UNIVERSITÄT LEIPZIG

Mainframe Internet Integration

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

SS2013

Work Load Management Teil 2

System Resource Manager

Resource Management

Prozesse, die in einem z/OS System laufen, benötigen Betriebsmittel (Resourcen)

- **CPU Zeit**
- **Hauptspeicherbedarf (Rahmen im Hauptspeicher)**
- **I/O Operationen**

Es existieren große Unterschiede, in welchem Umfang die einzelnen Prozesse Resourcen benötigen. Manche Prozesse benötigen viel CPU Zeit, aber wenig Hauptspeicherplatz. Bei anderen Prozessen ist es umgekehrt.

Der Resourcen Verbrauch eines Prozesses wird in Service Units (Resourcenverbrauch einer Work Unit) gemessen. Es existieren unterschiedliche Service Unit Definitionen für die CPU Zeit, den Hauptspeicherbedarf und die I/O Operationen.

Wir schauen uns das Resourcen Management an Hand der Hauptspeicherverwaltung näher an.

Flattern (Thrashing)

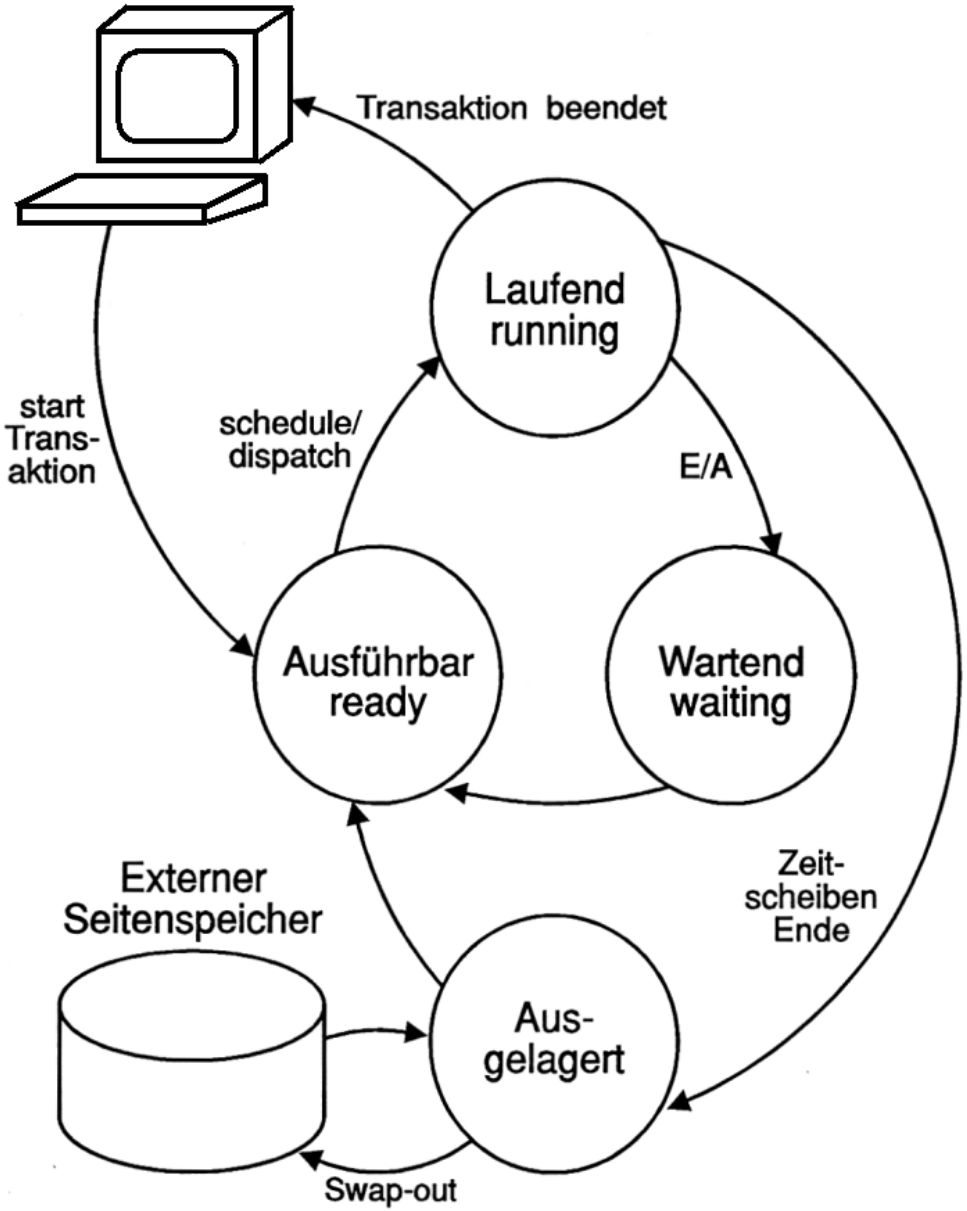
Auf einem Rechner laufen gleichzeitig zahlreiche Prozesse. Sie wechseln zwischen den Zuständen laufend, wartend und ausführbar. Wir bezeichnen diese Prozesse als aktive Prozesse (siehe Einführung in z/OS, Verarbeitungs-grundlagen, Teil 1, http://jedi.informatik.uni-leipzig.de/de/Vorles/Einfuehrung/Grundlag/vg01.pdf#page=12). Normalerweise steigt die Auslastung der CPUs, je größer die Anzahl der aktiven Prozesse ist.

