Einführung in z/OS
Enterprise Computing

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

WS 2008/2009

Teil 10

Transaktionsverarbeitung mit CICS

Literatur

J. Gray, A. Reuter:

„Transaction Processing“.
Morgan Kaufmann, 1993.

J. Horswill:

„Designing and Programming CICS Applications“. O’Reilly, 2000

Mark Little, Jon Maron, Greg Pavlik:

Java Transaction Processing :
Design and Implementation
Prentice Hall 2004, ISBN 0-13-035290-X

Transaktionen

Transaktionen sind Stapelverarbeitungs- oder 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(en) 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. Dieser Vorgang wird als Rollback, Backout oder Abort bezeichnet die Begriffe sind Synonyme).

Wird eine Transaktion abgebrochen, werden keine Daten abgeändert

ACID Eigenschaften

Atomizität (Atomicity)

Konsistenzerhaltung (Consistency)

Isolation

Dauerhaftigkeit (Durability)

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 die transaktionsgeschützten Daten des Systems von einem konsistenten Zustand in einen anderen konsistenten Zustand.

Diese Eigenschaft garantiert, dass die Daten der Datenbank nach Abschluss einer Transaktion schemakonsistent sind, d. h. alle im Datenbankschema spezifizierten Integritätsbedingungen erfüllen

Daten sind konsistent, wenn sie durch eine Transaktion erzeugt wurden.

Isolation

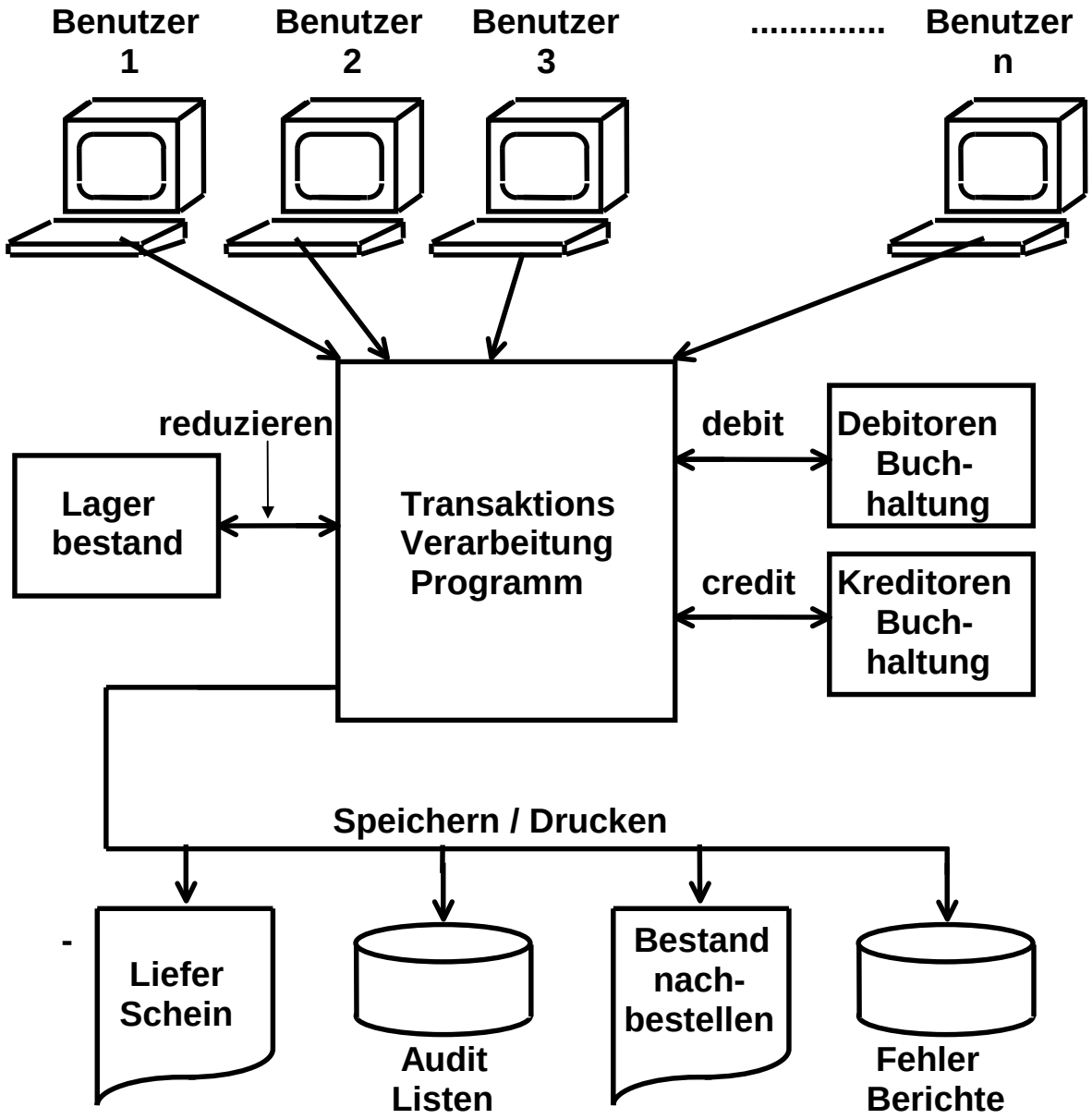
Die Auswirkungen einer Transaktion werden erst nach ihrer erfolgreichen Beendigung für andere Transaktionen sichtbar

