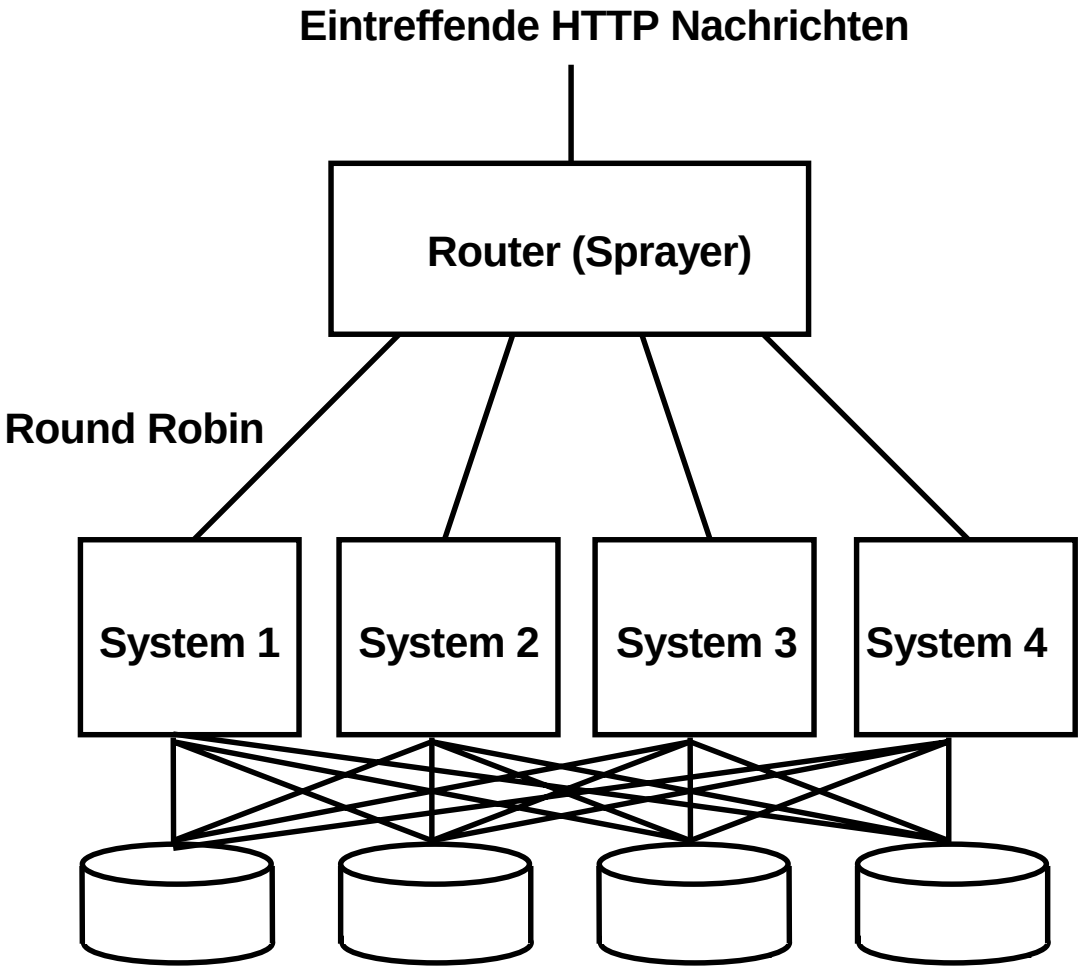
Einführung in z/OS

Prof. Dr. Martin Bogdan
Dr. rer. nat. Paul Herrmannn
Prof. Dr.-Ing. Wilhelm G. Spruth

WS 2008/2009

Teil 4

Work Load Management



Work Load Management für einen WWW Cluster

Bei großen Providern kann die Web Site aus hunderten oder tausenden von Systemen bestehen

Vorteil: Einheitliche Anwendung, nur Lesezugriffe zu den HTML Daten

Warum Work Load Manager (WLM)

Traditionelle Verfahren - viele Einstellungen:

- **Run-to-Completion**
- **Zeitscheibensteuerung, z.B. exponentiell**
- **Anzahl aktiver Prozesse - Größe des Working Sets**
- **Multithreading, Anzahl von Subsystem Instanzen**
- **Zuordnung von Anwendungen auf physikalische Server**
- **Zuordnung der E/A Kanäle**
- **Prioritäten**
- **etc.**

Komplexität wächst mit der Systemgröße

- **Stapel - interaktive Verarbeitung**
- **Belastungsschwankungen (Während des Tages, Woche, Jahr)**
- **Affinität Prozesse - Daten**
- **Zuordnung Prozesse - reale CPU´s**
- **Unterschiedliche E/A Anforderungen/Belastungen**

Zwei Alternativen

1. Unterschiedlichen Anwendungen unabhängige physikalische Server zuordnen; schlechte Auslastung akzeptieren (Hardware ist billig)

- **Viele Server**
- **Heterogene Landschaft**
- **Komplexe LAN strukturen**
- **Hoher Administrationsaufwand**

2. Workload Manager

WLM
OS/390 Work Load Manager (2)

Einstellungen justieren:

- **automatisch**
- **dynamisch**

Leistungsverhalten der Installation durch Vorgaben für „Business Goals“ festlegen

Work Load Manager setzt Vorgaben in Einstellungen um

Bei der Vielzahl der Einstlellungsmöglichkeiten ist es denkbar, daß eine Änderung keine Verbesserung, sondern eine Verschlechterung bringt.

WLM

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

WLM
OS/390 Work Load Manager (3)

Menge aller möglichen Arbeitsanforderungen in Dienstklassen (Service Classes) einordnen (klassifizieren).

Erfolgt nach Attributen wie

- **User ID**
- **Accounting Information**
- **Art des aufgerufenen Prozesses (Transaktionstyp, Stapel,)**
- **.....**

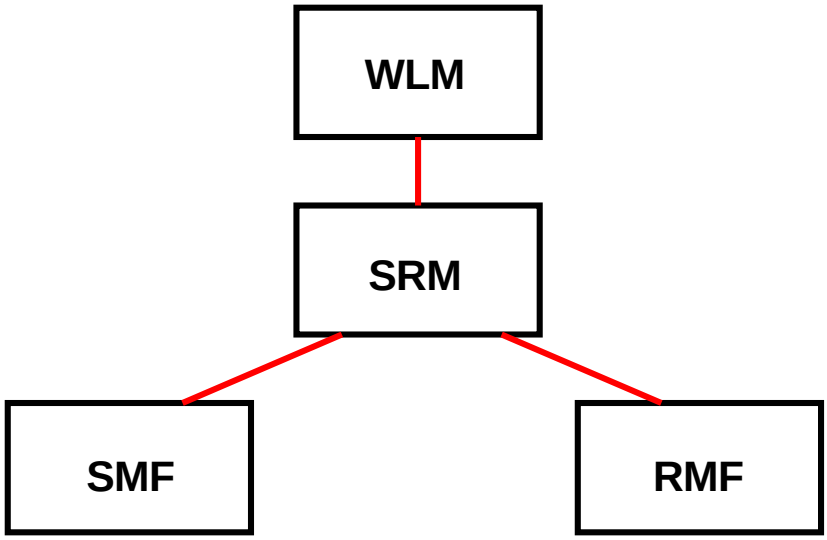
Jeder Dienstklasse sind „Ziele“ zugeordnet

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

Jede Dienstklasse besteht aus Perioden

- **Während einer Periode begrenzte Ressourcen verfügbar (CPU Zyklen, E/A Zugriffe, ...)**
- **Nach Ablauf der Periode i Migration nach i+1**
- **Unterschiedliche Ziele in jeder Periode**

WLM plaziert Arbeitsanforderungen so, daß die Wahrscheinlichkeit alle Ziele zu erreichen optimiert wird



Vier z/OS Work Load Management Komponenten

- **Work Load Manager**
  (Goal oriented) Work Load Manager

- **System Resources Manager (SRM) is a component of the System Control Program (SCP).**

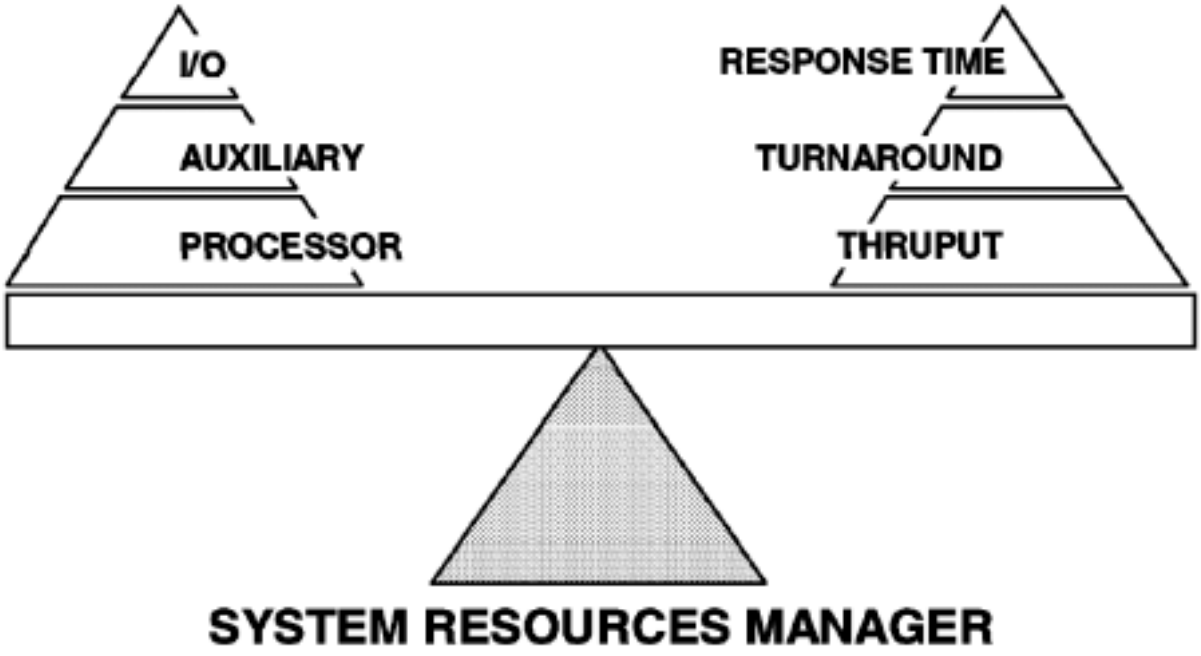
- **System Management Facility (SMF) collects system-related information about the configuration, the workload, and paging/swapping activities and writes it to SYS1.MANn data sets**

- **Resource Measurement Facility (RMF) is an optional feature for both z/OS and OS/390. RMF is supported by Tivoli Manager for z/OS und OS/390**

System Resources Manager

Die System Ressource Manager Komponente des Work Load Managers beobachtet für alle angeschlossenen Systeme:

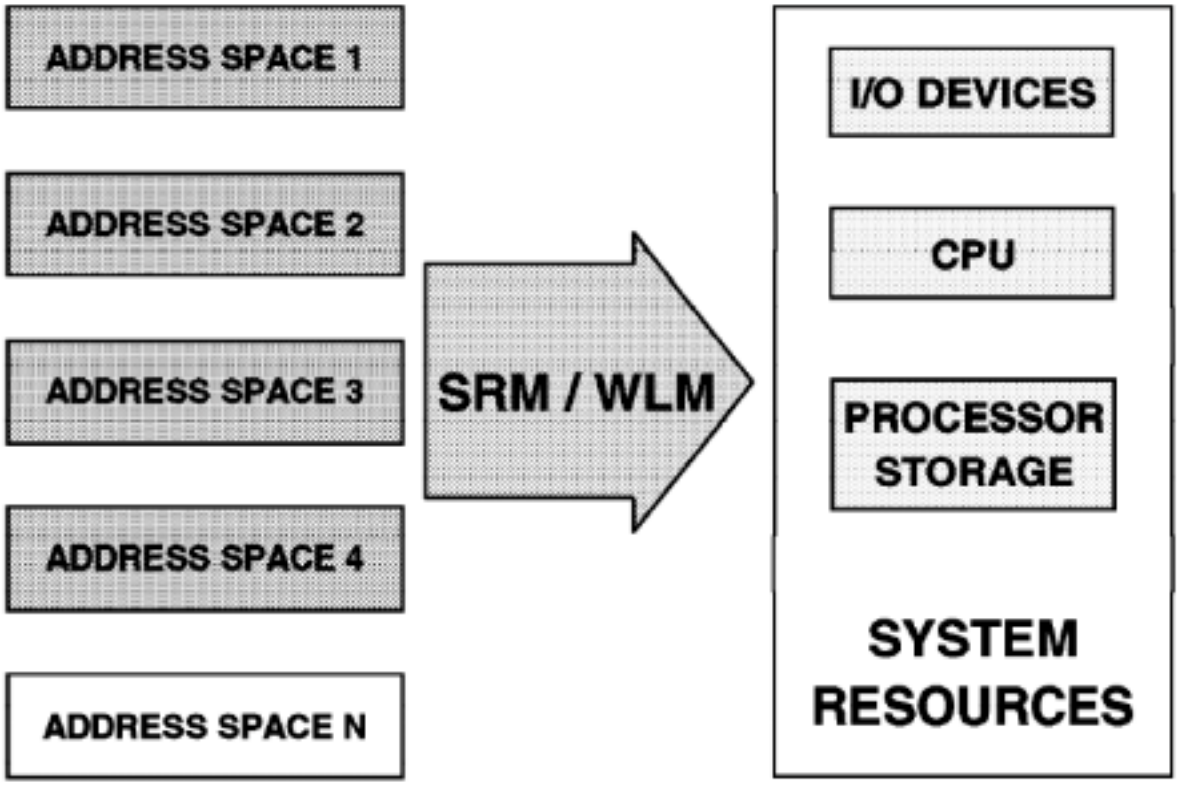
- **CPU Auslastung**
- **Hauptspeicher Nutzung**
- **E/A Belastung**



The System Resources Manager has two objectives:

- **To make sure the system resources are fully utilized but not overutilized.**
- **To try to give all users their fair share of system resources.**

SRM attempts to ensure optimal use of system resources by periodically monitoring and balancing resource utilization. If resources are underutilized, SRM attempts to increase the system load. If resources are overutilized, SRM attempts to reduce the system load.



System Resources Manager

SRM determines which address spaces, of all active address spaces, should be given access to system resources and the rate at which each address space is allowed to consume these resources. SRM bases its decision on two fundamental objectives:

1. **To distribute system resources among individual address spaces in accordance with the installation's response, turnaround, and work priority requirements.**

2. **To achieve optimal use of system resources in terms of system throughput.**

System Management Facility

Wie stellt die System Management Facility (SMF) fest, welche Betriebsmittel eine Workload benutzt und auf welche sie wartet.

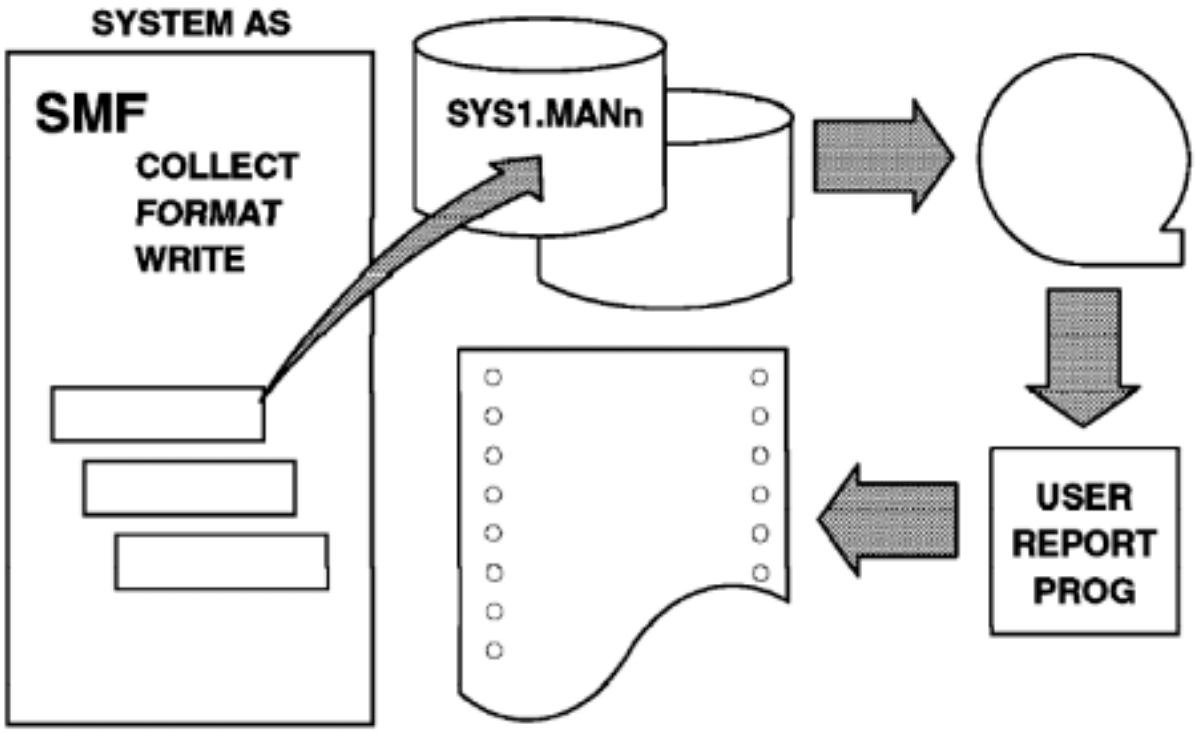
SMF sammelt Daten über den aktuellen Zustand aller verwalteten Betriebsmittel. Dies sind Information über

- **die Prozessoren,**
- **den Speicher und**
- **die Nutzung der Platteneinheiten ubd Kanäle.**

Zeiträume, in denen die Workload nicht gearbeitet hat und auch keine Anforderungen an das System gestellt hat, werden nicht berücksichtigt.

Dasselbe gilt für Zustände, die von SMF nicht erfasst werden können, wie zum Beispiel das Warten auf die Freigabe eines Locks, das durch eine Anwendung verwaltet wird.

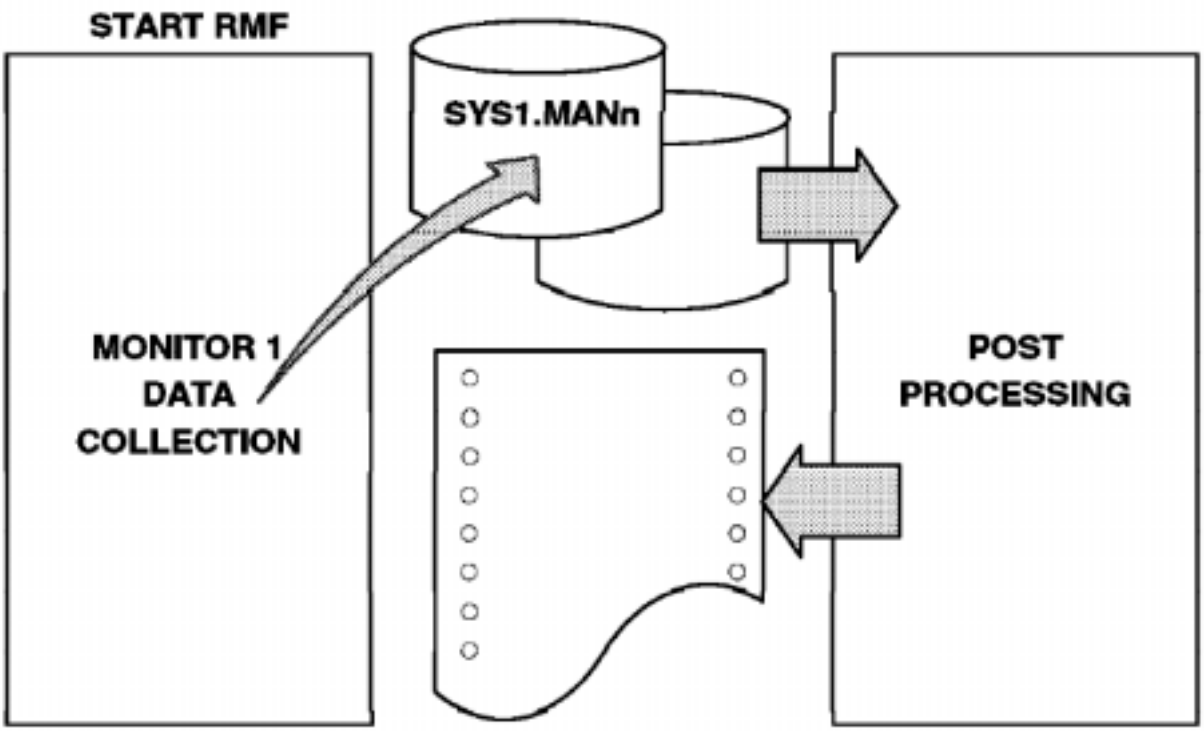
Aus den gesammelten Daten wird für jede Workload festgestellt, ob sie Betriebsmittel benutzt oder darauf wartet. Der Workload-Manager verwendet diese Daten, um den Zugang der Service-Klassen zu den Betriebsmitteln zu regeln.