Jeder aktive Prozess beansprucht realen Hauptspeicher für seine aktiven Seiten (im Hauptspeicher abgebildet). Nur ein Teil der Seiten des virtuellen Speichers ist in jedem Augenblick in Rahmen des realen Hauptspeichers abgebildet. Der größere Teil der Seiten eines virtuellen Speichers ist in jedem Augenblick auf einem externen Seitenspeicher ausgelagert. Der reale Speicher besteht somit aus 2 Teilen: dem realen Hauptspeicher und dem externen Seitenspeicher (pagefile.sys unter Windows). Beim Zugriff zu einer ausgelagerten Seite (nicht in einem Rahmen des Hauptspeichers abgebildet) erfolgt eine Fehlseitenunterbrechung (Page Fault). Diese bewirkt den Aufruf einer Komponente des Überwachers (Seitenüberwacher, Paging Supervisor), der die benötigte Seite aus dem externen Seitenspeicher holt und in den Hauptspeicher einliest. In den meisten Fällen muss dafür Platz geschaffen werden, indem eine andere Seite dafür auf den externen Seitenspeicher ausgelagert wird. Dieser Vorgang wird als Demand Paging bezeichnet.

Wenn Ihr Windows-, Linux oder Apple Rechner genügend viel Hauptspeicher hat, können evtl. 100 % der Seiten eines virtuellen Speichers in Rahmen des realen Hauptspeichers abgebildet werden. Bei einem Mainframe mit bis zu mehreren TByte Hauptspeicher ist dies fast nie der Fall.

Steigt die Anzahl der aktiven Prozesse, so wird der Prozentsatz der Seiten immer kleiner, die für jeden Prozess in Rahmen des realen Hauptspeichers (an Stelle des externen Seitenspeichers) abgebildet werden. Je kleiner der Prozentsatz, um so größer ist die Wahrscheinlichkeit einer Fehlseitenunterbrechung. Die Seitenfehlerrate steigt umso mehr an, je geringer die Anzahl der im Hauptspeicher verfügbaren Rahmen ist.

Wird die Anzahl zu gering, tritt ein als „Flattern“ (Thrashing) bezeichnetes Problem auf. Die Prozesse stehlen sich gegenseitig benötigte Rahmen und können auf Grund zahlreicher Fehlseitenunterbrechungen kam noch produktive Arbeit leisten.



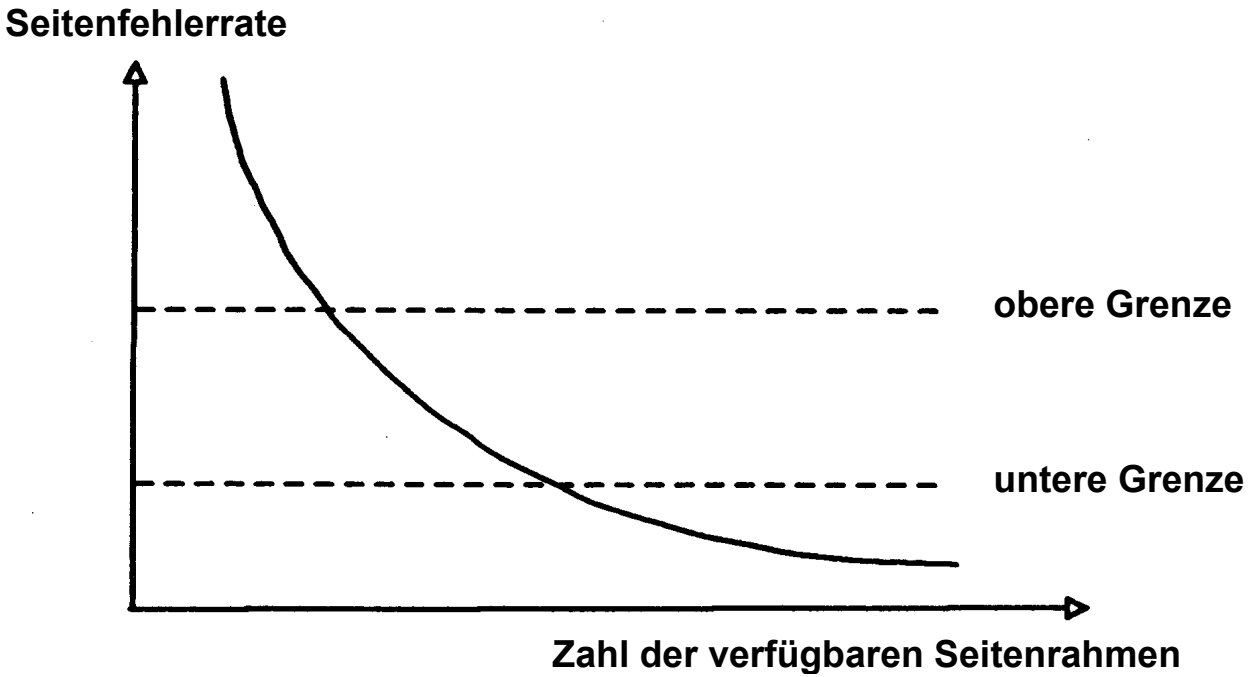
Aktive Prozesse beanspruchen Rahmen im realen Hauptspeicher.