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

Mehr als zwei Drittel aller in der Wirtschaft oder im öffentlichen Dienst durchgeführten Datenverarbeitungsvorgänge werden als Transaktionen ausgeführt und erfüllen die ACID Bedingungen.

Typische Struktureines transaktionalen Programms

- •
- •
- •

start transaction

- •
- •
- •

if ok then commit else rollback

- •
- •
- •

commit bewirkt, dass die Datenbank in einen neuen konsistenten Zustand überführt wird.

Statt rollback werden auch die Schlüsselworte abort oder backout verwendet.

Wie schreibt man
eine transaktionale Anwendung ?

Drei Möglichkeiten

1. Man macht alles selber
- **Sehr effektiv**
- **Sehr aufwendig**
- **Vermutlich 95 % des Codes beschäftigen sich mit der Einhaltung der ACID Bedingungen**

2. Stored Procedures
- **als Teil des Datenbankprozesses**
- **Für einfache Anwendungen**

3. Transaktionsmonitor

Frage: Wie verarbeitet der Server gleichzeitig Hunderte oder Tausende von Transaktionen ?

Transaktionsmonitor

- **Stored Procedures nur für einfache Aufgaben**

- **Bei allen leistungskritischen Anwendungen werden ausschliesslich Transaktionsmonitore eingesetzt.**

- **Ein Transaktionsmonitor ist eine Laufzeitumgebung für Transaktions-Anwendungen. Anwendungen laufen in einem eigenen Adressenraum gesteuert von einem Transaktionsmonitorprozess**

Ein Transaktionsmonitor ist eine Softwarekomponente, welche den atomaren Charakter vieler gleichzeitig ablaufender Transaktionen sicherstellt. Der TP Monitor stellt die Kernfunktionen für ein *Transaktions-verarbeitungssysten* bereit. Zu diesen Kernfunktionen gehören:

- **Message Queuing**
- **Lock Verwaltung**
- **Log Verwaltung**
- **2-Phase Commit Synchronisation**
- **Rollback Funktion**
- **Laststeuerung (Load Balancing)**

Beispiel für Transaktionsmonitore

Transaktionsmonitore sind Softwarepakete (auch als Middleware bezeichnet), die von verschiedenen Herstellern vertrieben werden. die wichtigsten sind:

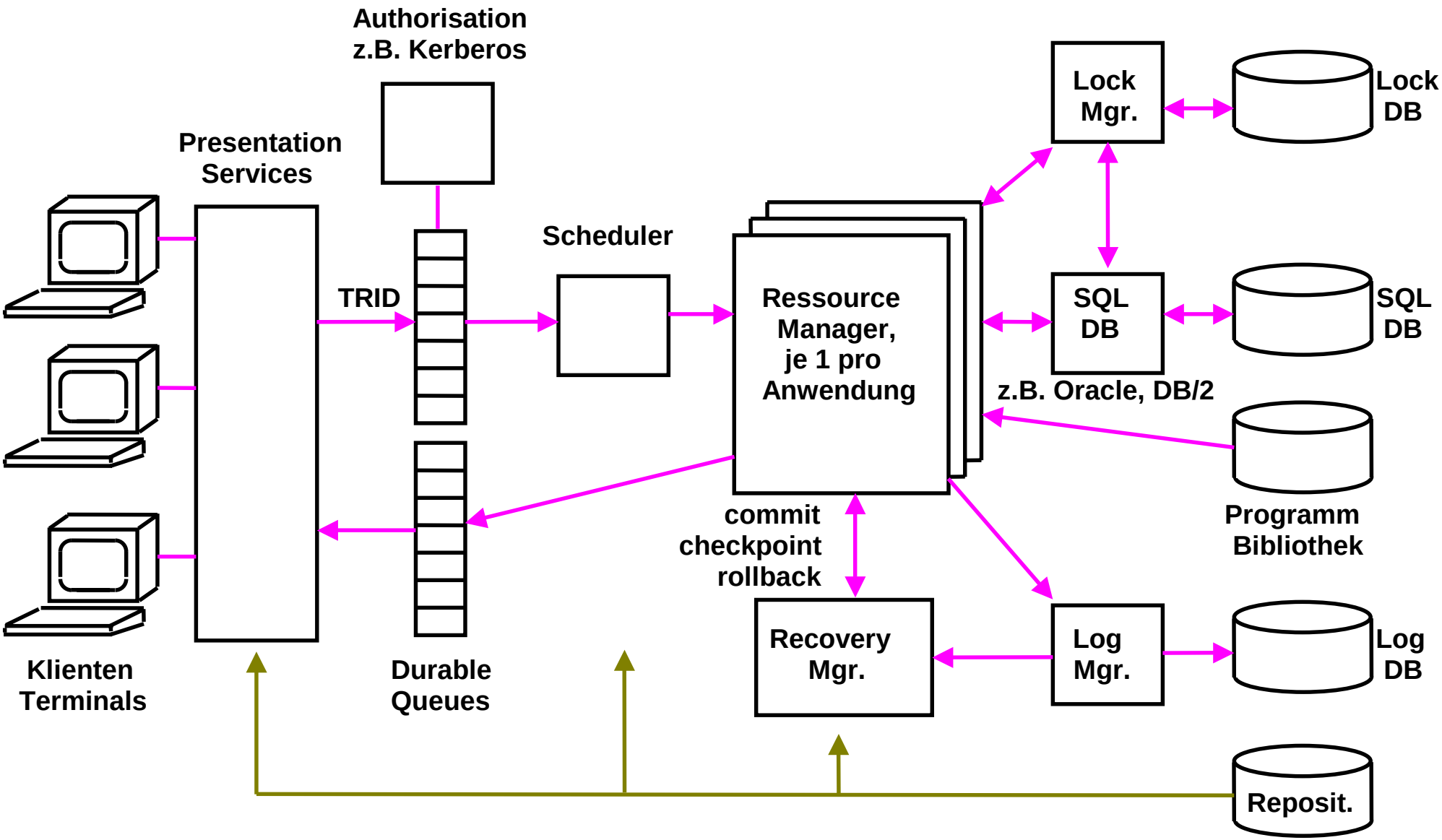
- **Tuxedo der Firma Bea (heute ein Tochterunternehmen von Oracle)**
- **CICS der Firma IBM**
- **SAP R/3 der Firma SAP**
- **Transaction Server (MTS) der Firma Microsoft**

Daneben von Bedeutung sind :

- **IMS-DC der Firma IBM**
- **TPF (Transaction Processing Facility) der Firma IBM**
- **UTM der Firma Siemens**
- **NonStop / Guardian / Pathway der Firma Hewlett Packard**

Für die objekt-orientierte Programmierung sind von wachsender Bedeutung:

- **Corba OTS (Object Transaction Service)**
- **EJB JTS (Enterprise Java Bean Transaction Service)**



Struktur eines TP Monitors

Klienten (Arbeitsplatzrechner) werden häufig als „Terminals“ bezeichnet.