Von SMF benutzte Input-Daten

System Management Facility (SMF) writes system-related information about the configuration, the workload, and paging/swapping activities to SYS1.MANn data sets (or similarly named data sets).

Job-related SMF records are also written to SYS1.MANn data sets. These records contain information about start time, stop time, processor activity, and storage usage for each job step and TSO session. User Report Programs analyse the data and produce reports.



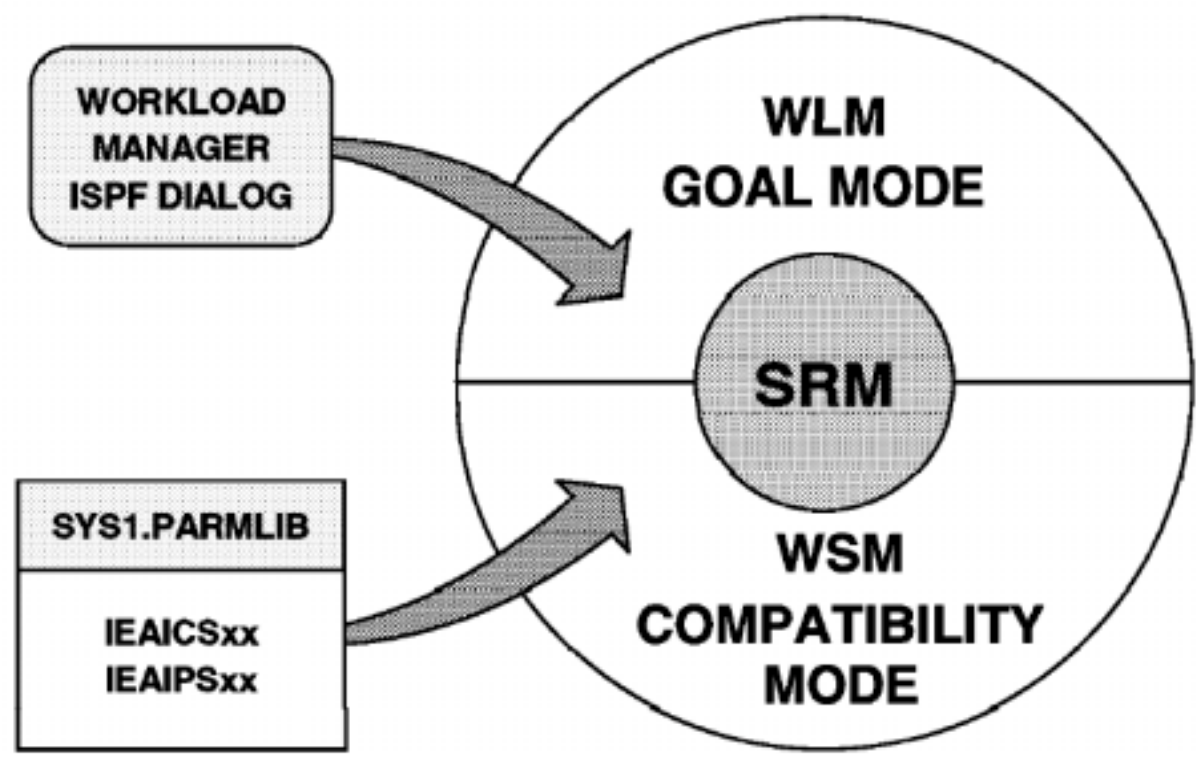
RMF Monitor

Resource Measurement Facility (RMF) is an optional feature for both z/OS and OS/390.

RMF is a started task that runs in its own address space. An RMF Monitor session collects system data (for example, CP utilization, DASD activity, processor storage utilization, and workload activity) over a period of time and provides printed reports. The user specifies the sample cycle, the interval, and the type of report (for example, CPU, Device, Path).

RMF is supported by Tivoli Manager for OS/390, which means that it can be accessed in three ways:

- **From TSO using the RMFMON command.**
- **From ISPF using the ISPF interface**
- **From the workstation using Tivoli Manager for OS/390.**

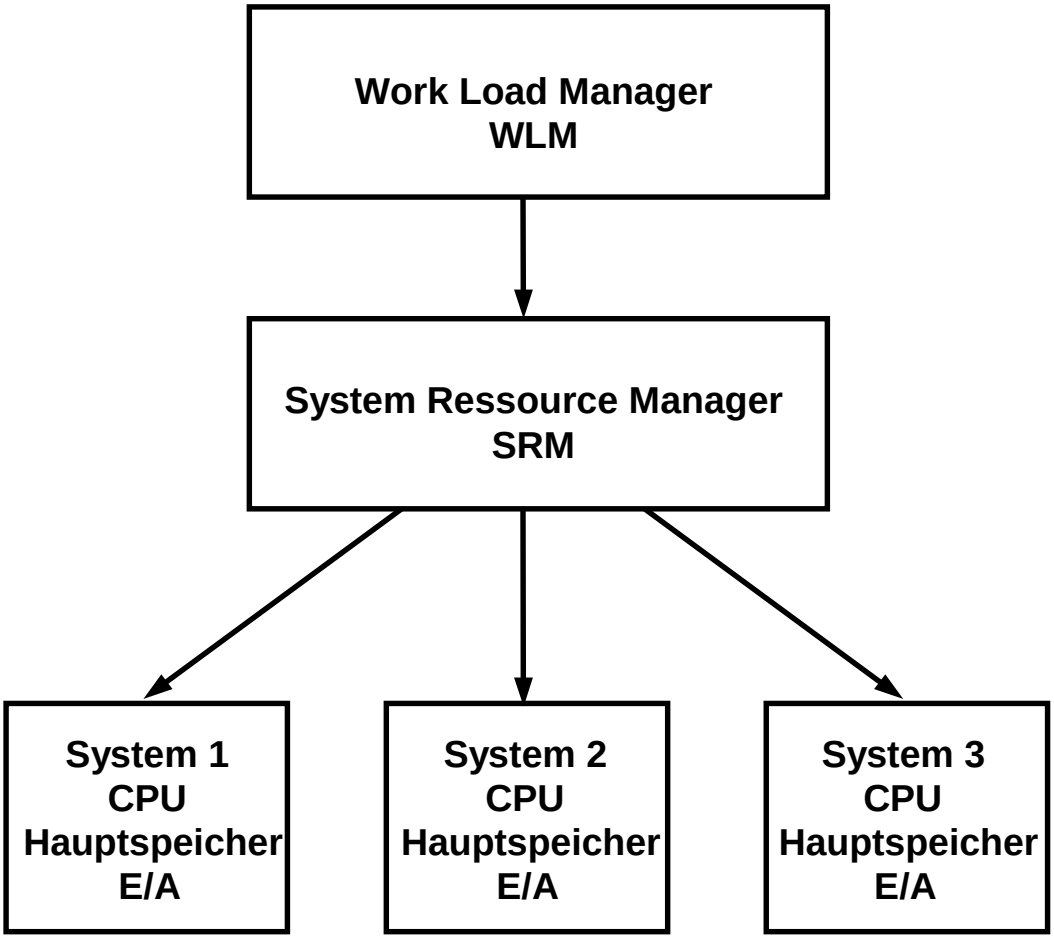


Der z/OS Workload Manager (WLM) erweitert den Funktionsumfang des SRM, und dehnt ihn auf den ganzen Sysplex aus. Er stellt eine Shell für den System Resource Manager (SRM) dar.

Workload management requires a shift in focus from

- **tuning at the system resource level, to**
- **defining performance expectations.**

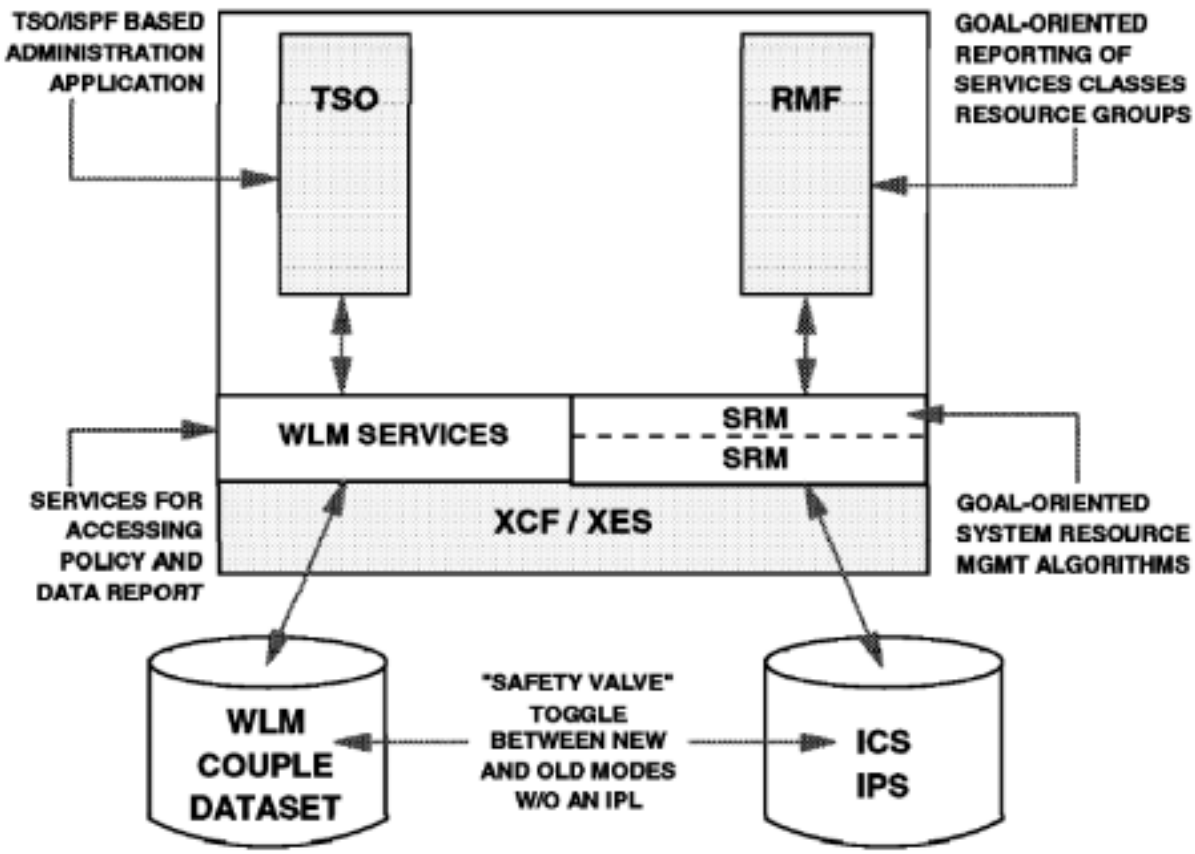
An installation defines performance expectations in the form of performance goals and the relative importance of the goals to each other. The system matches the available resources to the work to meet those goals, constantly monitoring and adapting job processing to achieve the performance goals defined.



SysPlex Betrieb

Die System Ressource Manager Komponente des Work Load Managers beobachtet für alle angeschlossenen Systeme:

- **CPU Auslastung**
- **Hauptspeicher Nutzung**
- **E/A Belastung**

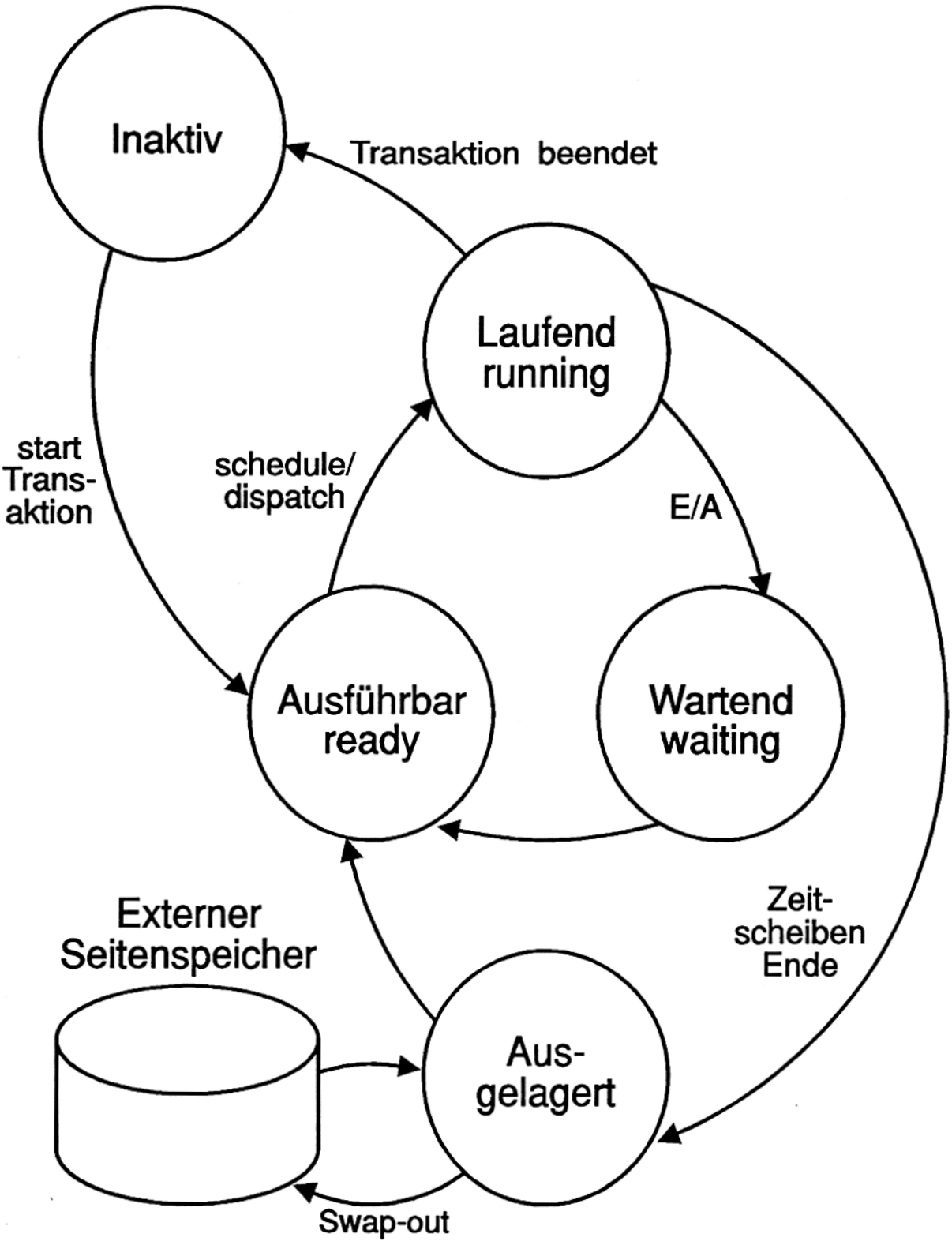


Each z/OS or OS/390 system contains the following functions relating to the workload management process:

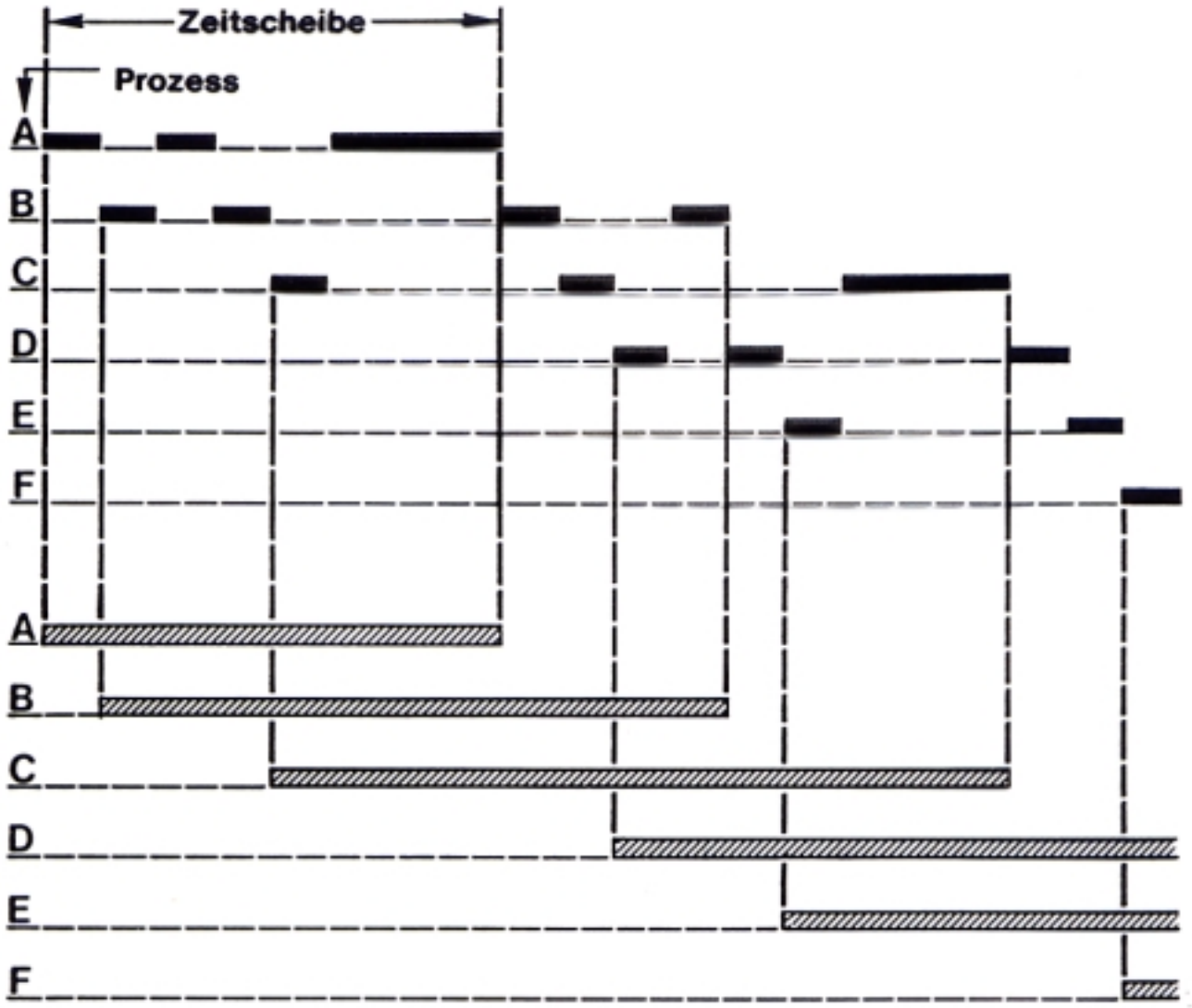
- **ISPF-based administrative application used to define and manage the various policies and the WLM couple data set.**
- **Workload management services to allow authorized applications (for example, CICS, IMS, VTAM) to cooperate with WLM in the management of the installation's resources to attain installation specified goals.**
- **Performance monitors and reporting programs (such as RMF) use WLM services to collect performance data and report how well the installation-specified goals are being achieved.**
- **In addition, a comprehensive set of RMF reports, containing both single-system and sysplex-wide data for workload management, is available.**

SRM Techniques

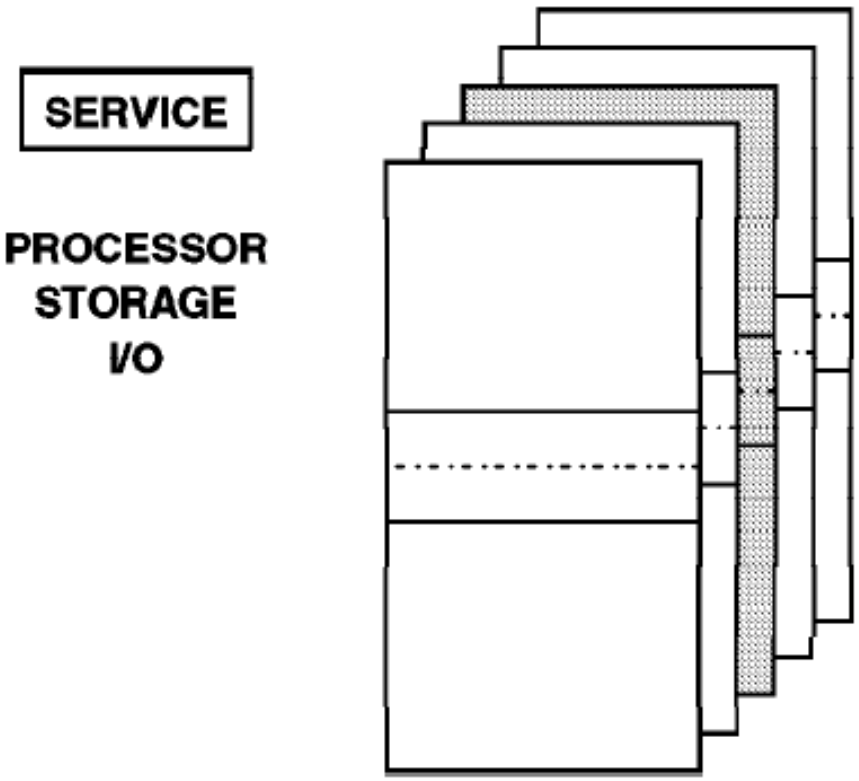
- Adjust multiprogramming level
- Swapping
- Steal page frames
- Restrict address space creation
- Adjust dispatching priority