Inaktive Prozesse sind auf den externen Seitenspeicher (auxiliary Storage) ausgelagert und verfügen über keine Rahmen im realen Hauptspeicher.

Verringert man die Anzahl der aktiven Prozesse indem man die Anzahl der (ausgelagerten) inaktiven Prozesse erhöht, stehen den restlichen aktiven Prozessen mehr Rahmen im realen Hauptspeicher zur Verfügung. Die Seitenfehlerrate sinkt.

Wenn die Anzahl der aktiven Prozesse zu klein ist, werden möglicherweise andere Resourcen, z.B. CPU Zeit schlecht ausgenutzt.

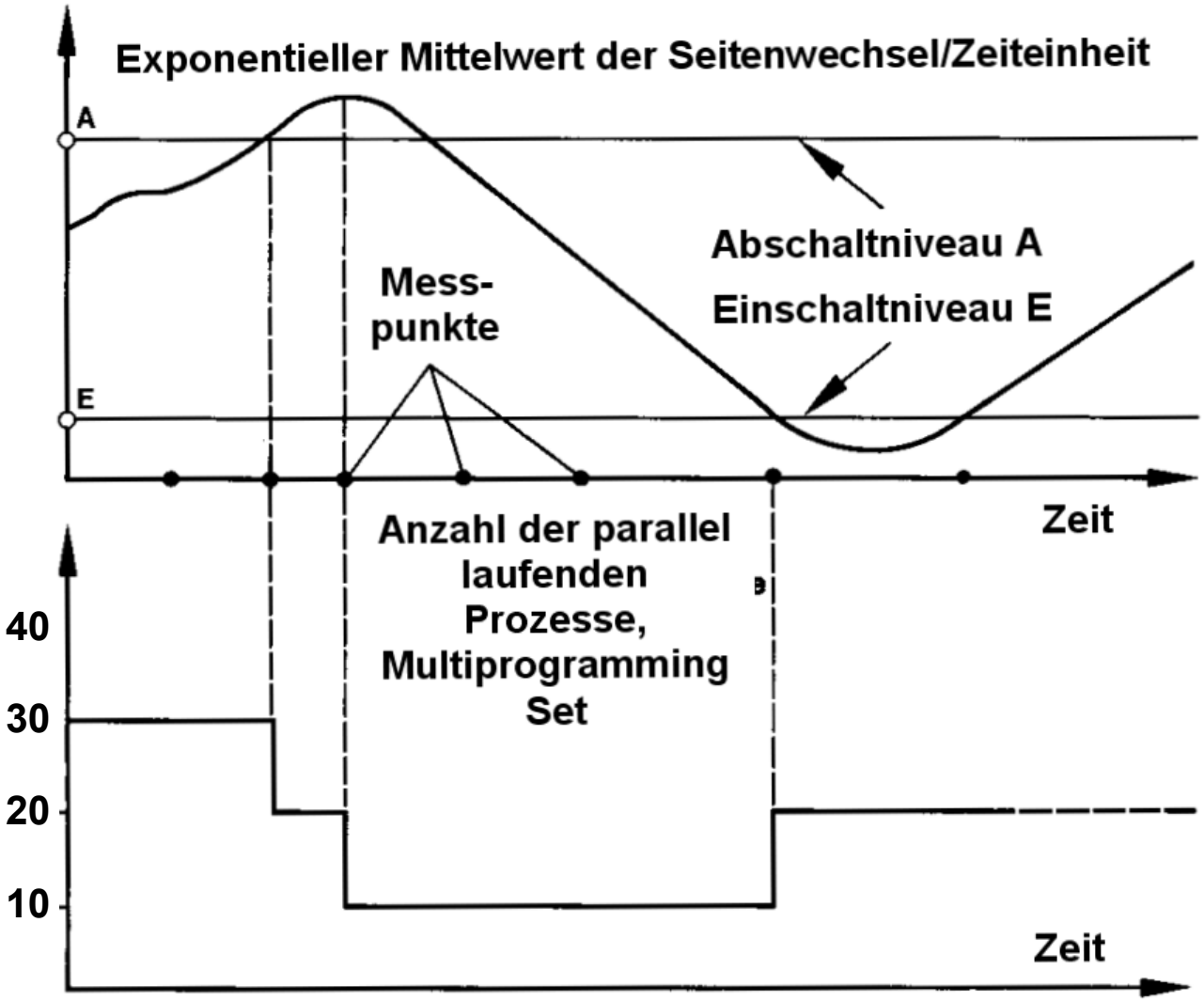
Auslagern von Prozessen



Begrenzung der Seitenfehlerrate

Um Flattern zu verhindern, beobachtet der Workload Manager ständig die Anzahl der Fehlseitenunterbrechungen pro Zeitintervall (z.B. alle 10 Sekunden). Übersteigt die Anzahl der Fehlseitenunterbrechungen eine festgelegte obere Grenze, reduziert