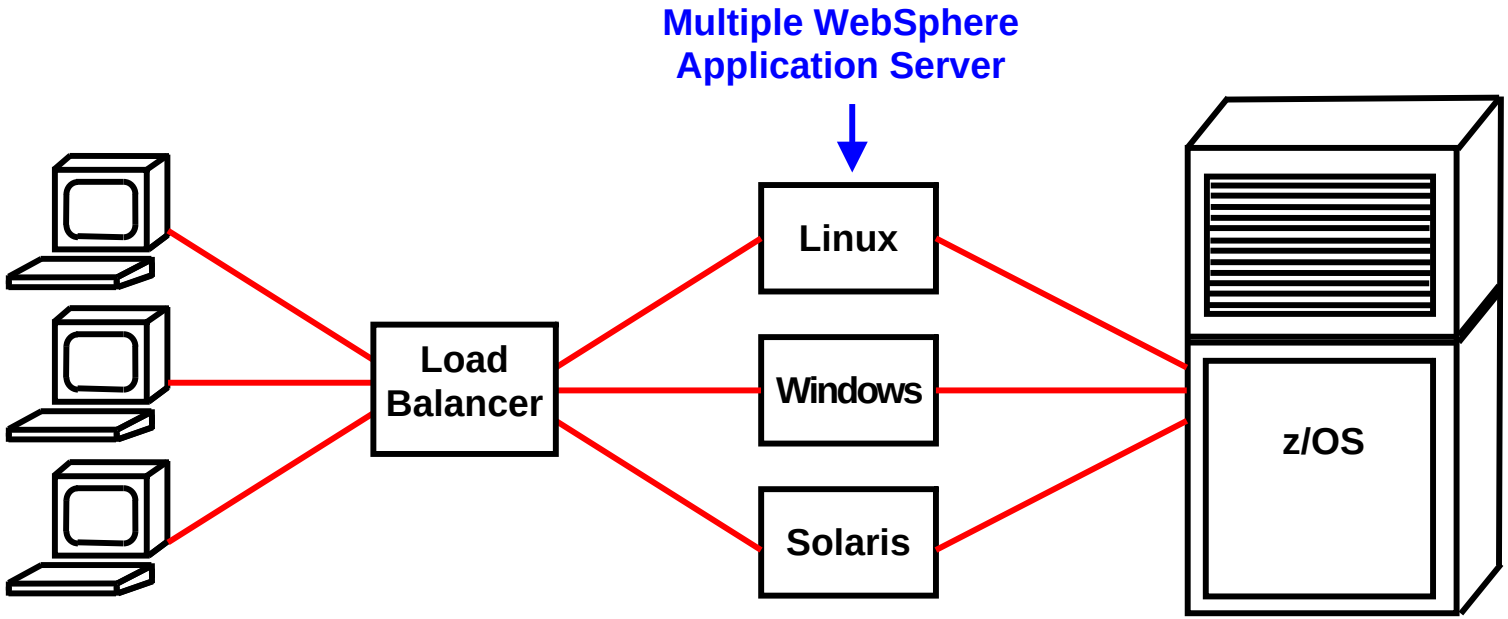
Der z/OS WebSphere Application Server (WAS) ist als Unix Anwendung konzipiert. Die z/OS Version läuft deshalb unter Unix System Services.

Structured File Server (SFS)

Ein Structured File Server ist ein File Server, der u.A. mit der distributed Version von CICS oder WAS eingesetzt wird. Er hat gegenüber einem normalen File Server diese zusätzlichen Eigenschaften:

- **Record-orientierte Dateien. SFS unterstützt Record orientierte Dateitypen. SFS organisiert die Datensätze in Feldern und bietet sowohl Record-Level als auch Feld-Level Zugriff auf Daten. Der Zugang auf Records erfolgt durch Indizes, so dass eine Anwendung schnell und einfach Zugriff auf Records hat.**
- **Transaction Protection. SFS bietet transaktionalen Zugriff auf Daten, die in einer Datei gespeichert sind. Dateien, die von SFS verwaltet werden, sind somit vollständig wiederherstellbar im Falle von Server Problemen, Netzwerkausfällen und Medien Ausfällen. SFS protokolliert automatisch alle Änderungen von Daten, die in SFS-Dateien gespeichert sind. SFS stellt sicher, dass alle Änderungen, die während der Verarbeitung einer Transaktion anhängig sind, entweder abgeschlossen oder vollständig rückgängig gemacht werden.**

SFS ist eine Art Ersatz für VSAM bei der z/OS Version von CICS.