Auslagern von Prozessen



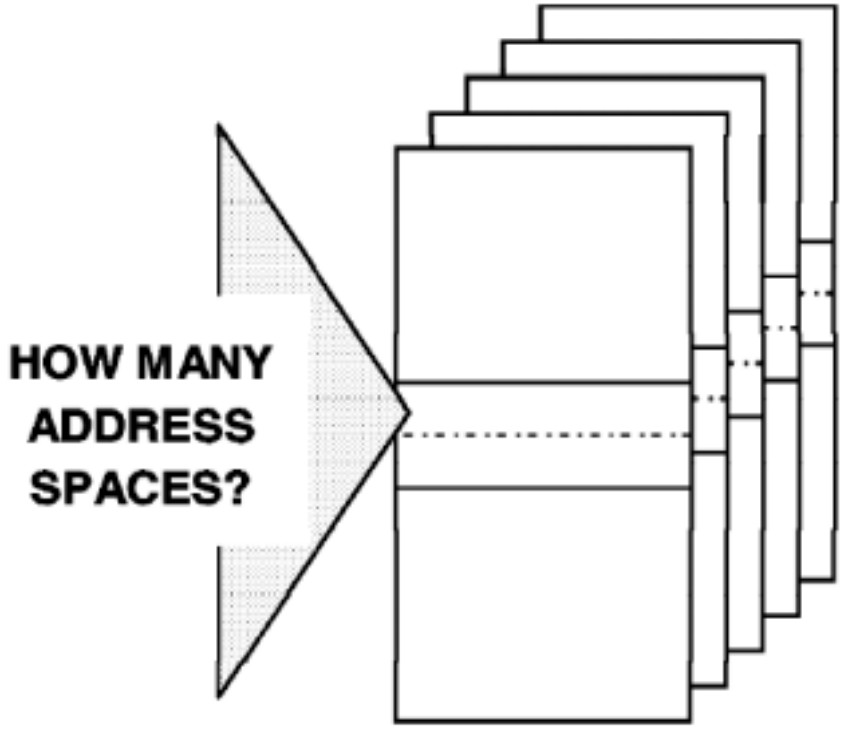
Multiprogrammierung innerhalb von Zeitscheiben



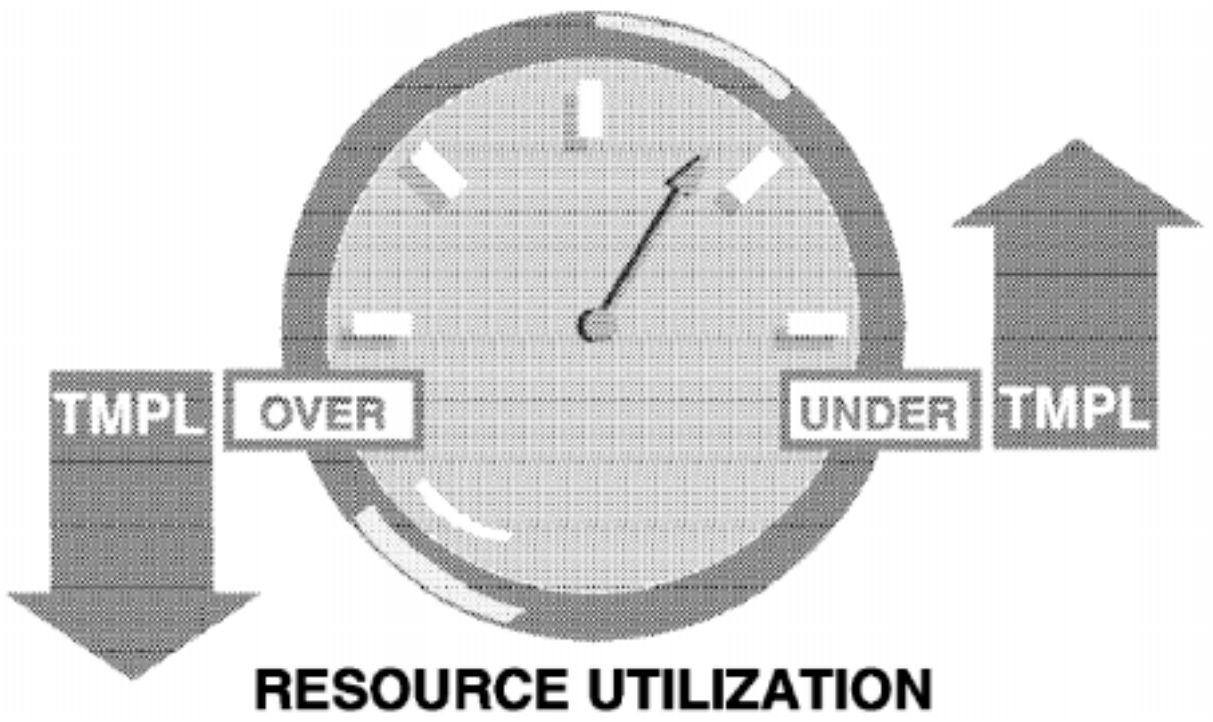
Swapping is making an address space either dispatchable or nondispatchable. The main technique used by SRM to achieve its objectives is the swapping of address spaces.

In addition to swapping because the Target Multiprogramming Level (TMPL) has changed, these are the the normal reasons why address spaces are swapped:

- **A TSO user address space is waiting for input from the terminal.**
- **There is a critical shortage of a resource, such as central storage.**



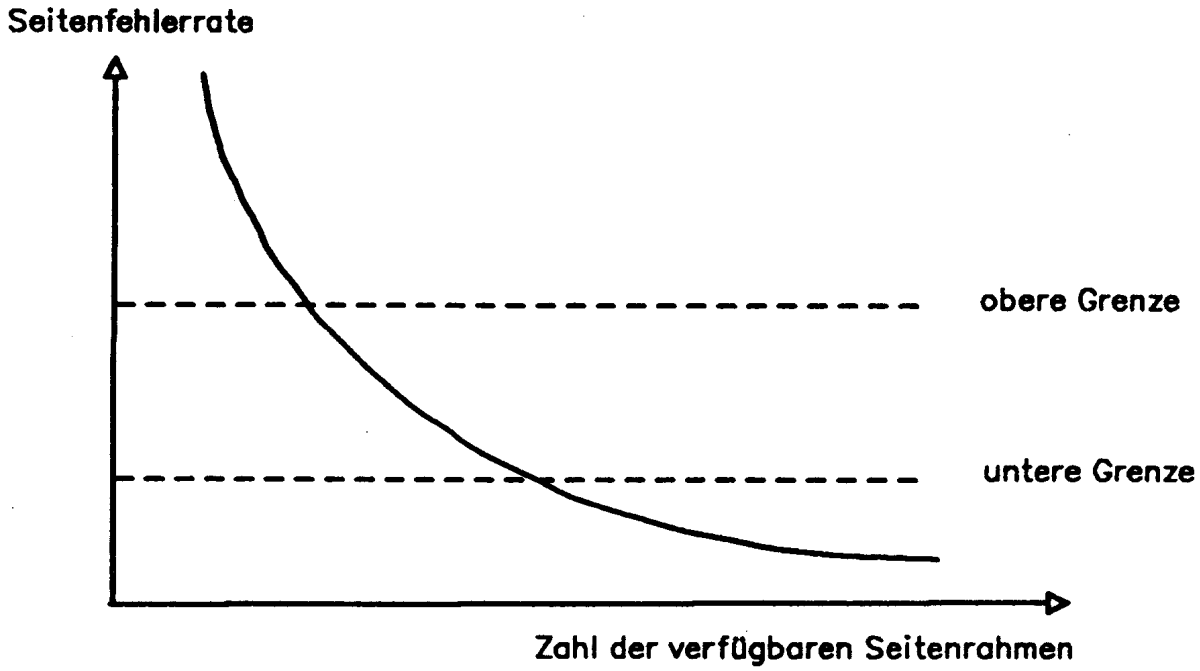
SRM must determine the most efficient number of address spaces to reside in central storage at any one point in time. This is referred to as the Multiprogramming Level (MPL).



The Target Multiprogramming Level (TMPL) is the number of swapped-in address spaces SRM believes the system can handle based on the available resources.

This is changing constantly.

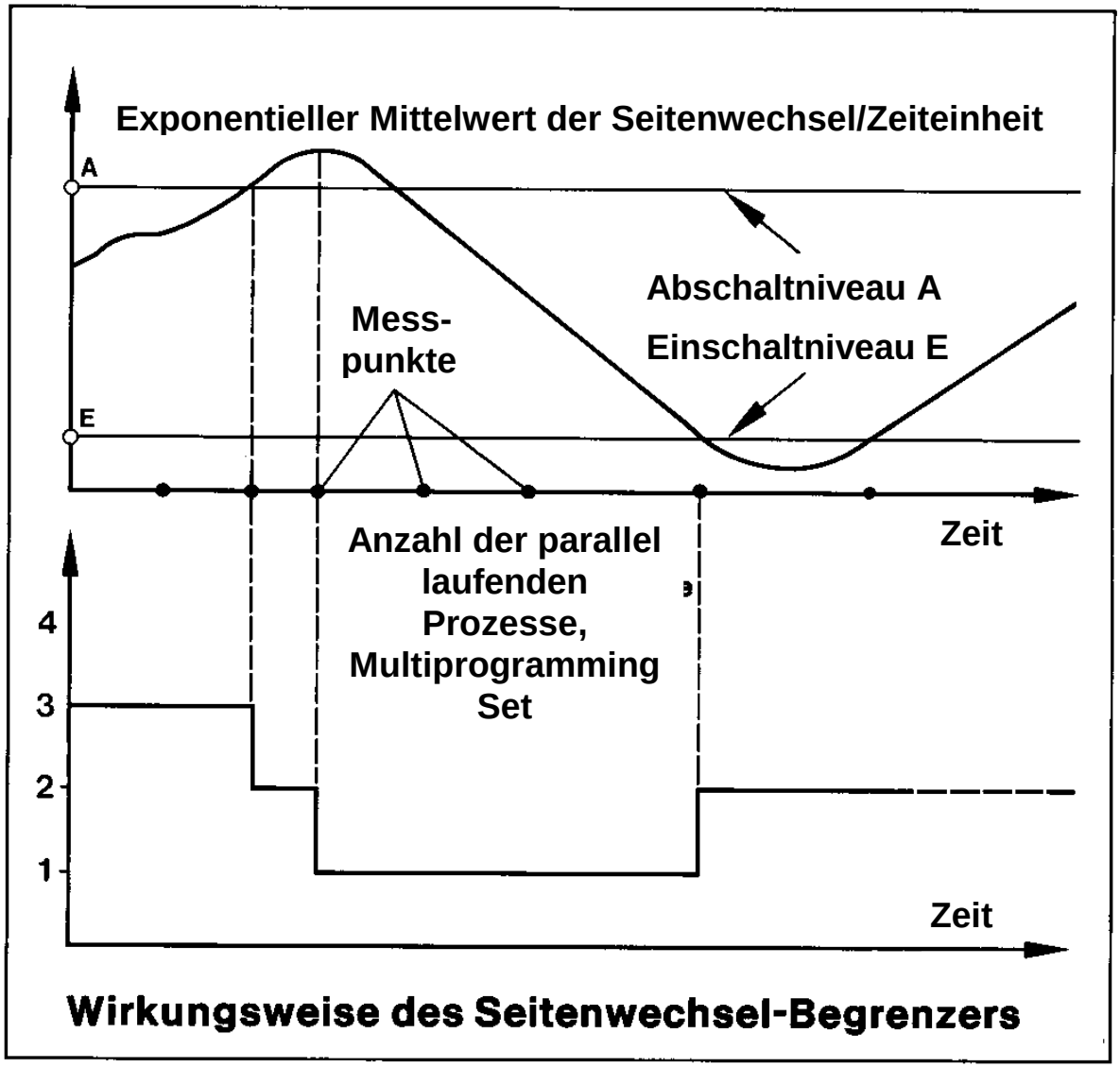
The Current Multiprogramming Level (CMPL) is the actual number of address spaces swapped in and competing for resources.



Seitenfehler - Begrenzung

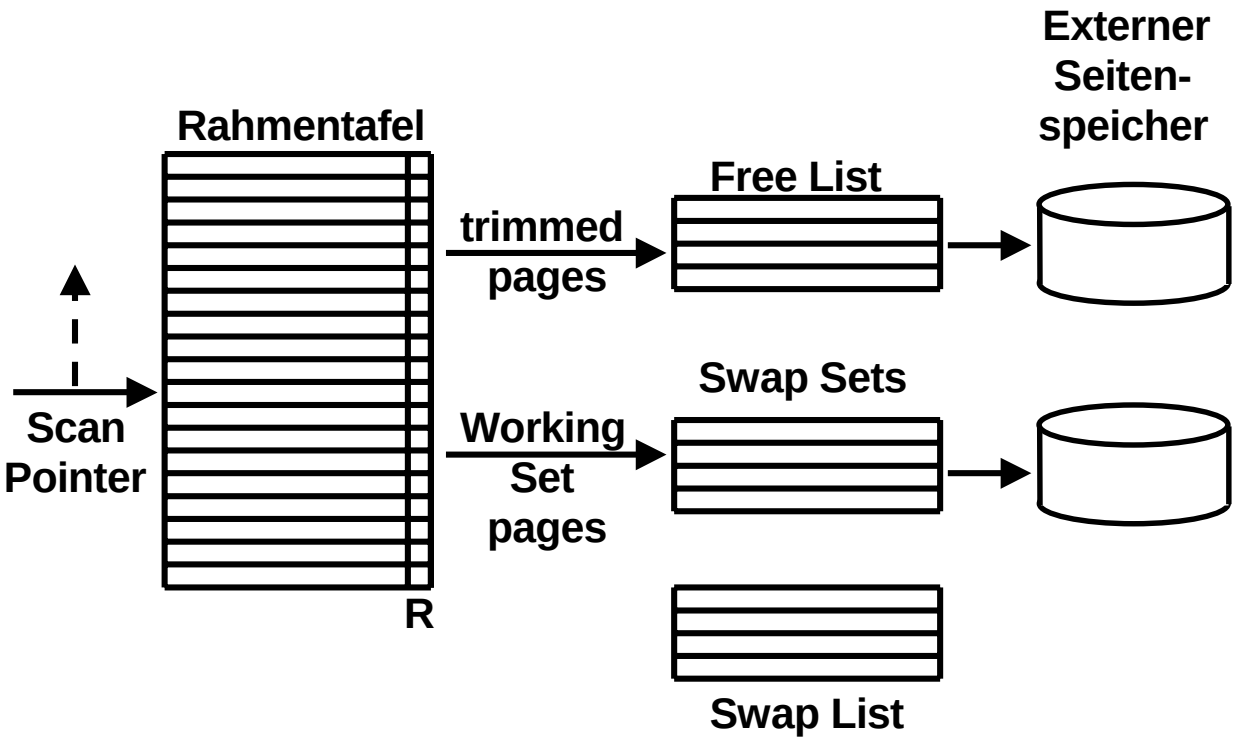
Bei Überschreiten der oberen Grenze werden solange Prozesse niedriger Priorität deaktiviert, bis die Seitenfehlerrate auf einen akzeptablen Wert sinkt

Flattern (Thrashing)



Anzahl der gleichzeitig (multiprogammiert) aktiven Prozesse (multiprogramming set) begrenzen

if paging rate > maxvalue
then drop process from queue



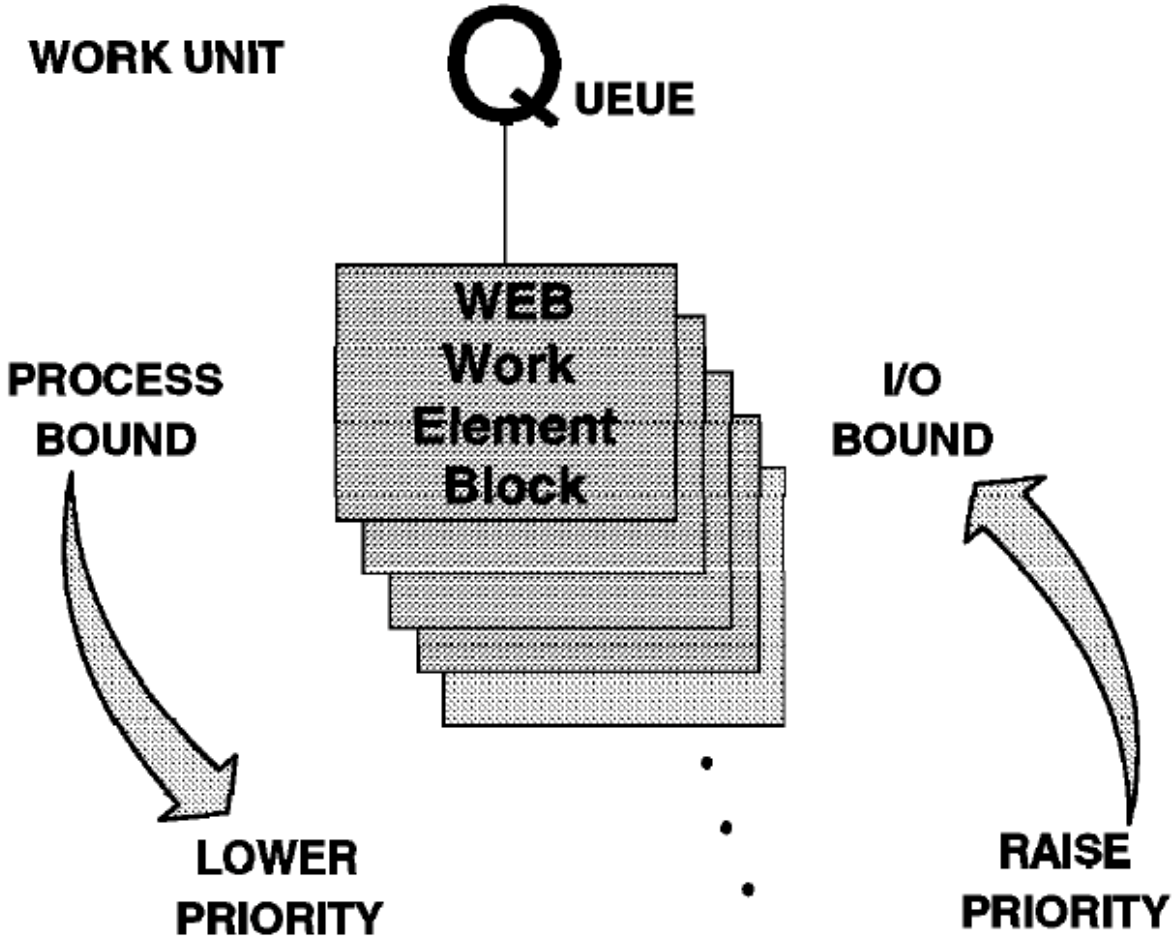
Der Scan Pointer wird um +1 bei jedem Seitenzugrihh hochgezählt. das dortige R-Bit wird auf 0 gesetzt. Im Eintrag der referenzierten Seite wird das R-Bit auf 1 gesetzt.

In der Free List stehen Seiten, deren Referenz Bit (R-Bit) = 0 ist.

Sinkt die Anzahl der Seiten in der Free List unter einem kritischen wert, so wird 1 Swap Set (z.B. 16 Seiten) auf den externen Seitenspeicher ausgelagert (block Paging, physical Swap-Out. Der entsprechende Prozess steht in der Swap List).