WLM die als „Multiprogramming Level“ bezeichnete Anzahl der aktiven Prozesse. Einige Prozesse werden inaktiviert, indem sie mit einem als Swap-Out bezeichneten Verfahren ganz auf den externen Seitenspeicher ausgelagert werden; sie verlieren alle Rahmen im realen Hauptspeicher. Den übrigen aktiven Prozessen stehen damit mehr Hauptspeicherrahmen zur Verfügung, und die Seitenfehlerrate sinkt. Es werden solange Prozesse mit niedriger Priorität inaktiviert, bis die Seitenfehlerrate auf einen akzeptablen Wert sinkt.

Unterschreitet die Seitenfehlerrate eine festgelegte untere Grenze, wird WLM die Anzahl der aktiven Prozesse wieder erhöhen (Swap-in), um die Auslastung der CPUs und/oder anderer Resourcen zu verbessern.



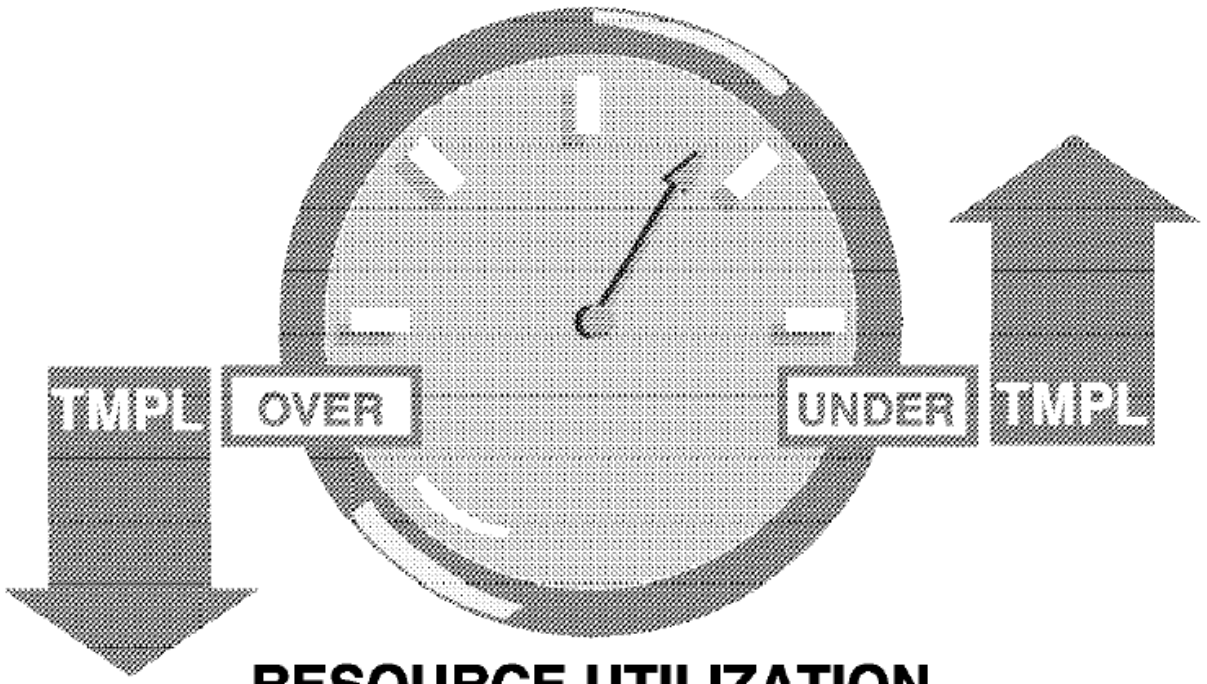
Als Multiprogramming Set wird die Menge der derzeitig aktiven Prozesse bezeichnet.