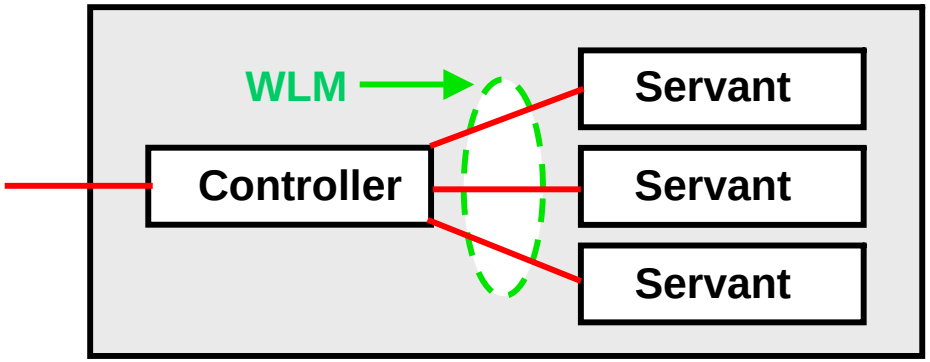
Beim Einsatz der distributed WebSphere Application Server Version läuft auf einem physischen Server häufig nur eine einzige Anwendung, möglicherweise parallel auf mehreren Clones. Für mehrere Anwendungen sind dann mehrere physische Server vorhanden, z.B. 30 Server für 30 unterschiedliche Anwendungen.

Der http Server braucht eine Einrichtung, um eintreffende Nachrichten auf die richtigen WebSphere Server und Clones zu routen. Dies ist die Funktion des Interactive Network Dispatchers (Load Balancer). In einfachen Fällen wird ein simpler Round Robin Algorithmus benutzt.

Load-Balancing Internet Servers. IBM Redbook, SG24-4993-00, Dezember 1997,
http://www.redbooks.ibm.com/redbooks/pdfs/sg244993.pdf

WebSphere Application Server for z/OS

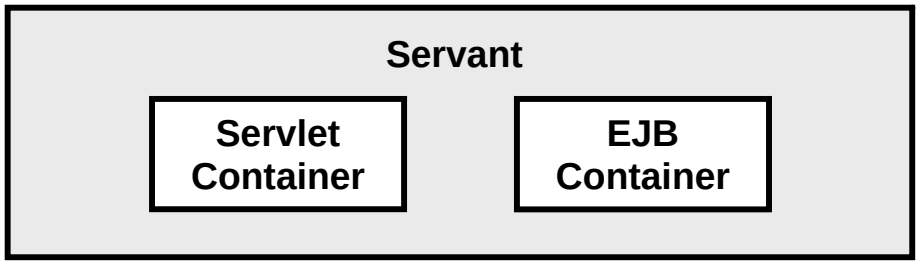
Die z/OS Version von WebSphere benutzt an dieser Stelle die „Controller-Servant“ Konfiguration. Der Controller übernimmt die Aufgabe des Interactive Network Dispatchers für eine Gruppe von Servants. Letztere stellen die eigentlichen Application Server dar. Controller und Servant besitzen jeder eine eigene JVM.



WebSphere Application Server (WAS)

Controller und Servant laufen in unterschiedlichen virtuellen Adressenräumen; der Controller läuft mit Speicherschutzschlüssel 2 und im Kernel Status. Die Servants laufen mit Speicherschutzschlüssel 8 im User Status. Vielfach läuft eine einzige (oder nur wenige) Anwendung(en) in einem Servant.

Der Controller benutzt den z/OS Work Load Manager (WLM) um eingehende Anfragen an die Servants zu verteilen.