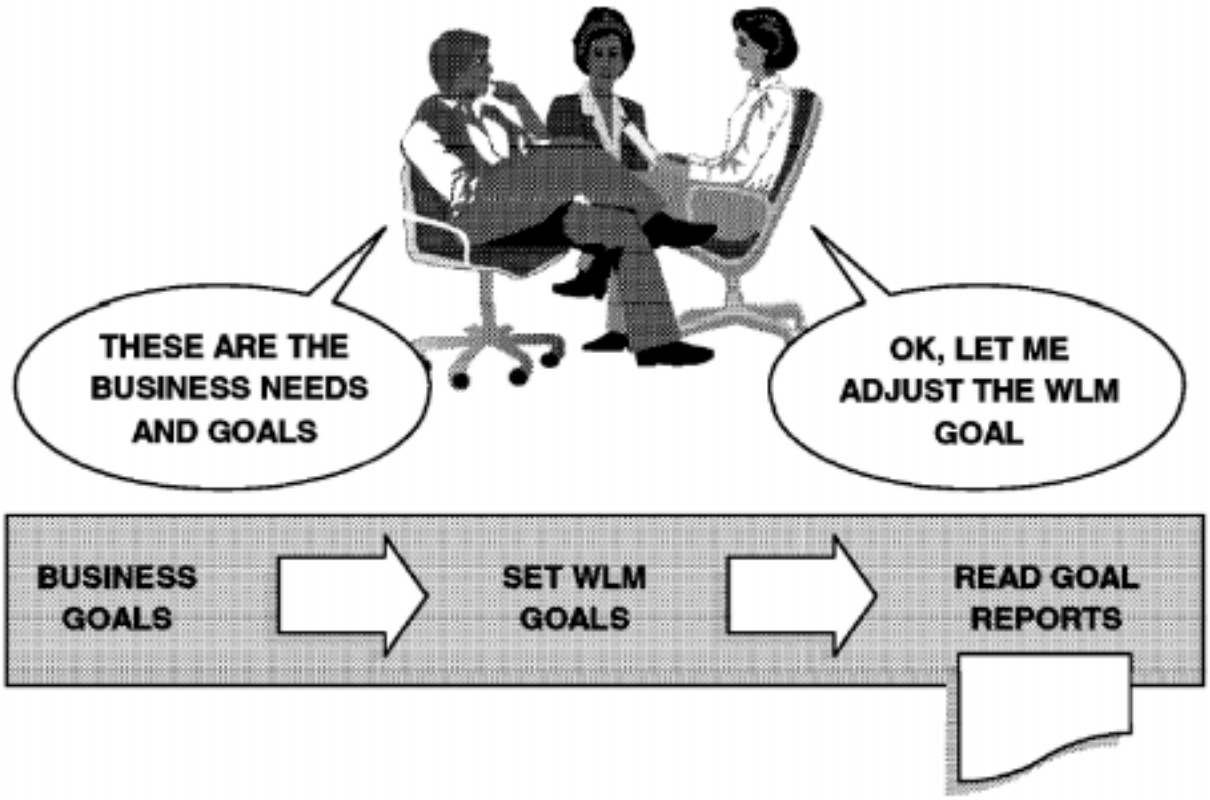
Ist auch der Swap Set leer, erfolgt ein Page Steal

1. Für zusätzlichen Bedarf an Hauptspeicherrahmen steht zunächst die Free List zur Verfügung.
2. Sinkt die Zahl der Rahmen in der Free List unter einen kritschen Wert, so wird 1 Swap Set (z.B. = 16 Seiten, Big Page) auf den Swapping Device (Plattenspeicher) ausgelagert (physical Swap-out). Der entsprechende Prozess steht in der Swap List. Ein spezifischer Prozess verliert alle Swap Sets, ehe der nächste in der Swap List betroffen wird.
Wird der betroffene Prozess wieder aktiv, erfolgt zum Anfang ein physikalisches Swap-in.
3. Ist auch der Swap Set leer, erfolgt ein Page Steal mit Hilfe des Vorrückens des Scan Pointers.



Dispatching priority determines the position of a work element block on the work unit queue. Dispatching priority can be assigned by DPRTY= in the JCL, but most installations allow SRM to control dispatching priorities.

SRM will raise the priority of a Work Element Block (WEB) representing an I/O bound job, and lower the priority of a WEB representing a processor-bound job to achieve optimum multi-programming overlap.



Goal Orientatierung

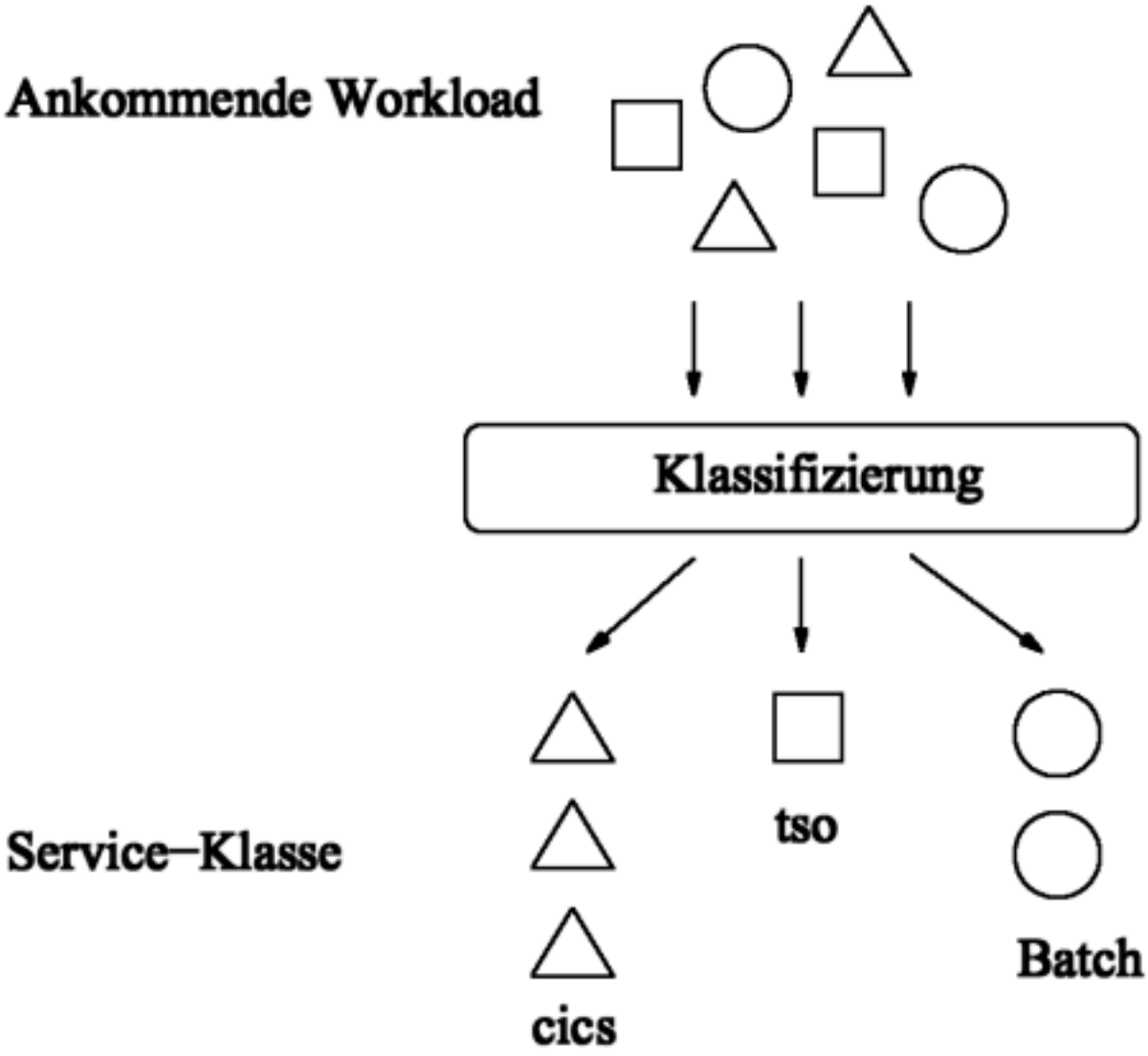
Es sit die Aufgabe des Workload Manager, System Ressource Manager (SRM) Parameter in Geschäftsbegriffe (business terms) zu übersetzen.

Das Leistungsverhalten der Installation wird durch Vorgaben für „Business Goals“ festlegt. Der Work Load Manager setzt die Vorgaben in Einstellungen für den SRM um.

Bei der Vielzahl der Einstellungsmöglichkeiten ist es denkbar, dass eine Änderung keine Verbesserung, sondern eine Verschlechterung bringt.

WLM

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



Service Klassen

Service-Klassen sind Einheiten von Workload mit sehr ähnlichen Charakteristiken, für die die Ziele definiert werden. Die Service-Klasse stellt das Grundkonstrukt im *Goal Mode* dar. Sie ist das Ergebnis der Klassifizierung der Workloads mit unterschiedlichen Leistungsmerkmalen in Gruppen, die mit Ziel-Vorgaben versehen werden können.

Einordnung in Dienstklassen (Classification Rules)

Menge aller möglichen Arbeitsanforderungen in Dienstklassen (Service Classes) einordnen (klassifizieren). Die Klassifikation basiert auf den Attributen einer individuellen Arbeitsanforderung. Dies kann sein:

- **User ID**
- **Art des aufgerufenen Prozesses (Transaktionstyp, Stapel,)**
- **die Arbeits-Umgebung oder das Subsystem**
- **Accounting Information**
- **.....**

Jeder Dienstklasse sind „Ziele“ zugeordnet

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

Jede Dienstklasse besteht aus Perioden

- **Während einer Periode begrenzte Ressourcen verfügbar (CPU Zyklen, E/A Zugriffe, ...)**
- **Nach Ablauf der Periode i Migration nach i+1**
- **Unterschiedliche Ziele in jeder Periode**

WLM plaziert Arbeitsanforderungen so, daß die Wahrscheinlichkeit alle Ziele zu erreichen optimiert wird

Service Definition

Eine Workload-Manager Service Definition wird mittels einer speziellen administrativen Anwendung, der *WLM Administrative Applicatio*n, erstellt und abgeändert.

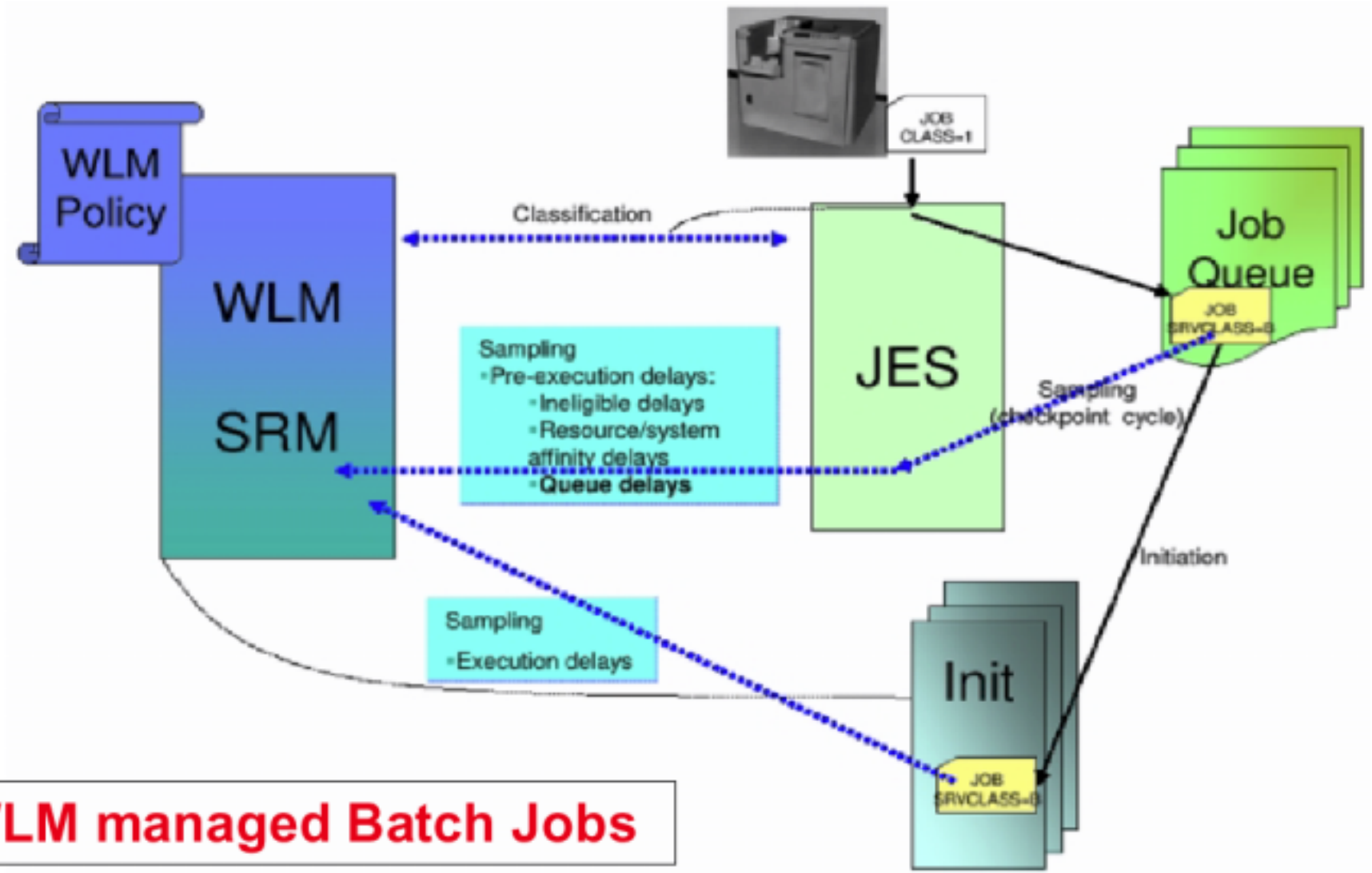
Sie kann unter *Interactive System Productivity Facility* (ISPF von TSO-Benutzern gestartet werden.

Der wichtigste Punkt bei der Definition einer Service-Klasse ist es, festzulegen, wie wichtig sie ist, um die übergeordneten Geschäftsziele zu erreichen.

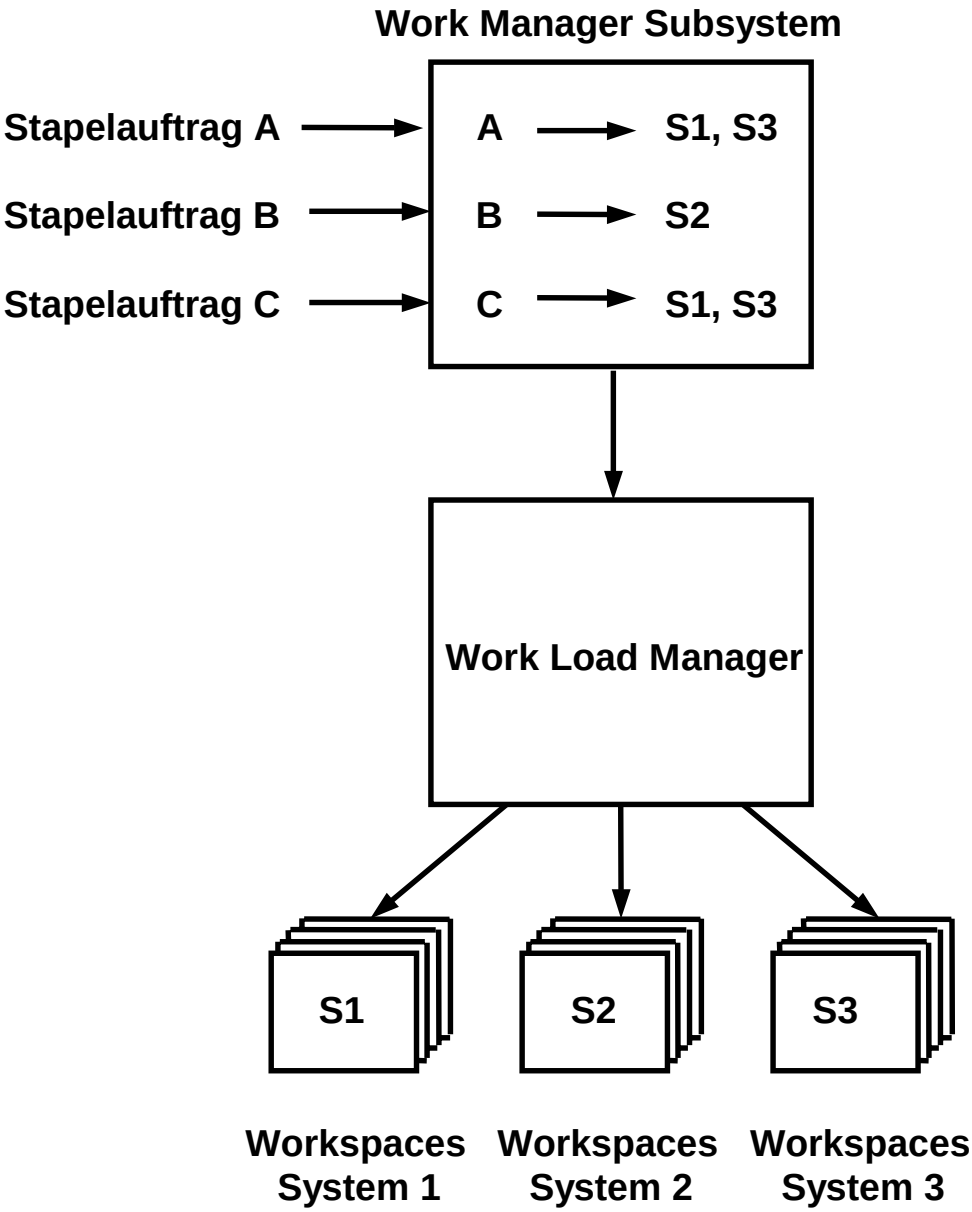
Die Wichtigkeit wird dargestellt, indem eine *Business Importance* für die Service-Klasse definiert wird. Diese *Business Importance* ist später für das System der wesentliche Entscheidungsfaktor, wenn der Zugang zu den Betriebsmitteln geregelt werden muss.