Dargestellt ist, wie über festgelegte Zeitintervalle die durchschnittlich Seitenfehlerrate ermittelt wird.

Übersteigt die durchschnittliche Seitenfehlerrate den Grenzwert A, so wird die Multiprogramming Level (Anzahl der Prozesse im Multiprogramming Set) reduziert.

Unterschreitet die durchschnittliche Seitenfehlerrate den Grenzwert E, so wird die Multiprogramming Level (Anzahl der Prozesse im Multiprogramming Set) wieder erhöht.

Wirkungsweise des Seitenwechsel-Begrenzers



Als Target Multiprogramming Level (TMPL) bezeichnet man die Anzahl der swapped-in Address Spaces (aktive Prozesse), die nach der Meinung von WLM das System mit seinen verfügbaren Resourcen optimal bedienen kann.

Dieser Wert ändert sich ständig..

Die Current Multiprogramming Level (CMPL) ist im Gegensatz zu TMPL die tatsächliche Anzahl der swapped in Address Spaces. Diese befinden sich untereinander im Wettbewerb um die verfügbaren System Resourcen. CPML ist identisch mit der Anzahl der aktiven Prozesse.

WLM bemüht sich, den Unterschied zwischen CMPL und TMPL möglichst klein zu halten.

Prioritätssteuerung für inaktive Prozesse

WLM entscheidet unter Prioritätsgesichtspunkten, , welcher aktive Prozess für ein Swap-out selektiert wird bzw. welcher inaktive Prozess für ein Swap-in selektiert wird. Die zugeordneten Prioritäten können sich ständig ändern. Ist z.B. zu einem Zeitpunkt die CPU Auslastung hoch, I/O Kapazität ist aber verfügbar, dann wird WLM die Priorität von CPU-lastigen Prozessen heruntersetzen, und die Priorität von I/O lastigen Prozessen heraufsetzen.

Automatische Steuerung der System-Resourcen

Die unterschiedlichen Arten von Prozessen haben unterschiedliche Zielsetzungen. So sind zB. für viele interaktive CICS oder Webserver Zugriffe optimal Antwortzeiten (gemessen in Sekundenbruchteilen) auf Kosten einer optimalen Resourcen Ausnutzung erwünscht. Bei Stapelverarbeitungsprozessen ist es umgekehrt. Es kann aber auch sein, dass für bestimmte Stapelverarbeitungsprozesse eine optimale Durchlaufzeit erwünscht ist.

Diese Zielsetzungen müssen dem System Resource Manager (SRM) vom Systemadministrator in irgend einer Form mitgeteilt werden. Die Zusammenhänge zwischen den unterschiedlichen Zielsetzungen sind jedoch sehr komplex, und ändern sich im Laufe einer Stunde, eines Tages oder Monats ständig. Mit der zunehmenden Komplexität wird es für den Systemadministrator immer schwieriger zu entscheiden, welche von zahlreichen Einstellparametern er für einen optimalen Betrieb ändern muss. Oft kann er auch gar nicht mehr schnell genug reagieren.

Immer häufiger tritt auch das Problem auf, dass die Änderung von Systemparametern unerwartete und unerwünschte Seiteneffekte erzeugt. Bei der Vielzahl der Einstellungsmöglichkeiten ist es denkbar, dass eine bestimmte Änderung keine erwartete Verbesserung, sondern eine Verschlechterung bringt.

Als Lösungsansatz wird dem Rechner die automatische Verwaltung und Steuerung aller System Resourcen übertragen. Hierzu muss ihm aber gesagt werden, unter welchen Gesichtspunkten die Optimierung erfolgen soll.