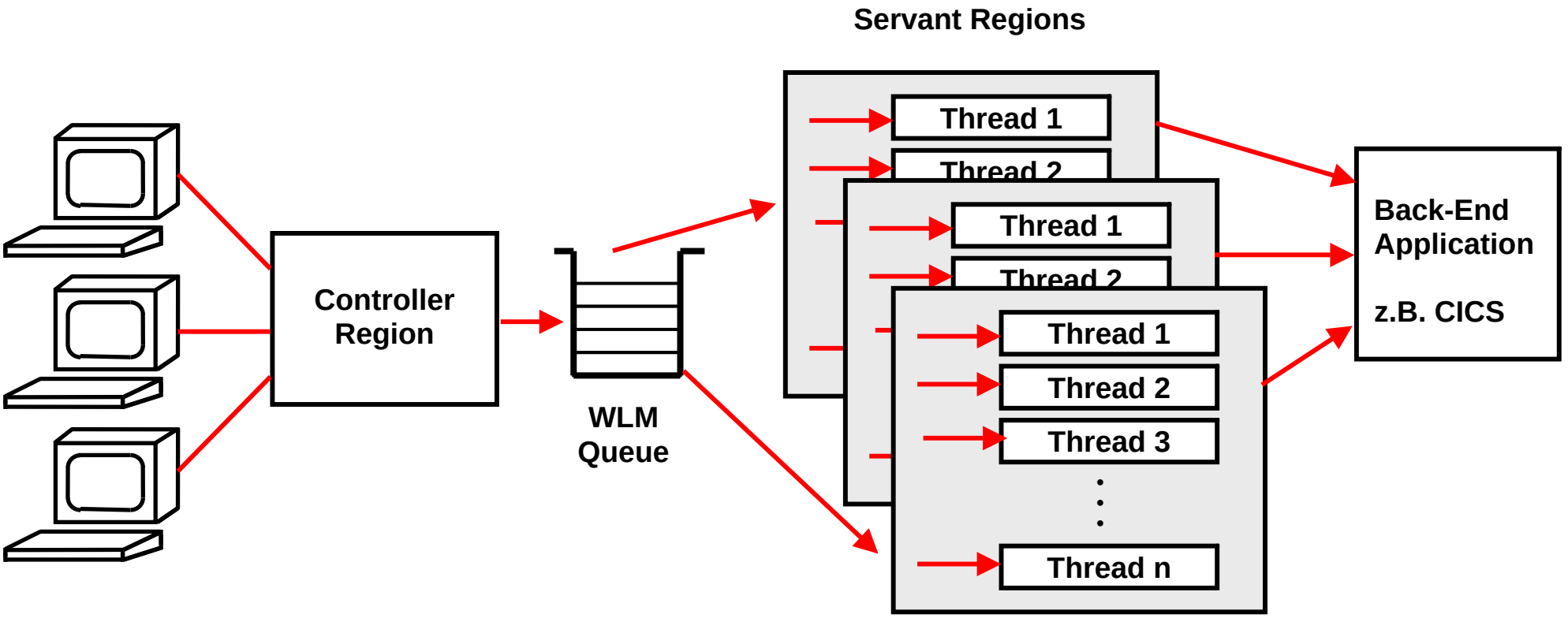
WebSphere Runtime Structure on z/OS

Dies sind die Merkmale

- **Die grundlegende Ausführungsumgebung für den WebSphere Application Server (WAS) unter z/OS ist ein Server.**
- **Ein Server ist ein Controller / Servant Konfiguration**
- **Controller und Servant laufen in getrennten Regions (virtuelle Adressenräume)**
- **Für jede Controller-Region existiert eine oder mehrere Servant Regionen**
- **Der Controller ist der Kommunikationsprotokoll Einstiegspunkt (empfängt Nachrichten)**
- **Unterschiedliche Nachricht Kommunikationsprotokolle werden von dem Controller unterstützt, besonders HTTP(S) und IIOP**
- **Die JEE-Komponenten (besonders EJBs) werden in der Servant Region ausgeführt, mit einer WLM Queue zwischen Controller und Servant**

Darüber hinaus:

- **Es können mehrere WAS auf einem einzigen z/OS-System existieren, jeder mit seinem eigenen Controller und mehreren Servants**
- **Auf einem z/OS System, spezifisch einem Sysplex, laufen häufig mehrere WebSphere Web Application Server.**



Zur Verbesserung des Leistungsverhaltens laufen mehrere Servant Prozesse auf dem Web Applikations-Server. In jeder Servant Region existiert (normalerweise) eine einzige JVM. In der Servant JVM können multiple Java Threads individuelle Anwendungen parallel verarbeiten.

Mögliche Backend Applications sind z.B. CICS, SAP/R3, DB2 oder IMS.

In der WebSphere Application Server für z/OS erfolgt die Zuordnung (classification) jeder Transaktion durch eine Controller Region. Der Controller ist ein getrennter Process, der in einem eigenen Address Space läuft. Die Controller Region agiert als ein Queuing Manager. Sie verwendet den z/OS Work Load Manager, und schedules eintreffende Work Requests zur Ausführung in multiple Server Address Spaces, die als Servants bezeichnet werden.

Aufgaben der Queues

Aufgaben der Control Region und ihrer Queues sind:

- **Lastverteilung**
- **Schutz (Isolation) der Anwendungen gegeneinander**
- **Availability und Reliability 24 Stunden/Tag, 7 Tage/Woche. Verabschiedet sich ein Prozess, läuft der Rest weiter**
- **Austesten neuer Anwendungen**

Unter z/OS optimiert der Work Load Manager die Lastverteilung. Die Distributed WebSphere Application Server Version hat statt dessen eine eigene primitiven Work Load Manager Funktion. Anforderungen von dem Controller gehen (je nach Policy) zu einer von mehreren Queues. Jede Queue wird von mehreren Java Prozessen bedient.

Der Administrator legt die Anzahl und die Policies jeder Queue fest. Die Queue Policy bestimmt

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

Ein z/OS WebSpere Application Server (WAS) Servant beherbergt Servlets, EJBs und andere Java Klassen. Die vom Work Load Manager verwalteten Queues verteilen eintreffende Nachrichten auf die einzelnen Servants. Jeder Servant läuft in seiner eigen z/OS Region.

z/OS and Work Load Manager

Um die Arbeit des Systems auszubalanzieren, verwendet WebSphere die Dienste des z/OS Workload Managers (WLM). Drei verschiedene WLM Dienstleistungen werden von WebSphere eingesetzt:

1. **Routing**
   Die WLM Routing Service wird benutzt, um Klienten mit einem bestimmten Servant zu verbinden, unter Berücksichtigung der derzeitigen Systemauslastung und des WLM Performance Indexes

2. **Queuing und Adressraum Management**
   WLM manages Servant Adressenräume um Performance Goals und deren WLM Performance Index (PI) für die derzeitige Work Load zu optimieren.

3. **Priorisierung um Performance-Ziele zu erreichen**
   WebSphere delegiert die Verantwortung für das Starten und Stoppen von Servant Regionen an den WLM Address Space Management Service.