Der Workload-Manager unterstützt beispielsweise folgende Subsysteme :

- **IMS**
- **JES2/JES3**
- **TSO/E**
- **CICS**
- **OMVS**
- **DB2**



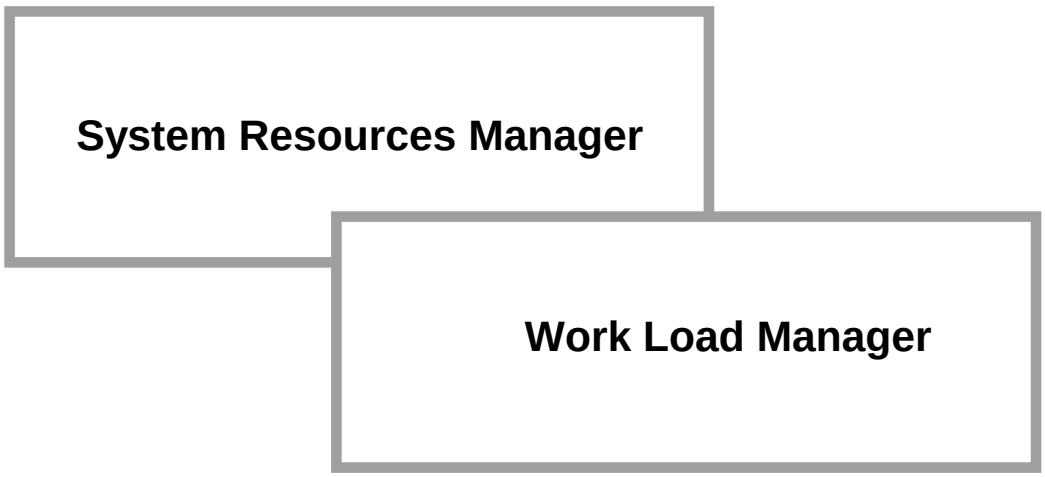
WLM managed Batch Jobs



Zuordnung von Aufträgen zu Systemen

Einstellungen der einzelnen Systeme dynamisch an sich ändernde Belastungen anpassen und automatisch justieren

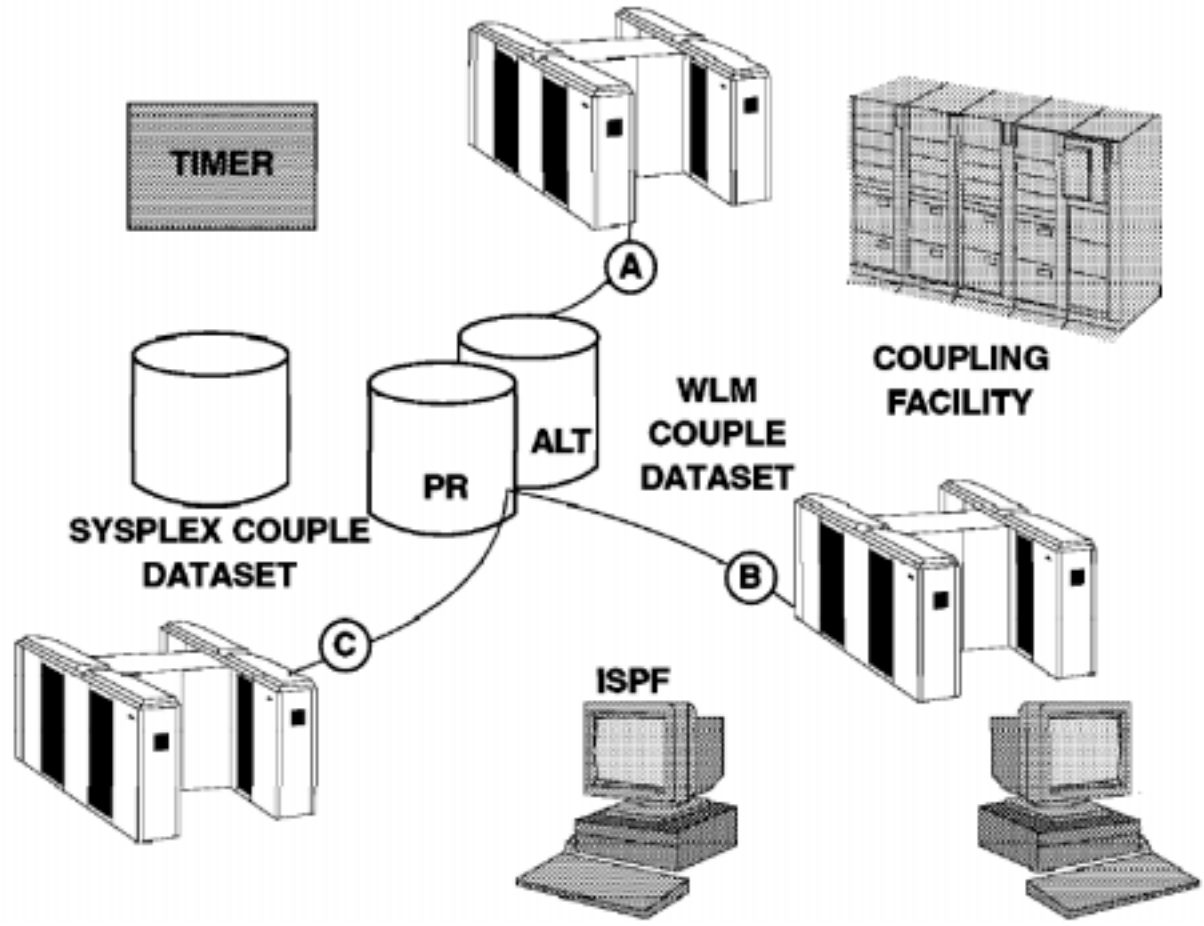
Bei widersprüchlichen Anforderungen (Regelfall) verfügt der WLM über Algorithmen, einen möglichst optimalen Kompromiss zu erreichen



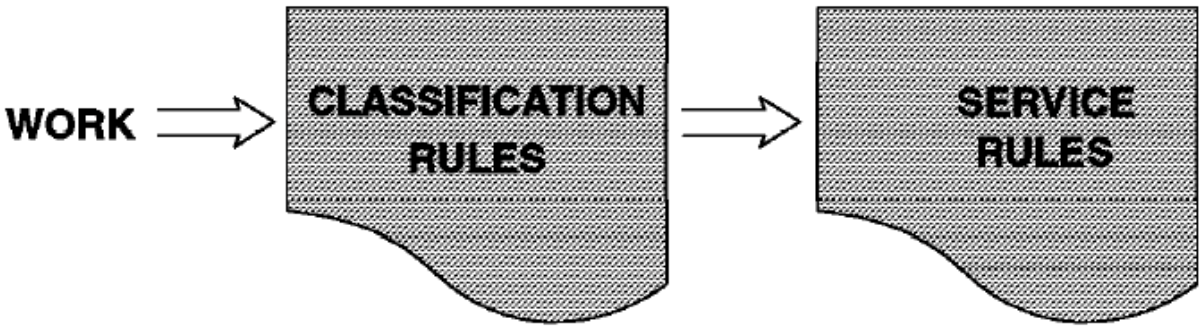
Der z/OS System Resource Manager (SRM) misst die gesamte Systemleistung und beobachtetdie Nutzng der Systemresourcen (z.B. CPU, Hauptspeicher, Ein/Ausgabe).

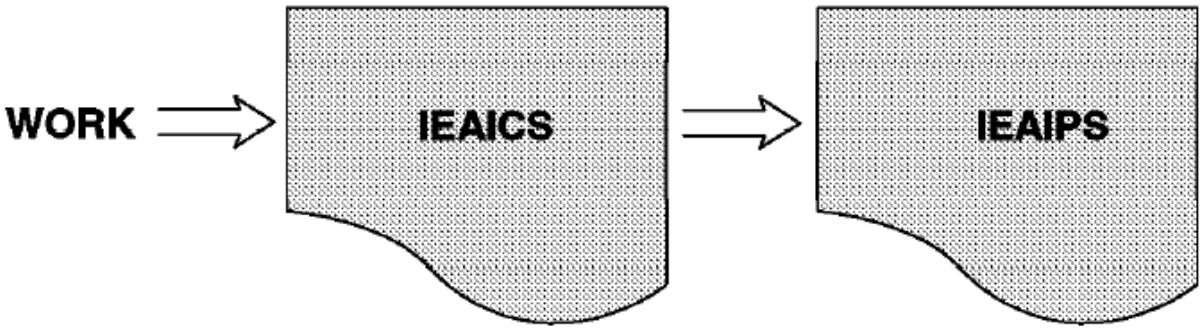
Engpässe an Resourcen werden in der Regel durch eine Verringerung des Multiprogramming Sets und durch das Swapping von Adressenräumen aus dem Haputspeicher in den externen Seitenspeicher aufgelöst.

Der z/OS Workload Manager (WLM) erweitert den Funktionsumfang des SRM, und dehnt ihn auf den ganzen Sysplex aus



To use WLM in goal mode, you must be in sysplex mode (sysplex or monoplex). This then assumes that you have shared DASD configuration, sysplex timer, and the configuration couple dataset. You will also have a WLM couple dataset. It provides a location to keep the service policy which is common to all systems in the sysplex. It need not be DFSMS managed. There is an ISPF administrative application to create a Service Definition and place it on the WLM couple dataset. Any system may run this application; you may manage the sysplex from any convenient location. In addition, this application may be used to format the dataset. Initial service policies may be defined and kept as ISPF datasets and placed on the WLM dataset when desired to activate the new one.





For any system there needs to be a way to identify various groups of work based on importance, type, resource usage, and other criteria. Once these categories have been identified, goals may be set for their processing priorities. For us old timers there is a parallel between the old and the new way. That is the IEAICS and IEAIPS members of SYS1.PARMLIB will be replaced by classification rules and service rules (or policies). A new hierarchy will be used:

- **Service Definition**
- **Classification rules**
- **Policy (or Service Policy)**
- **Workload**
- **Service Class**
- **Service Class Period**
- **Resource Group**

Arten von Zielen

Für die Definition von Zielvorgaben bietet der z/OS-Workload-Manager drei Arten von Zielen:

- **Response Time Goals**
- **Execution Velocity Goals**
- **Discretionary**

Arten von Zielen

Response Time Goals

Unter einem Response Time Goal versteht man die Zeitspanne zwischen Start und Fertigstellung einer Workload, die in Sekunden pro Programm gemessen wird. Als Zeit wird die vollständige Verweildauer inklusive der Zeit, in der die Workload nicht arbeitet, betrachtet. Dadurch kann man die Wartezeit eines Benutzers erkennen.

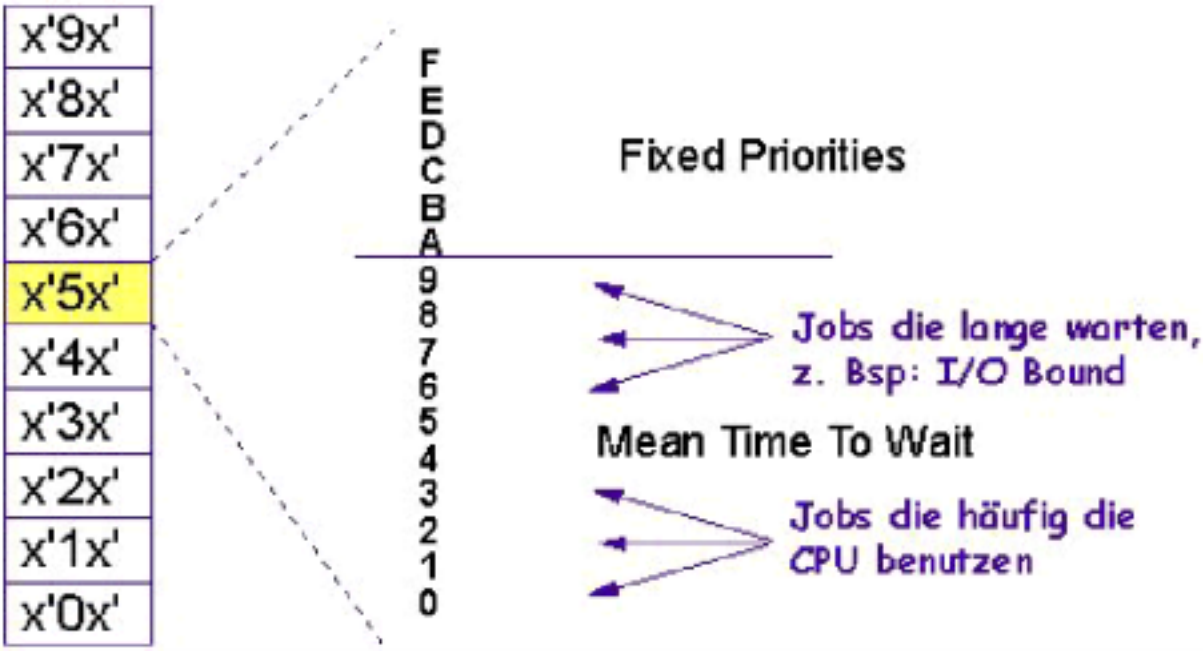
Execution Velocity Goals

Wenn zu wenige Workloads in einer Service-Klasse enden, kann ein Response Time Goal nicht verwendet werden. Um für derartige Workloads Ziele definieren zu können, kann ein Execution Velocity Goal verwendet werden. Bei einem Execution Velocity Goal werden nur die Zustände betrachtet, bei denen eine Workload Betriebsmittel benutzt (using) oder darauf wartet (delay).

Discretionary

Für eine Gruppe von Workloads existieren keine bestimmten Zielvorgaben. Diese Workloads erhalten nur dann Betriebsmittel, wenn diese ausreichend vorhanden sind, und sie sind die Ersten, die keinen Zugang mehr erhalten, wenn es eng im System wird. Es tritt ein Workload Management Regelmechanismus in Kraft, der den Betriebsmittelzugang für ServiceKlassen mit Zielvorgaben begrenzt, wenn diese ihre Ziele deutlich übererfüllen.

Prioritäten von Prozessen



Für das Scheduling werden alle Prozesse 10 Gruppen zugeordnet, wobei jede Gruppe noch einmal in 16 Prioritäten unterteilt ist.

Die unteren 10 Prioritäten werden für einen *Mean-Time-To-Wait* Algorithmus verwendet. Es wird der Tatsache Rechnung getragen, dass Workloads unterschiedliches Verhalten bei der Benutzung der Prozessoren und des I/O-Subsystems zeigen. Dieses Verhalten ändert sich mit der Abarbeitungszeit von Programmen und trifft insbesondere auf langlaufende Batch-Jobs zu, die unter Umständen eine längere Zeit Daten ein-oder auslesen, im Wechsel mit Perioden, in denen sie Rechenvorgänge ausführen. Um dies zu berücksichtigen, kann eine Installation eine Dispatching Priority festlegen, die durch den Buchstaben M und eine Zahl charakterisiert wird.

SRM beobachtet die Address Spaces in diesen Gruppen und verändert die Dispatching Priorities über einen Alterungsprozess, abhängig davon, ob die Address Spaces die Prozessoren benutzten oder auf die Abarbeitung von I/O-Operationen warten. Address Spaces, die lange Zeit auf I/O-Operation warten, bekommen eine Dispatching Priority am oberen Ende der Gruppe, und solche, die häufig die Prozessoren benutzen, liegen am unteren Ende. Für alle anderen Arten von Arbeit werden feste Dispatching Priorities vergeben, wobei eine hohe Zahl einen besseren Zugang zum Prozessor bedeutet. Aus diesem Grund

Work Load Manager Operation

Response Time oder Antwortzeit drückt den Wunsch aus, dass die Zeit, die Transaktionen im System verweilen, einem vorgegebenen Wert entsprechen.

WLM sammelt alle 250 Millisekunden den aktuellen Zustand aller durch WLM verwalteten Betriebsmittel. Dies sind die Prozessoren, der Speicher, die Benutzung der Platteneinheiten und die Benutzung von WLM für die von Anwendungen verwaltete Warteschlangen.

Aus den gesammelten Daten wird für jede Arbeitseinheit oder Work Unit festgestellt ob sie ein Betriebsmittel benutzt (Using) oder darauf wartet (Delay). Diese Daten werden verwendet, um die Verteilung von Betriebsmitteln auf Service Classes vorzunehmen. Da sie die Basis für die WLM-Algorithmen darstellen, können sie auch für die Definition von Zielen verwendet werden.

Die Zieldefinition legt den Anteil der akzeptablen Wartezeit bei der Ausführung von Programmen fest. Dies wird mit Execution Velocity bezeichnet

Es gibt eine Gruppe von Arbeit, für die keine bestimmte Zielvorgabe existiert. In z/OS Systemen werden diese Gruppen von Arbeit als *Discretionary* bezeichnet. Sie erhalten Betriebsmittel nur dann, wenn diese ausreichend vorhanden sind, und sie sind die Ersten, die keinen Zugang mehr erhalten, wenn es eng im System wird.

WLM Service Goals

Durchschnittliche Antwortzeit (Average Response Time)

Avg. Resp. Time = 1 s

Prozentsatz Antwortzeit (Percentile Response Time)

% Resp. Time: 90 % < 0,5 s

Velocity = theoretisch beste Zeit / wirklich verbrauchte Zeit

$$\text{Velocity} = \frac{\text{CPU + I/O}}{\text{CPU + I/O + paging + MPL + swapin + queues}}$$

Beliebig (Discretionary)

MPL = Multiprogramming Level

Execution Velocity

Execution Velocity ist wie folgt definiert:

$$\text{Execution Velocity} = \frac{\text{Total Usings}}{\text{Total Usings + Total Delays} \times 100}$$

Wie man aus der Formel erkennt, legt die Zieldefinition den Anteil der akzeptablen Wartezeit bei der Ausführung von Programmen fest.

***Execution Velocity* lässt sich einfach durch ein Beispiel veranschaulichen. Man nehme an, dass eine Workload fünf Mal auf Betriebsmittel wartet und fünf Mal Betriebsmittel benutzt.**

Man erhält man eine *Execution Velocity* von 50.

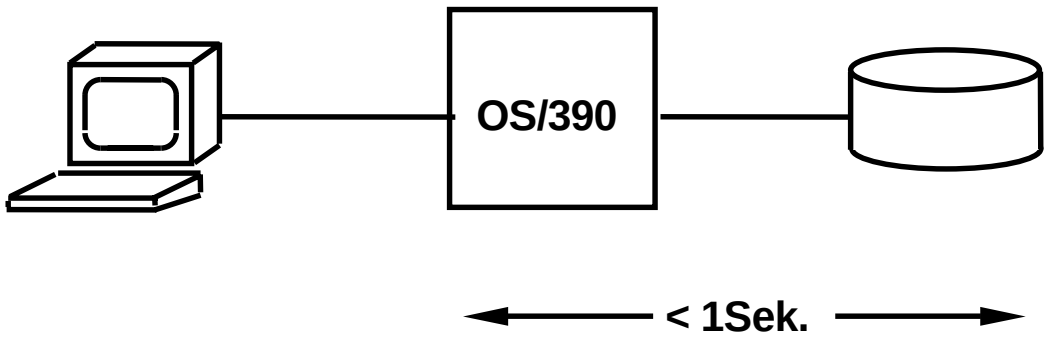
$$\text{Execution V elocity} = \frac{\text{5 Usings}}{\text{5 Usings + 5 Delays} \times 100} = 50$$

Der Wert der Execution Velocity liegt immer zwischen 1 und 100.

z/OS Work Load Manager (WLM)

Ziel orientiert, z.B. Antwortzeit für 90% aller Transaktionen < 0,3 Sek.

(keine end-to-end Antwortzeit, OS/390 steuert nicht die Netzwerk Verzögerungszeiten)



Für Jobs mit niedriger Priorität wird der Durchsatz optimiert. Z.B. erhalten hier Jobs mit viel E/A Aktivität eine höhere Priorität als CPU intensive Jobs.

Wenn erforderlich werden Jobs mit niedriger Priorität ausgelagert (expanded storage oder Plattenspeicher). Subsysteme wie CICS und DB2 sind hiervon ausgenommen.

Performance Index

Frage; Wie erfüllen Service-Klasse ihre Ziele und wie verhalten sie sich im Vergleich zu anderen Service-Klassen.

Workload-Manager-Algorithmen lösen die dieses Problem durch die Definition eines Performance Indexes (PI).

Definition des Performance Index für das Execution Velocity Goal:

$$PI = \frac{\text{Execution Velocity Goal}}{\text{Aktuell erreichte Execution Velocity}}$$

Definition des Performance Index für das Response Time Goal:

$$PI = \frac{\text{Aktuelle Response Time}}{\text{Response Time Goal}}$$

Der Performance Index für Average Response Time Goals und Execution Velocity Goals kann aus den Zieldefinitionen und den aktuell gemessenen Werten ermittelt werden.

Ziel - Erfüllung

Definition des Performance Index für das Execution Velocity Goal:

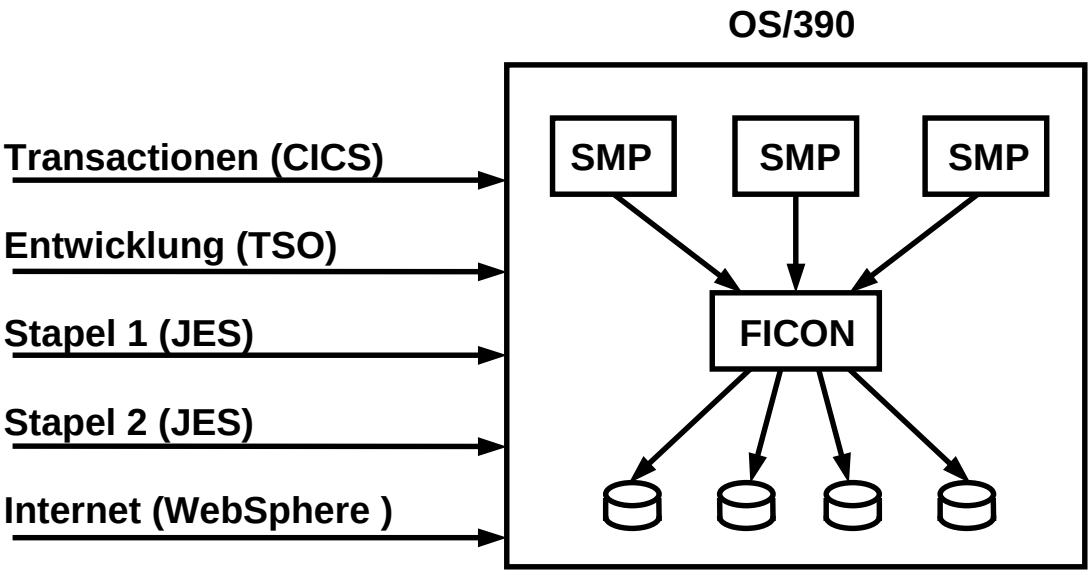
$$PI = \frac{\text{Execution Velocity Goal}}{\text{Aktuell erreichte Exec Velocity}}$$

Definition des Performance Index für das Response Time Goal:

$$PI = \frac{\text{Aktuelle Response Time}}{\text{Response Time Goal}}$$

In beiden Fällen bedeutet ein Wert von größer als 1, dass das Ziel nicht erfüllt wird und dass die Workload-Manager-Algorithmen aktiv werden müssen, um den Engpass zu beheben.

Ein Wert von 1 signalisiert Zielerfüllung. Ist ein Wert < 1, dann ist das Ziel übererfüllt.