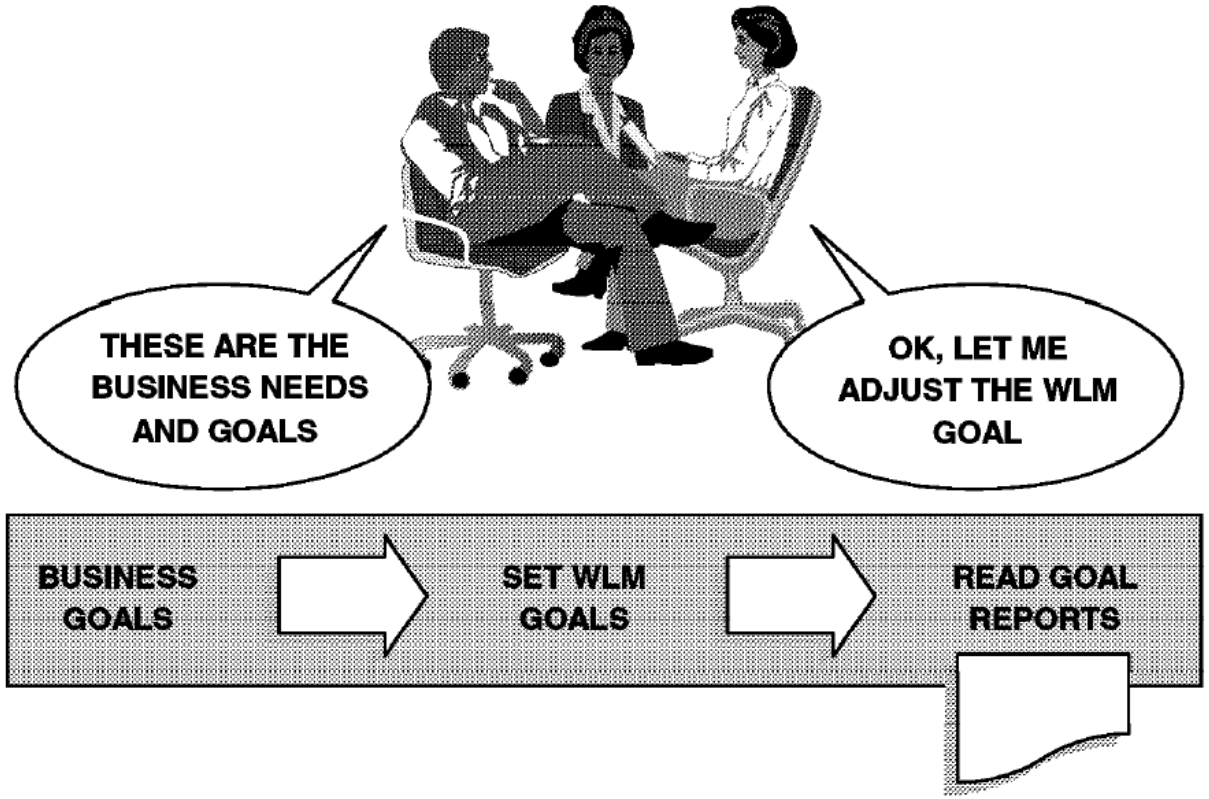
Dies ist die Aufgabe des Goal Oriented Work Load Managers.

Vor der Einführung des Goal orientierte Workload-Managers existierte bereits der System Resource Manager (SRM). SRM ist in der Lage, die Verteilung der Betriebsmittel abhängig von der anfallenden Systemlast zu steuern. Die Verteilung der Betriebsmittel erfolgte abhängig von der anfallenden Systemlast.

Hierzu sammelt die System Management Facility (SMF) System-relevante Information über die Auslastung der einzelnen Systemkomponenten. SMF misst die gesamte Systemleistung und beobachtet die Nutzung der System Resourcen (z.B. CPU, Hauptspeicher, Ein/Ausgabe).

Engpässe an Resourcen können von SRM zum Beispiel durch eine Verringerung des Multiprogramming Level und durch das Swapping von Adressenräumen aus dem Hautspeicher in den externen Seitenspeicher aufgelöst werden.

Die Steuerung erfolgte durch die mehr oder weniger statische Eingabe von SRM Konfigurationsparametern durch den Systemadministrator. Diese diese ließen sich in der Vergangenheit nur schlecht den sich dynamisch ändernden Gegebenheiten anpassen. Weiterhin ist es schwierig, bestimmte angestrebte Ziele (Business Goals, z.B. alle Transaktionen vom Typ x sollen eine Anwortzeit < 0,3 s haben) in SRM Konfigurationsparameter (z.B. Target Multiprogramming Level ≤ y) zu übersetzen.



Der Goal oriented Work Load Manager

- **reduziert den Aufwand für den System Administration**
- **reduziert die Notwendigkeit für Detailwissen**

Goal Orientierung

Der Goal oriented Workload Manager (WLM) ist ein integraler Bestandteil des z/OS-Betriebssystems. Er erweitert den Funktionsumfang des SRM, und dehnt ihn zusätzlich auf den ganzen Sysplex aus.