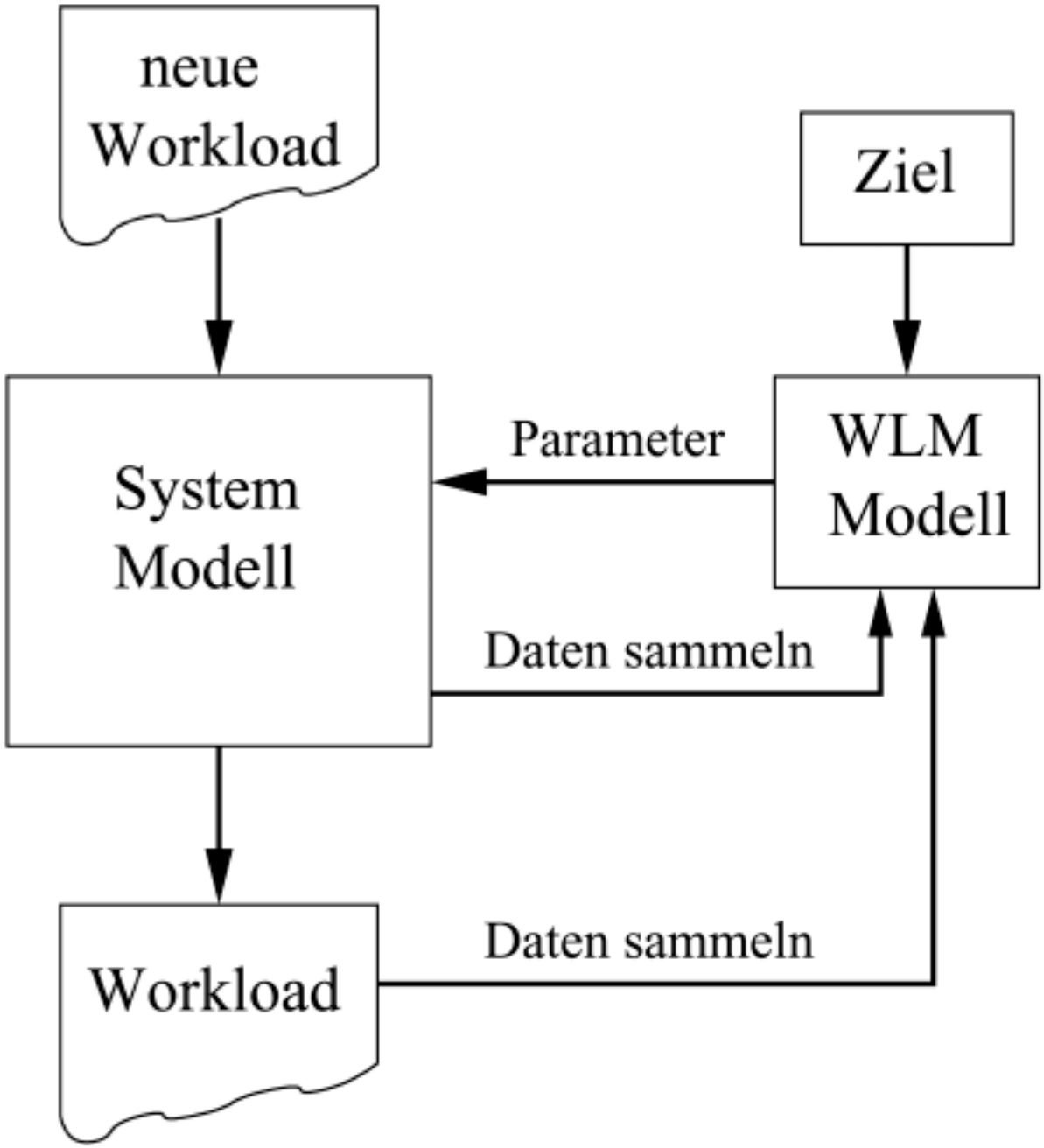


Classification Rules Beispiel

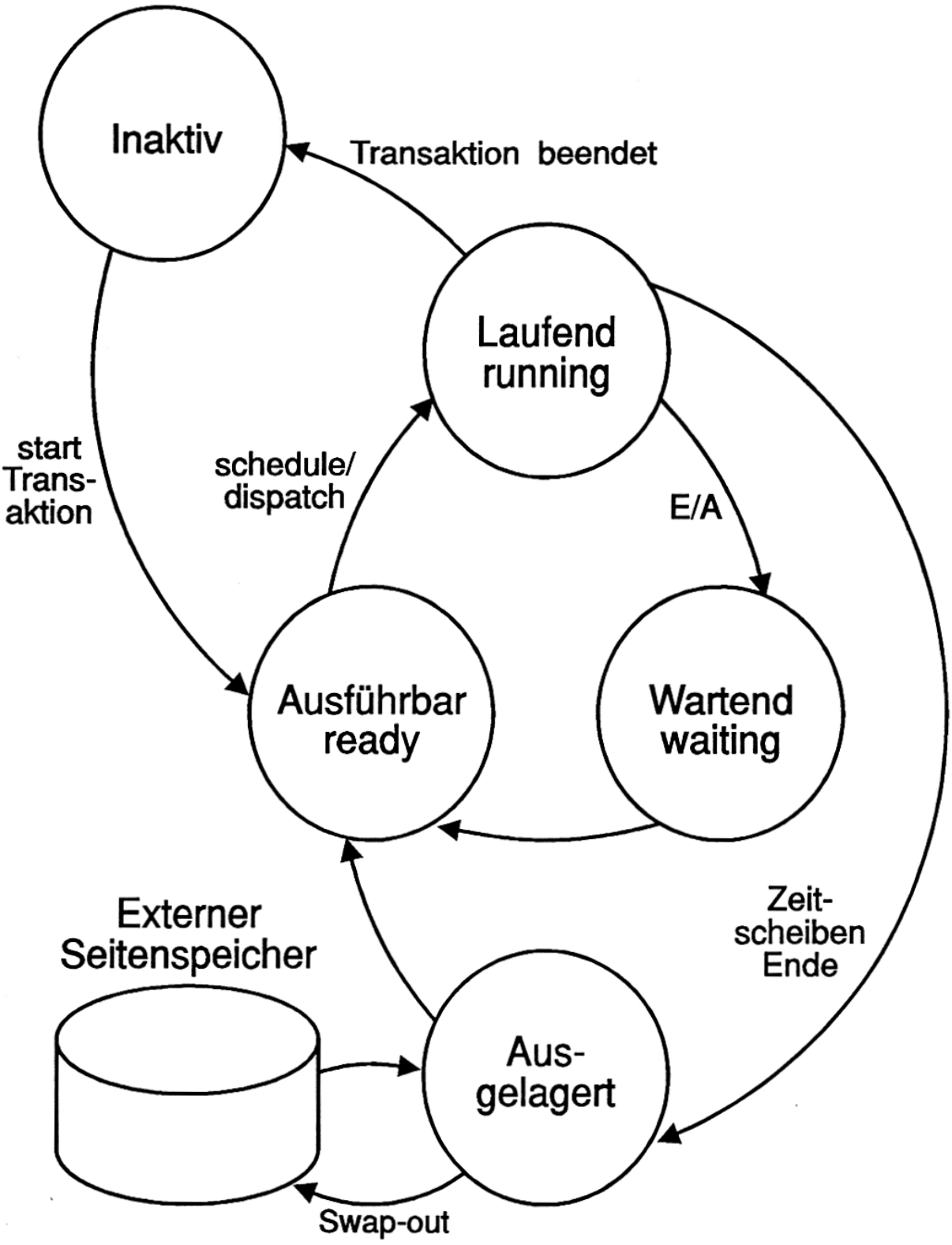
Unterschiedliche Ziele für Service Classes:
Wichtigkeit

Transaktionen, 90 % Antwortzeit < 0,3 s	**1**
Stapel 1, 90 % complete < 3 Stunden	**4**
Stapel 2, optimiert für E/A Operationen	**2**
Internet (WebSphere) , 90 % Antwortzeit < 5 s	**3**
Entwicklung (TSO), 90 % Antwortzeit < 2 s	**5**

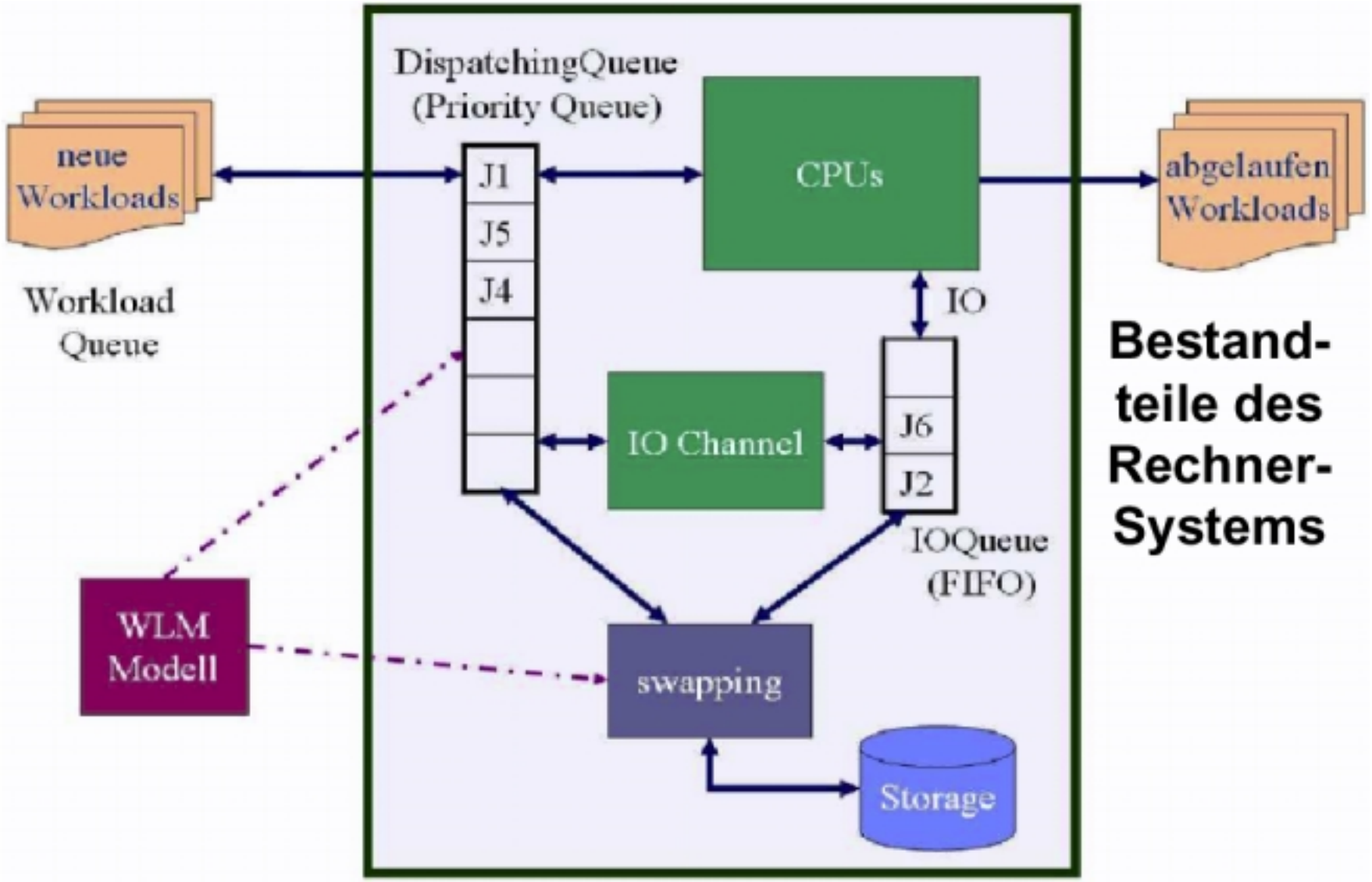

Wähle Receiver aus
Ermittle Receiver-Engpaß
Behebe Receiver-Engpaß
1. Ermittle Doner
2. Schätze Auswirkung der Änderung ab
3. Führe die Anpassung durch
Existieren weitere Engässe
End



Ein einfaches WLM Modell



Auslagern von Prozessen


DispatchingQueue
(Priority Queue)
neue
Workloads
Workload
Queue
J1
J5
J4
CPUs
abgelaufen
Workloads
IO
IO Channel
J6
J2
IOQueue
(FIFO)
WLM
Modell
swapping
Storage
Bestand-
teile des
Rechner-
Systems

Performance Beispiel

Es werden insgesamt 5 Benutzer-Service-Klassen modelliert:

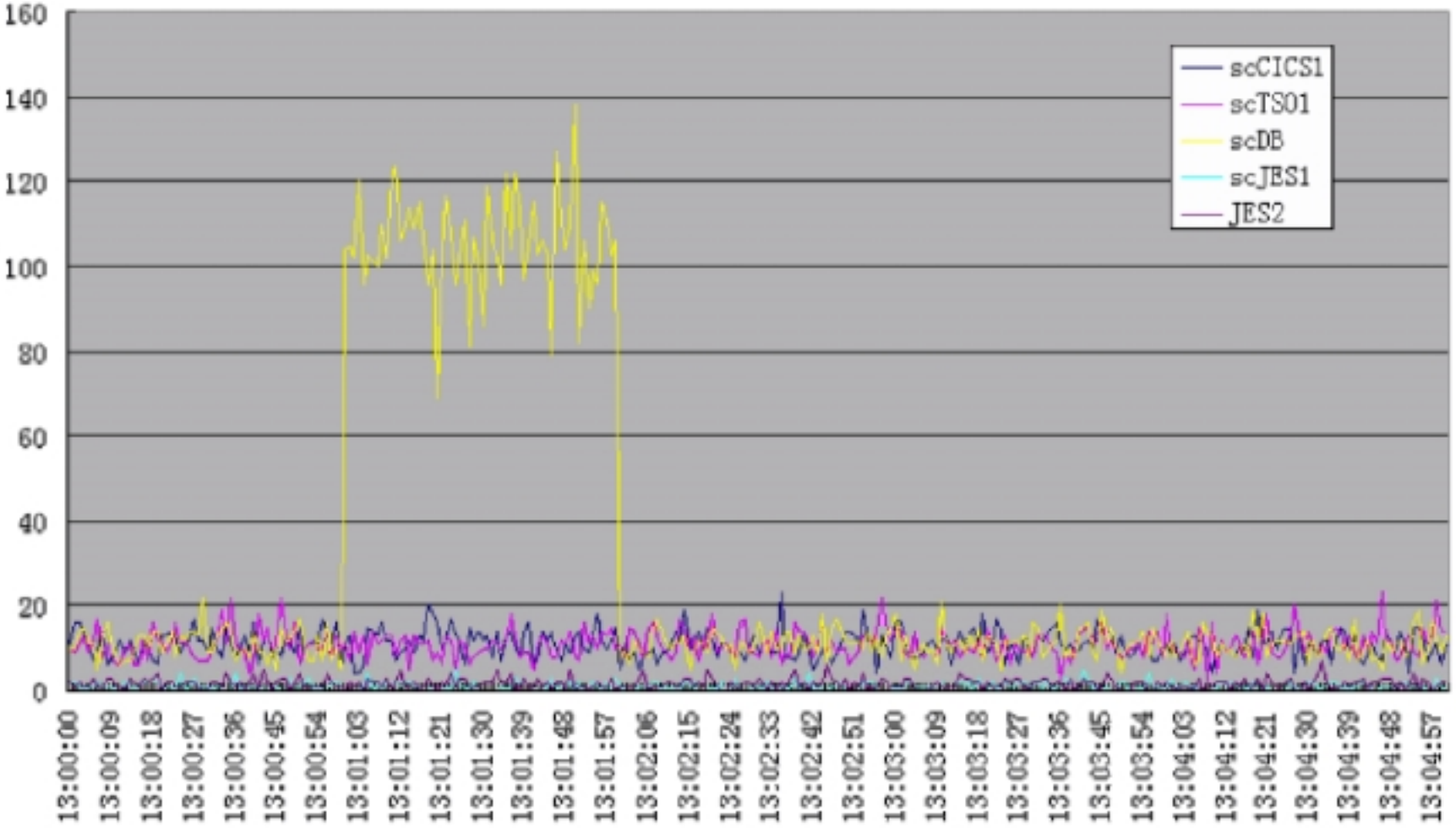
- **scCICS1 für CICS-Arbeitseinheiten.**
- **scTSO1 für TSO-Arbeitseinheiten.**
- **scDB21 für Datenbank-Anwendungen.**
- **scJES1 für Batch-Workloads mit höherer Priorität.**
- **scJES2 für Batch-Workloads mit niedrigen Priorität.**

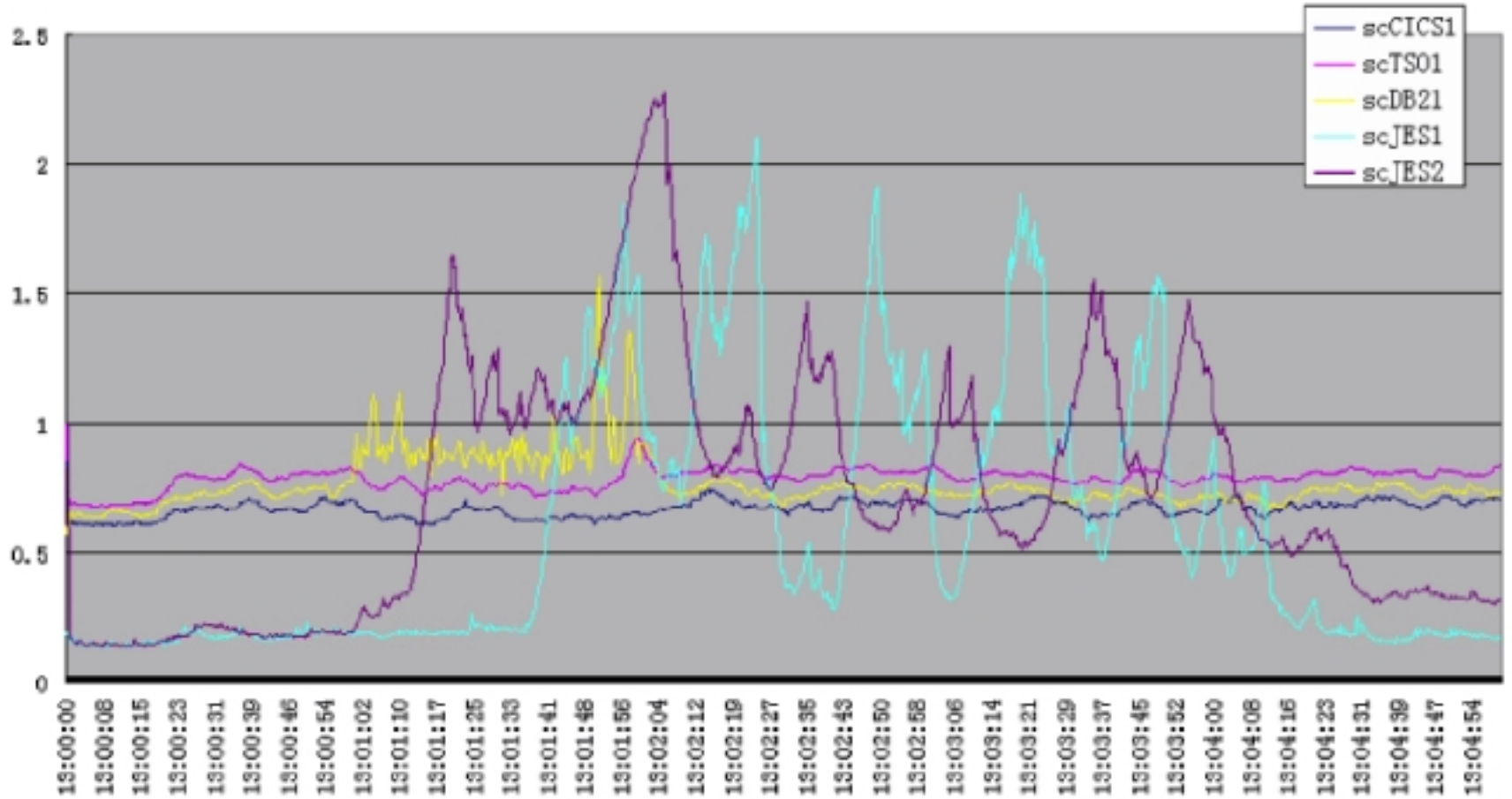
Service-Klassen Definition für das Experiment

Service-Klasse ID	sc Name	Importance	Goal
0	scCICS1	1 (hoch)	0,35 s Response
1	scTSO1	2	0,57 s Response
2	scDB21	3	0,3 s Response
3	scJES1	4	15 Velocity
4	scJES2	5 (niedrig)	15 Velocity

Die Tabelle zeigt, welche Benutzer-Service-Klassen und Ziele in dem Beispiel definiert wurden. Inder folgenden Graphik ist der Job-Verteilung von Service-Klassen.