Es ist die Aufgabe des Goal oriented Workload Managers, die von menschlichen Benutzern gewünsche Optimierungs-Anweisungen in der Form von Geschäftsbegriffen (business terms, wie z.B. erwünschte Antwortzeit) entgegen zu nehmen. Der Goal oriented Workload Manager übersetzt diese Anweisungen in System Resource Manager (SRM) Parameter und passt sie dynamisch an die sich laufend ändernde Systembelastung an.

Das Leistungsverhalten der Installation wird durch Vorgaben für Zielsetzungen (Business Goals) festlegt. Der Goal oriented Work Load Manager setzt die Vorgaben in Einstellungen für den SRM um.

Instrumentierung

z/OS Subsysteme brauchen zusätzliche Funktionen, um mit WLM kooperieren zu können.

Der Goal oriented Workload-Manager unterstützt beispielsweise folgende Subsysteme :

- **IMS**
- **JES2/JES3**
- **TSO/E**
- **CICS**
- **OMVS**
- **DB2**
- **TCP**
- **SNA**
- **HTTP Server**
- **WebSphere**
- **MQSeries**
- **non-z/OS LPARs, z.B zLinux**

Zur Integration der Subsysteme enthalten manche Subsysteme ihre eigenen Work Load Managent Komponenten. Beispiele sind CICS und WebSphere. CICS Transaktionen können nach Ihrem Namen (TRID) klassifiziert werden. Die z/OS WebSphere Version unterscheidet sich von den WebSphere Versionen auf anderen Betriebssystem Plattformen (distributed WebSphere) unter Anderem dadurch, dass alle z/OS WLM Funktionen in Anspruch genommen werden können.

Service Klassen

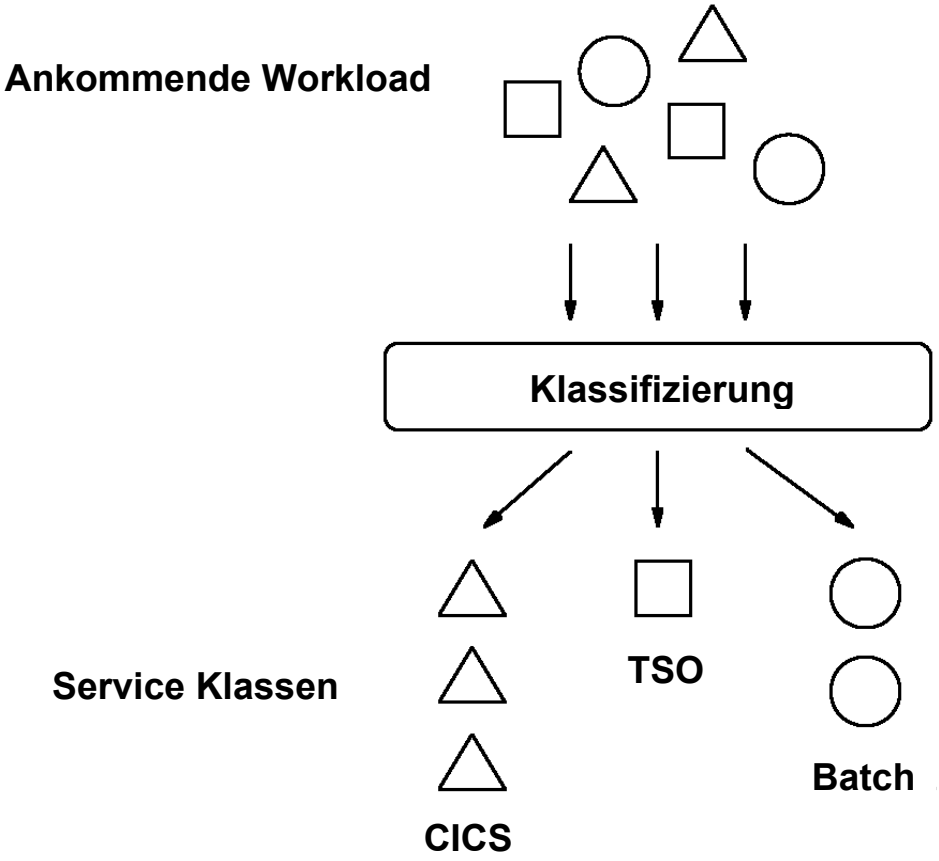
Die von einem Rechner zu verarbeitende Workload besteht aus vielen einzelnen Work Units (Arbeitseinheiten). Eine Work Unit oder Arbeitseinheit ist eine Verarbeitungsaufgabe für einen Rechner. Beispiele für Work Units sind

- **CICS Transaktionen,**
- **TSO Sitzungen,**
- **Batch Jobs.**

In einem z/OS Rechner werden in jedem Augeblick zahlreiche Work Units parallel verarbeitet. Eine Work Unit wird normalerweise durch einen Prozess ausgeführt. CICS Transaktionen sind ebenfalls Work Units. Bei der Stapelverarbeitung werden Work Units als Jobs bezeichnet.

Der Benutzer definiert mittels Zielvorgaben, welche Leistungen unter welchen Umständen von dem Rechnersystem zu erbringen sind. Das System überwacht während der Abarbeitung die Workloads, vergleicht ihre Laufzeitergebnisse mit den Zielvorgaben (Business Goals) und passt den Zugang zu und Verbrauch an Resourcen dynamisch auf der Basis der Vorgaben (Goals) an.

Theoretisch wäre es denkbar, für jede einzelne Work Unit eigene Zielvorgaben zu erstellen. Da dies unpraktisch ist, fasst man Work Units mit ähnlichen Zielvorgaben zu „Service Klassen“ zusammen. Alle in einer Service Klasse zusammengefassten Work Units erhalten die gleichen Zielvorgaben.



Service-Klassen (Dienst Klassen) sind Einheiten von Workloads mit ähnlichen Charakteristiken, für die die Zielvorgaben definiert werden. Eine mögliche - sehr einfache - Aufteilung der Work Units in Service-Klassen könnte z.B. drei Klassen vorsehen:

- **CICS Jobs,**
- **TSO Jobs,**
- **Batch Jobs**

Es ist aber auch möglich, z.B. CICS Jobs entsprechend ihrer User ID oder ihrer TRID in mehrere unterschiedliche Service-Klassen aufzuteilen

Service Klassen

Die Service-Klasse stellt das Grundkonstrukt für den Goal oriented Workload Manager dar. dar. Sie ist das Ergebnis der Klassifizierung der Workloads mit unterschiedlichen Leistungsmerkmalen in Gruppen, die mit den gleichen Ziel-Vorgaben versehen werden können. Die Anzahl der Service Klassen sollte nicht zu niedrig (schlechte Optimierung) und nicht zu hoch (hoher Administrationsaufwand und WLM Verarbeitungsaufwand) sein. Eine Anzahl von 25 - 30 Service Klassen ist in vielen Installationen ein guter Wert.

Einordnung in Service-Klassen

Classification Rules werden benutzt, um die Menge aller möglichen Arbeitsanforderungen in Serviceklassen (Dienstklassen) einzuordnen (klassifizieren). Die Klassifikation basiert auf den Attributen einer individuellen Arbeitsanforderung. Dies kann z.B. sein:

- **User ID**
- **Art des aufgerufenen Prozesses (Transaktionstyp, Stapel,)**
- **Standort oder Art des Terminals oder Klientenrechners (z.B. Standort Personalabteilung, Art Authorisierung = xxx)**
- **Accounting Information**
- **.....**

Jeder Dienstklasse sind „Ziele“ zugeordnet

- **Antwortzeit (Response Time)**
- **Geschwindigkeit (Velocity)**
- **Stellenwert (Importance)**
- **andere (discretionary)**

Jede Dienstklasse besteht aus zeitlichen Perioden (z.B. 10 Sekunden):

- **Während einer Periode sind begrenzte Resourcen verfügbar (z.B. CPU Zyklen, I/O Zugriffe, ...)**
- **Nach Ablauf der Periode `i` erfolgt eine Migration nach Periode `i + 1`**
- **Für jede Periode können unterschiedliche Ziele existieren.**

Eine realistische Annahme ist, dass nicht alle Ziele erreicht werden können. WLM platziert Arbeitsanforderungen so, dass die Wahrscheinlichkeit, alle Ziele zu erreichen, optimiert wird.

Service Definition

Die Einordnung aller Service Units in Service Klassen wird dem System in der Form einer Workload-Manager „Service Definition“ mitgeteilt. Hierzu benutzt der System Administrator eine speziellen administrative Anwendung, die „WLM Administrative Application“. Diese kann unter der Interactive System Productivity Facility (ISPF) von autorisierten TSO-Benutzern gestartet werden.

Der wichtigste Punkt bei der Definition einer Service-Klasse ist es festzulegen, wie wichtig (important) sie ist, um die übergeordneten Geschäftsziele zu erreichen. Entscheidend ist, dass manche Service-Klassen weniger wichtig eingestuft werden als andere. Bei einem optimierten System kann es durchaus sein, dass nicht alle Ziele erfüllt werden können.

Die Wichtigkeit wird dargestellt, indem eine Business Importance für die Service-Klasse definiert wird. Diese Business Importance ist später für das System der wesentliche Entscheidungsfaktor, wenn der Zugang zu den Resourcen geregelt/optimiert werden muss.