Um 13:01 wird eine große Anzahl von Transaktion in der SK ‚scDB21' gestartet.


160
140
120
100
80
60
40
20
0
scCICS1
scTS01
scDB
scJES1
JES2
13:00:00
13:00:09
13:00:18
13:00:27
13:00:36
13:00:45
13:00:54
13:01:03
13:01:12
13:01:21
13:01:30
13:01:39
13:01:48
13:01:57
13:02:06
13:02:15
13:02:24
13:02:33
13:02:42
13:02:51
13:03:00
13:03:09
13:03:18
13:03:27
13:03:36
13:03:45
13:03:54
13:04:03
13:04:12
13:04:21
13:04:30
13:04:39
13:04:48
13:04:57


scCICS1
scTS01
scDB21
scJES1
scJES2
2.5
2
1.5
1
0.5
0
13:00:00
13:00:08
13:00:15
13:00:23
13:00:31
13:00:39
13:00:46
13:00:54
13:01:02
13:01:10
13:01:17
13:01:25
13:01:33
13:01:41
13:01:48
13:01:56
13:02:04
13:02:12
13:02:19
13:02:27
13:02:35
13:02:43
13:02:50
13:02:58
13:03:06
13:03:14
13:03:21
13:03:29
13:03:37
13:03:45
13:03:52
13:04:00
13:04:08
13:04:16
13:04:23
13:04:31
13:04:39
13:04:47
13:04:54

Reaktion des Work Load Managers

In der Grafik sind die PI (Performance Indizes) dargestellt. Da nur wenige Jobs bis zu 13:01 in das System kamen, die auch nur sehr geringe CPU-Anforderungen hatten, wurde aller Ziele von Service-Klassen erfüllt. Die Ziele für ‚scJES1' und ‚scJES2' sind nicht so eng gefasst, deshalb ist es auch möglich, die Ziele deutlich überzuerfüllen. Da seit 13:01:02 die Anforderungen durch die DB2-Jobs sehr hoch sind, werden die Ziele für ‚scDB21' trotzdem nicht erreicht. WLM beginnt jetzt zu steuern.

<u>Um 13:01:02</u> war der Receiver ‚scDB21' und der Donor ‚scJES2'. Wir sehen, dass etwa 10 Sekunde später (um 13:01:10) der PI für ‚scDB2' wieder unterhalb von 1 ist und der PI für ‚scJES2' deutlich oberhalb von 1 steigt, da die MPL-Wert der scJES2 reduziert wird.

<u>Um 13:01:10</u> war der Receiver die ‚scJES2' und der Donor die ‚scJES1'. Danach bis um 13:01:40 ist der PI für ‚scJES1' gestiegen und der PI für ‚scJES2' trotz keiner Zielerfüllung besser geworden.

<u>Um 13:01:41</u> gabt es zwei Service-Klasse ‚scJES1' und ‚scJES2' mit den PI von größer als 1. Der Receiver war damals die ‚scJES1', da die Importance der ‚scJES1' wichtiger ist als der ‚scJES2'. Da der Donor die ‚scDB2' mit den PI von kleiner als 1 war, wird die Ressource bei den Service-Klasse ‚scDB2' abgezogen. Es führt dazu, dass ‚scDB2' ihr Ziel wieder nicht erfüllt haben kann.

<u>Um 13:01:48</u> haben Service-Klassen ‚scDB21', ‚scJES1' und ‚scJES2' seine Ziele nicht erreicht. Die ‚scDB2' wird als der Receiver und die ‚scTSO1' wird als der Donor. Die PI der ‚scTSO1' ist gestiegen.

<u>Ab 13:02:00</u> ist die ankommende Jobs der ‚scDB2' niedrige wie vorher geworden, deshalb sind die PI von ‚scDB2' und ‚scJES1' besser geworden. Da die PI der ‚scJES2' immer größer als 1 ist, hat die WLM-Anpassung noch durchgeführt.

<u>Ab 13:02:04</u> sind weitere WLM-Anpassungen notwendig, da die PI der ‚scJES2' noch größer als 1 ist. Wir sehen, dass Schwankungen ab 13:02:04 durch WLM-Anpassungen hervorgerufen werden.

Schwankungen bei der Anpassung sind in Wirklichkeit kleiner als in der Graphik, da WLM in z/OS relative genau vorhersagen kann, wie viel Prioritäten bzw. MPLs erhöhen oder reduzieren soll, um Ziele zu erfüllen. Da die Vorhersagekomponente in dem Modell Modell nicht implementiert ist, muss WLM mehrmalige Anpassungen durchführen. Es führt zu Schwankungen (Oszillationen).

Um etwa 13:05 endet das Test.

Service Level Agreement (SLA)

Kontrakt zwischen einem Dienstanbieter und einem Dienstabnehmer.

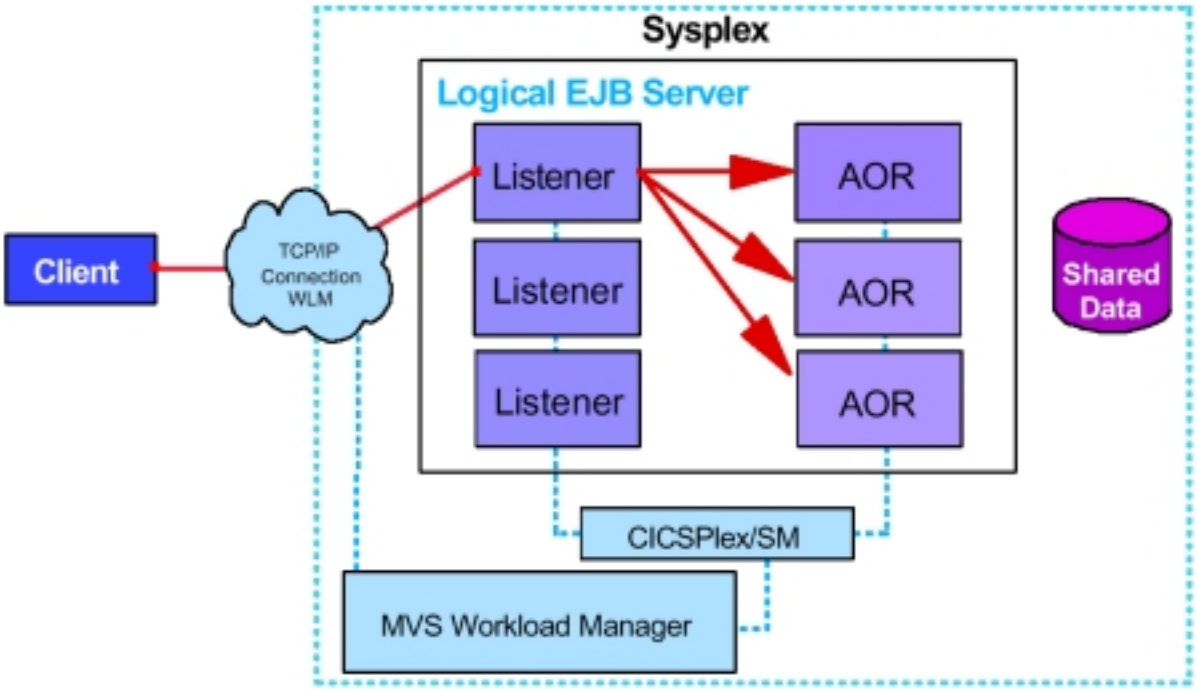
In der IT-Welt häufig ein Kontrakt zwischen Unternehmens-bereichen und der zentralen Unternehmens - IT (Rechenzentrum).

Mögliche Vereinbarungen:

- **Plattenspeicherplatz**
- **Rechenzyklen auf den Systemen**
- **Auslastung des Netzwerks**
- **Antwortzeiten**
- **Verfügbarkeitkriterien**

Die Aufgabe des Rechenzentrums besteht darin, die vorhandenen Systeme so abzustimmen, dass die Service Level Agreements eingehalten werden.

Das Einhalten von Service Level Agreements kann vereinfacht werden, wenn die Systeme Definitionen unterstützen, die in ihrer Form den Definitionen eines Service Level Agreements entsprechen oder nahe kommen. (Abbilden von SLAs auf Zieldefinitionen - Goals - des z/OS Workload Manager).

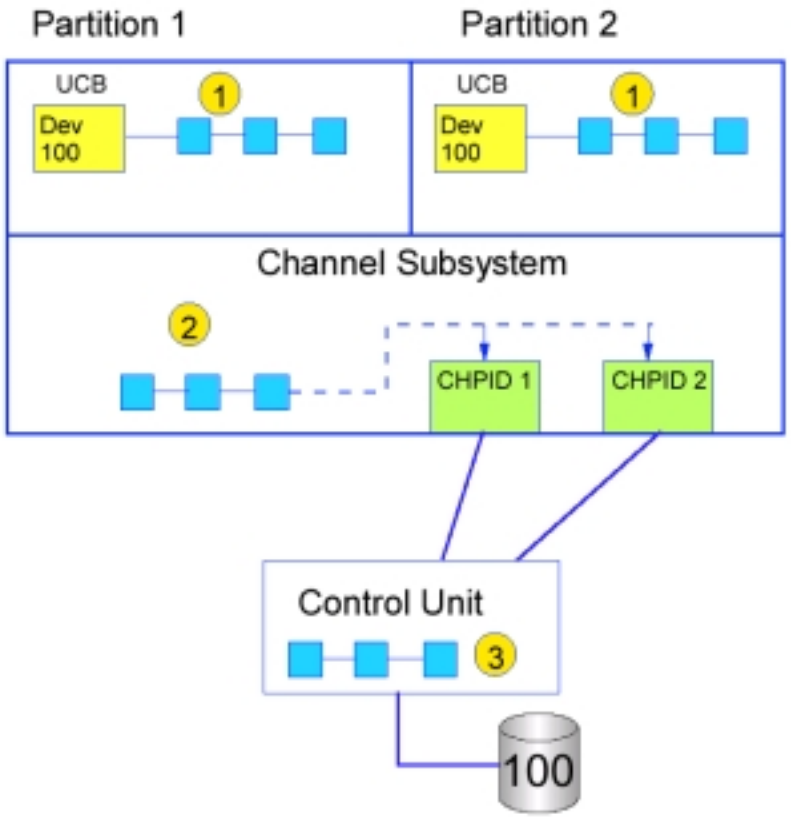

Sysplex
Logical EJB Server
Listener
Listener
Listener
AOR
AOR
AOR
Client
TCP/IP
Connection
WLM
Shared
Data
CICSPlex/SM
MVS Workload Manager

Service Level Agreement (SLA)

Vertrag zwischen einem Rechner und seinen Benutzern bezüglich der Dienstleistungsqualität, z.B. Antwortzeit. Beinhaltet häufig implizite oder explizite Vereinbarungen über Strafen, wenn ein SLA nicht erfüllt wird.

Lieblinge (Loved One´s)

Untermenge von Benutzern oder Transaktionsarten mit sehr engen SLA´s. Die Unternehmensleitung will Garantien bezüglich der Dienstleistungsqualität für bestimmte Anwendungen.



- WLM setzt I/O Priorität, die im Kanal Subsystem honoriert wird
  - WLM berechnet I/O Priorität als Teil des Policy Adjustement Algorithmus
  - I/O Priorität wird in eine Channel Subsystem Priorität konvertiert
  - Benutzer kann einen Bereich möglicher Prioritäten für eine LPAR oder LPAR Cluster definieren
    - Dies erlaubt auch statisch Partitionen gegeneinander zu priorisieren
- Ergänzt bestehende I/O Queueing Prioritäten
  1. an der UCB Queue
  2. im Channel Subsystem
  3. in einer Shark Control Unit

Workload Manager

Der **Workload Manager** (WLM) ist ein integraler Bestandteil des z/OS-Großrechner Betriebssystems von IBM. Es ist die Komponente, die für die Arbeit auf einem z/OS den Zugang zu den Betriebsmitteln steuert. Auf einem z/OS Großrechner ist dies notwendig, da viele unterschiedliche Anwendungen den Rechner gleichzeitig nutzen und eine den Kundenwünschen entsprechende Lastverteilung erfolgen muss. Der Workload Manager ist somit die zentrale Komponente, die eine verlässliche Ausführungszeit von Arbeitseinheiten unter z/OS garantiert und somit eine Grundvoraussetzung für einen verlässlichen Zugriff auf Datenbanksysteme ermöglicht.

WLM steuert die Betriebsmittelvergabe auf der Basis von Dienstklassen (im Fachbegriff: „Service Classes“). Arbeitseinheiten werden den Dienstklassen über einen Klassifizierungsmechanismus zugeordnet. Die Klassifizierung wird durch den Systemverwalter des z/OS Systems vorgenommen und kann anhand von Attributen, die für die Programmprodukte unter z/OS existieren vorgenommen werden, zum Beispiel Benutzernamen, Transaktionsnamen, Transaktionsklassen oder Programmnamen die in den Anwendungen verwendet werden. Als weiteres definiert der Verwalter eine Zielvorgabe für die Dienstklassen. Die Zielvorgabe kann die durchschnittliche Antwortzeit der Arbeitseinheiten, die in der Klasse laufen umfassen, einen prozentualen Anteil der Arbeitseinheiten, die in einer bestimmten Zeit enden sollen oder eine durchsatzorientierte Vorgabe darstellen. Welches Ziel für eine Dienstklasse vergeben werden kann hängt davon ab, wie viele Informationen der Workload Manager über die Anwendungen erhält. Neben der Zielvorgabe wird jeder Dienstklasse eine Wichtigkeit zugeordnet, die festlegt, welche Klassen bevorzugt bzw. benachteiligt werden sollen wenn die Betriebsmittel im System nicht mehr ausreichend zur Verfügung stehen.

WLM benutzt jetzt einen Regelmechanismus, um zur Laufzeit den Zugang zu den Betriebsmitteln zu steuern. Dazu werden kontinuierlich Daten aus dem z/OS System gesammelt. Dies sind Informationen über die Wartezustände der Arbeitseinheiten auf die Betriebsmittel, die Anzahl der laufenden Arbeitseinheiten und deren Abarbeitungszeiten. Die Informationen werden in Dienstklassen zusammengefasst entsprechend der Klassifizierung, die durch den Systemverwalter vorgenommen wurde. Dann wird auf der Basis dieser Informationen, die Zielerfüllung für jede Klasse berechnet und falls notwendig, der Zugang zu den Betriebsmitteln angepasst. Die Anpassung erfolgt immer in Abhängigkeit von der Wichtigkeit der Klassen und dem Grad, wie das Ziel verfehlt wird. Das heißt, die wichtigste Klasse, die am weitesten ihr vorgegebenes Ziel verfehlt hat wird als erste betrachtet und die Klassen mit der geringsten Wichtigkeit sind die potenziellen Kandidaten, um Betriebsmittel abzugeben. Dabei wird allerdings berücksichtigt ob ein potentieller Spender („Donor“) auch tatsächlich das benötigte Betriebsmittel verwendet. Dieser Regelmechanismus läuft alle 10 Sekunden im z/OS ab und in der Zwischenzeit werden die Daten für das nächste Berechnungsintervall gesammelt. Ein Berechnungsintervall endet wenn eine Anpassung zugunsten einer Dienstklasse durchgeführt werden kann.

WLM steuert den Zugang zu den Prozessoren und E/A Einheiten des Systems, den Zugang zum Speicher und die Bereitstellung von Adressräumen um Programme für

bestimmte Anwendungen abarbeiten zulassen. Der Zugang zu den Prozessoren wird zum Beispiel über „Dispatch Priorities“ geregelt. Dazu wird allen Arbeiteinheiten einer Dienstklasse dieselbe Priorität zugeordnet. Interessanterweise sollte festgehalten werden, dass die Vergabe dieser Priorität nicht in jedem Fall mit der Definition der Wichtigkeit der Dienstklasse übereinstimmen muss. Vielmehr orientiert sie sich an der aktuellen Auslastung des Systems, den Anforderungen der Klasse und ihrer Zielerfüllung. Dieses Verhalten des z/OS WLM nennt man auch zielorientiertes Workload Management und es ist ein wichtiges Unterscheidungskriterium zu anteilsorientiertem Workload Management, bei dem feste Zugänge zu den Betriebsmitteln vergeben werden. Letzteres findet sich häufig in Workload Management Komponenten von UNIX Systemen.

Der zweite essentielle Unterschied des z/OS WLM zu anderen Workload Management Komponenten ist die starke Verflechtung mit den Anwendungen und Programmprodukten, die unter einem z/OS Betriebssystem ablaufen. So ist durch die ständige Kommunikation zwischen dem WLM und diesen Anwendungen möglich, die Transaktionen der Anwendungen zu erkennen und im System durch den WLM zu steuern. Dies ist bis dato auf keinem anderen System möglich, in denen jedwede Steuerung auf Prozesse begrenzt ist.

Neben der Steuerung eines Systems bietet der z/OS WLM eine Reihe von Schnittstellen, die es Lastverteilungskomponenten erlauben Informationen aus dem System zu erhalten, um eine intelligente Verteilung von Arbeit auf eines oder mehrere z/OS Systeme vorzunehmen. Mehrere z/OS Systeme können in einem „Parallel Sysplex“ zusammengeschaltet werden und diese Kombination wird ebenfalls unterstützt, um nach außen ein einheitliches Bild abzugeben. z/OS WLM verfügt außerdem über eine Reihe von weiteren Funktionen, die die Lastverteilung auf einem physischen System zwischen mehreren logischen Systeme unterstützt und den Zugang zu großen Plattenfarmen in Abhängigkeit der daraufzugreifenden Arbeit steuert.

http://de.wikipedia.org/wiki/Workload_Manager

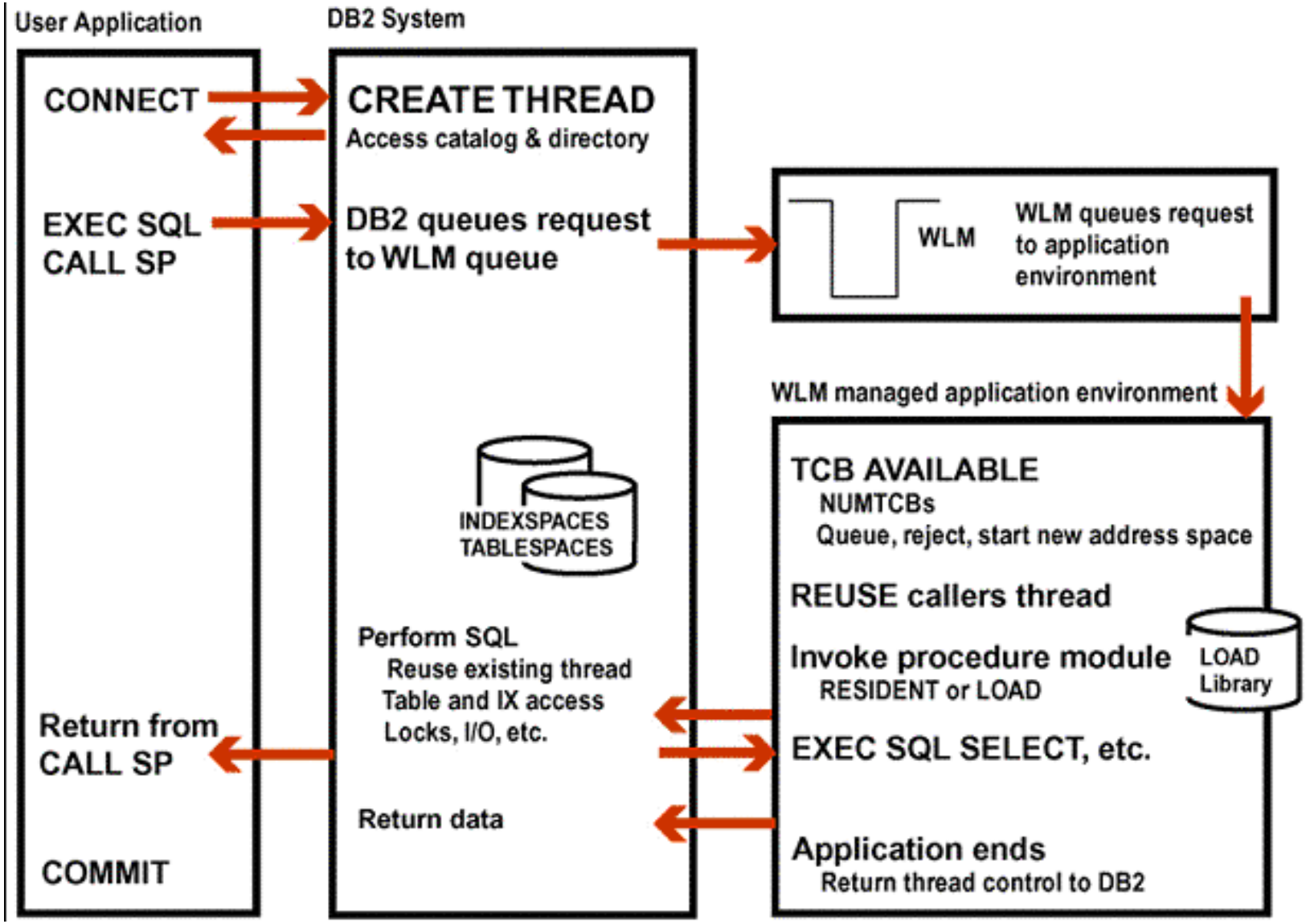
System Programmer's Guide to: Workload Manager
http://www.redbooks.ibm.com/abstracts/sg246472.html?Open

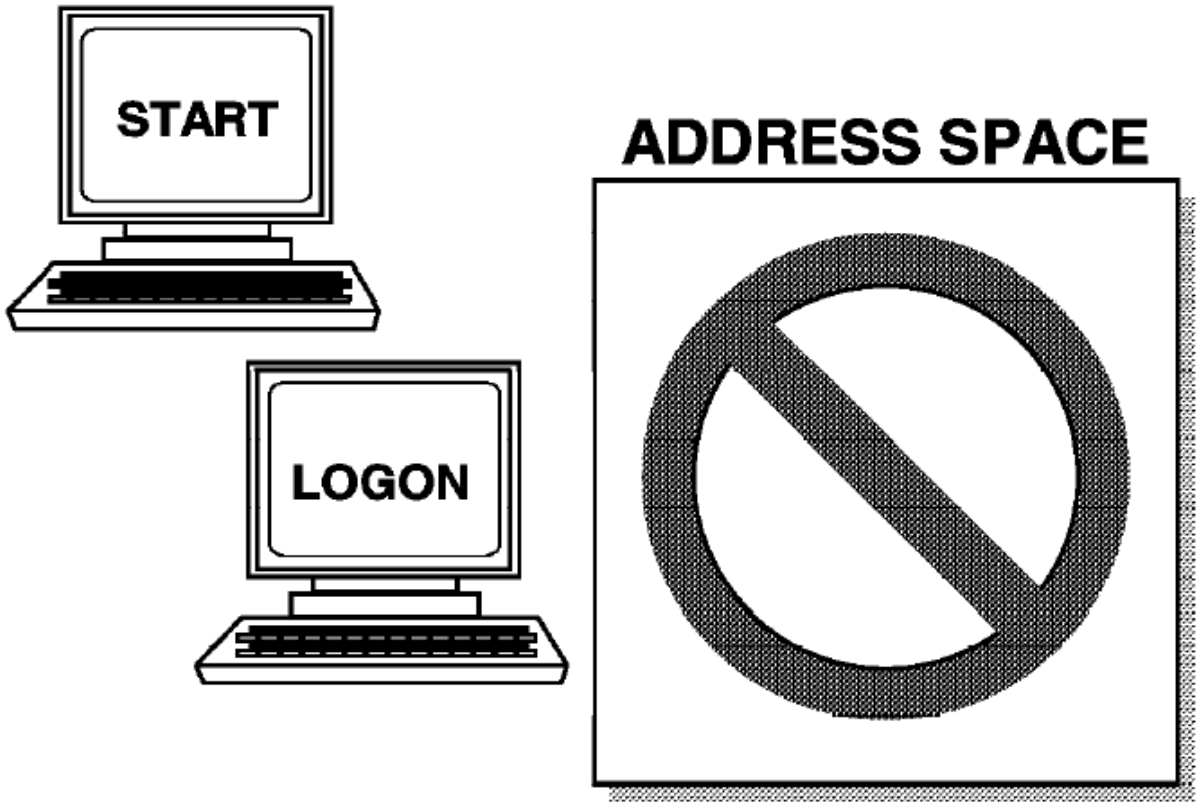
CICSPlex

Eine CICS Struktur kann aus einer einzigen Region auf einem einzelnen z/OS System bestehen. Eine CICS Struktur, die aus mehreren Regions besteht, wird als CICSPlex bezeichnet. Die unterschiedlichen Regionen eines CICSPlex können sich befinden:

- **Innerhalb des gleichen z/OS Systems**
- **Auf unterschiedlichen z/OS Systemen**
- **auf unterschiedlichen z/Series Betriebssystemen (z.B. z/OS, VSE**
- **auf unterschiedlichen Hardware Platformen (z.B. zSeries, PowerPC, Sun/Solaris, HP, PC/Windows, PC/Linux)**

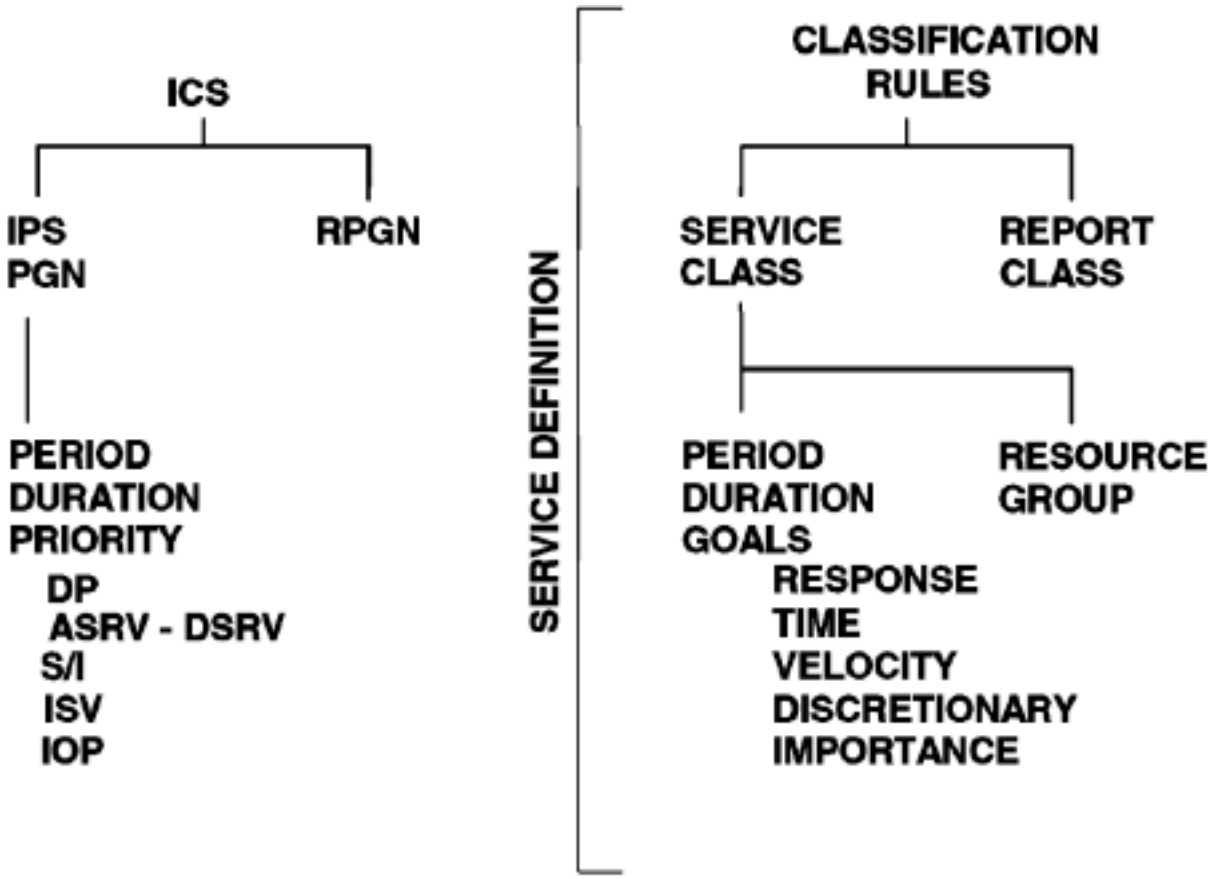
Eine Software Komponente, der CICSPlex System Manager (CICSPlex SM) ermöglicht die zentrale Steuerung und Verwaltung einer Gruppe von CICSPlex Systemen.


User Application
DB2 System
CONNECT
CREATE THREAD
Access catalog & directory
EXEC SQL
CALL SP
DB2 queues request
to WLM queue
WLM
WLM queues request
to application
environment
WLM managed application environment
INDEXSPACES
TABLESPACES
TCB AVAILABLE
NUMTCBs
Queue, reject, start new address space
REUSE callers thread
Invoke procedure module
RESIDENT or LOAD
LOAD
Library
Perform SQL
Reuse existing thread
Table and IX access
Locks, I/O, etc.
EXEC SQL SELECT, etc.
Return from
CALL SP
Return data
Application ends
Return thread control to DB2
COMMIT

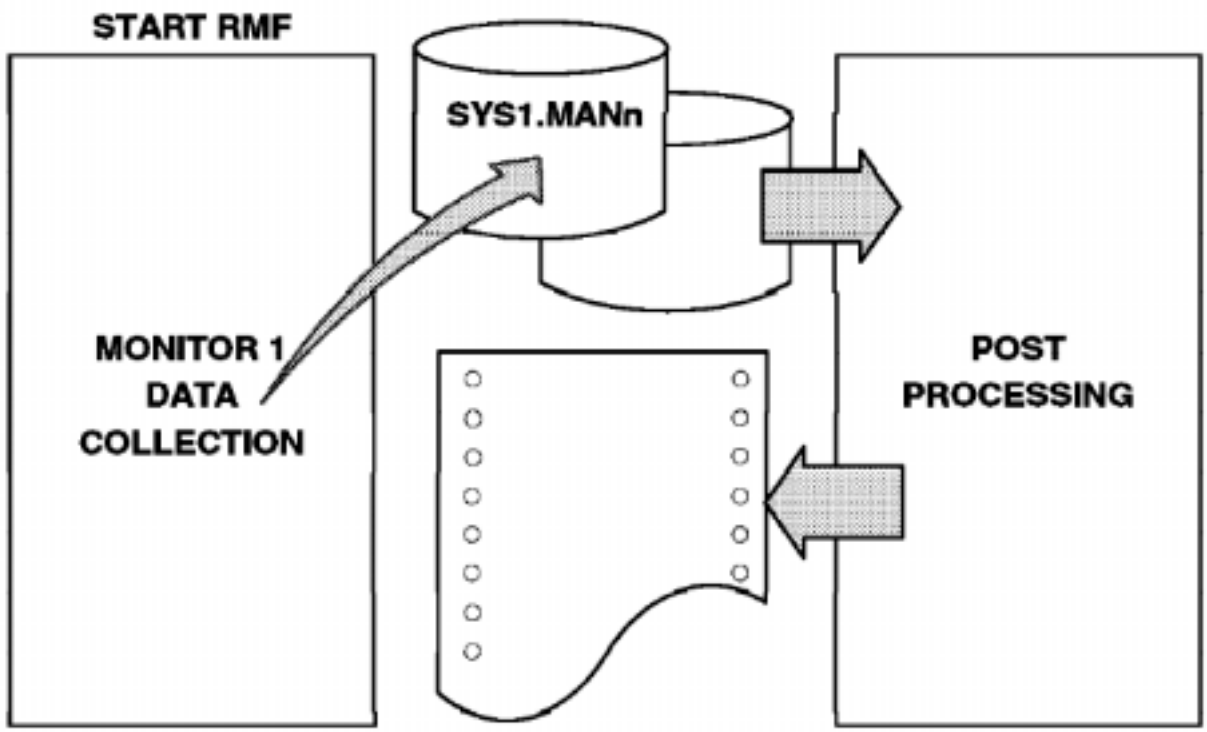


SRM will only allow address spaces to be created if there are enough available resources to handle them.

A shortage of frames, slots, or pages (in the System Queue Area) will result in SRM preventing the creation of address spaces.

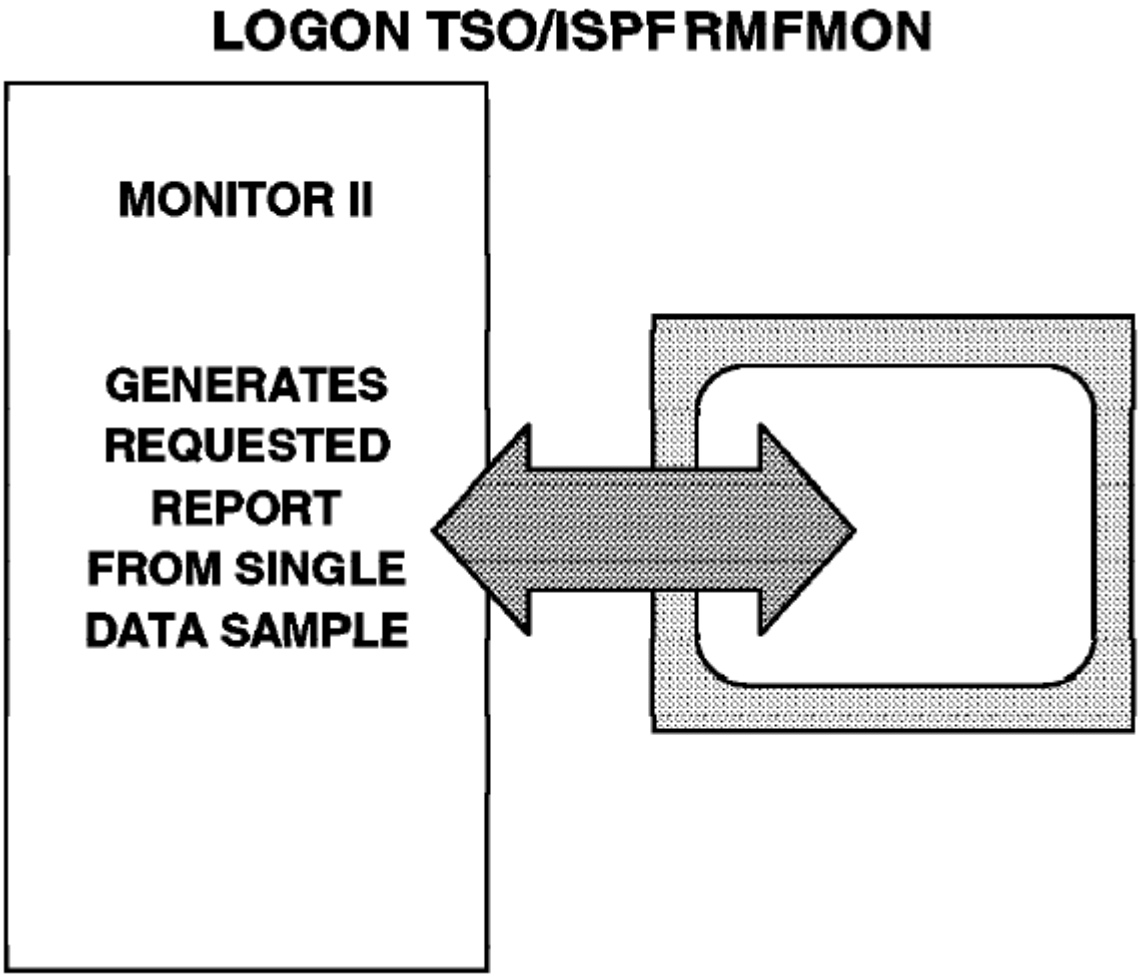


With old methods of communicating to SRM, we used to code values for various parameters which set bounds on various resources. Some examples are dispatching priority, MPL (ASRV and DSRV), and storage isolation. Controlling these, in turn, affected how the system processed the work: workflow. With the new system, once work is assigned a service class, most of the parmameters are related to goals. The resource group is an extra set of controls used to override normal resource usage.



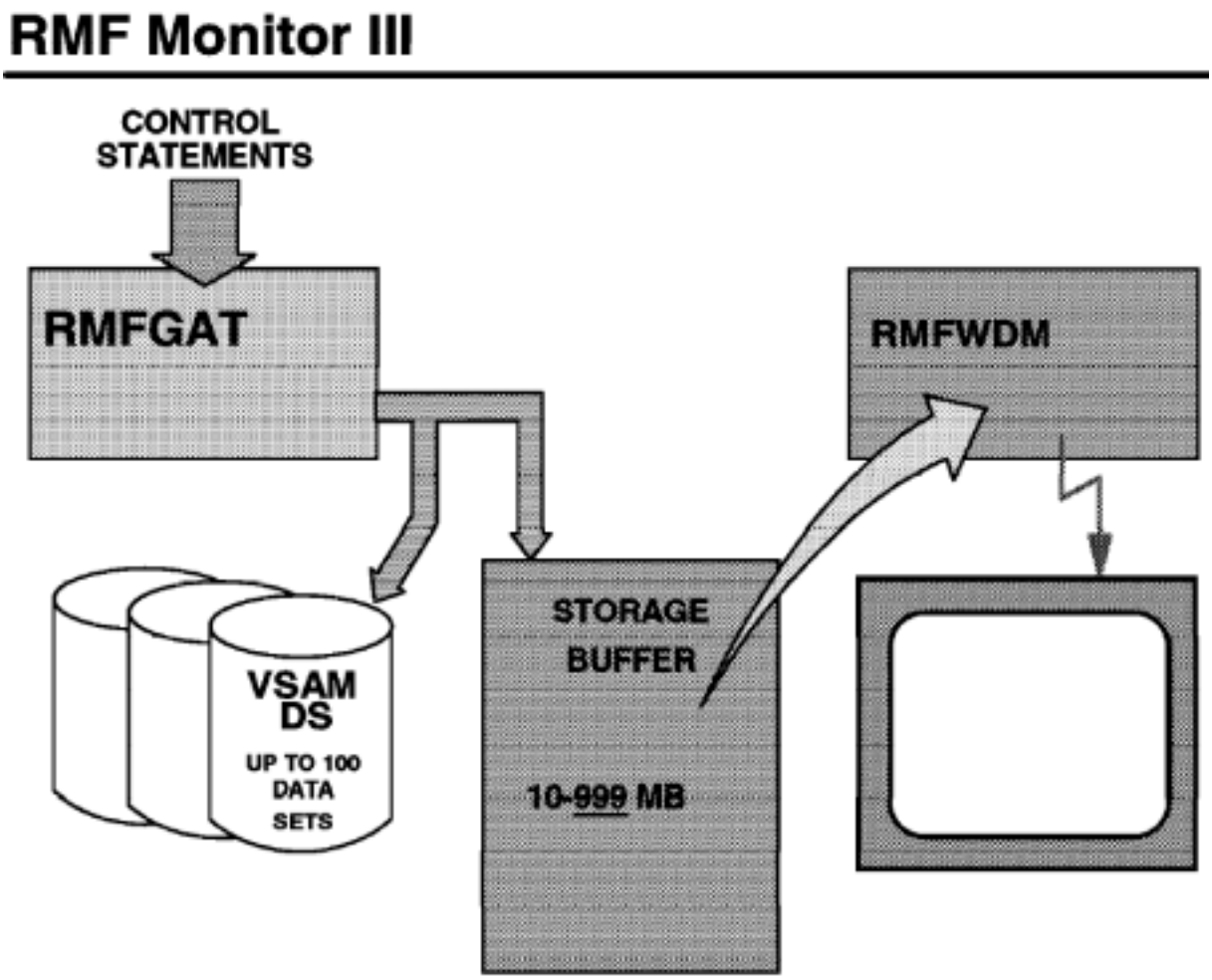
RMF Monitor 1

RMF is a started task that runs in its own address space. An RMF Monitor 1 session collects system data (for example, CP utilization, DASD activity, processor storage utilization, and workload activity) over a period of time and provides printed reports. The user specifies the sample cycle, the interval, and the type of report (for example, CP, Device, Path). RMF post processing summarizes the collected data and, with the use of a Report Writer, produces printed reports.



RMF Monitor 2

RMF Monitor 2 is used to take a snapshot of the system for a short period of time. Usually started from a TSO terminal with the RMFMON command.



RMF Monitor III

RMF Monitor III is used to detect bottlenecks within the system and to measure how fast transactions are moving through the system. A Monitor III session can be started from a TSO terminal with the RMFWDM command. The RMF gatherer program (RMFGAT) must have been executing at least one hour for the data to be of any value. The RMFGAT module gathers data from internal control blocks and places it in an internal buffer. If a VSAM data set is made available, data may also be placed there for permanent storage. The buffer is a wraparound in that RMFGAT continues to gather data. When the buffer is full, it starts over at the beginning. The RMFWDM module is invoked when a TSO user issues the RMFWDM command at a TSO terminal. This module scans the buffer for the requested data and presents it to the terminal. In a sysplex environment, data may be displayed as a composite of all systems in the sysplex.