Komponenten eines TP Monitors

Endbenutzer kommunizieren mit dem TP Monitor mit Hilfe von Nachrichten.

Presentation Services bilden die Datenausgabe auf die GUI des Benutzers ab.

Eingabe-Nachrichten werden mit einer trid (Transaktions ID) versehen und in einer Warteschlange gepuffert. Aus Zuverlässigkeitsgründen muß diese einen Systemabsturz überleben. Eine ähnliche Warteschlange existiert für die Ausgabe von Nachrichten.

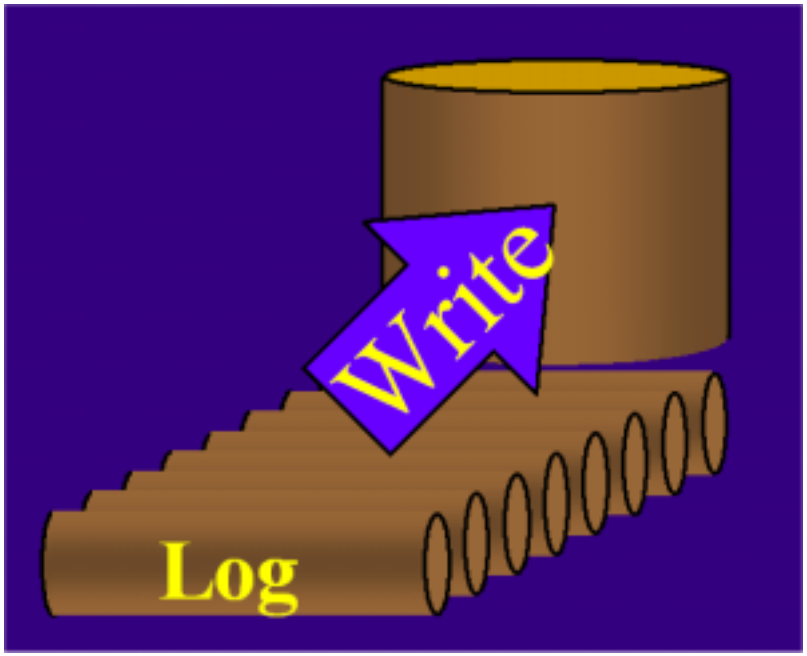
Der Scheduler verteilt eingehende Bearbeitungsanforderungen auf die einzelnen Server Prozesse.

Ein TP Monitor bezeichnet seine Server Prozesse als Ressource Manager. Es existiert ein Ressource Manager pro (aktive) Anwendung. Ressource Manager sind multithreaded; ein spezifischer Ressource Manager für eine bestimmte Anwendung ist in der Lage, mehrere Transaktionen gleichzeitig zu verarbeiten.

Der Lock Manager blockiert einen Teil einer Datenbanktabelle. In Zusammenarbeit mit dem Datenbanksystem stellt er die „Isolation“ der ACID Eigenschaft sicher.

Der LOG Manager hält alle Änderungen gegen die Datenbank fest. Mit Hilfe der Log Datenbank kann der Recovery Manager im Fehlerfall den ursprünglichen Zustand der Datenbank wiederherstellen (Atomizität der ACID Eigenschaft).

In dem Repository verwaltet der TP Monitor Benutzerdaten und -rechte, Screens, Datenbanktabellen, Anwendungen sowie zurückliegende Versionen von Anwendungen und Prozeduren.



Log plus alter Zustand ergibt neuen Zustand.

Beim „Commit“ einer Transaktion wird das Log persistent gespeichert (z.B. auf einer gespiegelten Platte).

Log wird zur Wiederherstellung einer Transaktion im Falle eines Fehlers benutzt:

- **System failure: lost in-memory updates**
- **Media failure (lost disk)**

Bewirkt die Dauerhaftigkeit (Durability) einer Transaktion.

Log File Konzept

Das Log ist eine historische Aufzeichnung aller Änderungen des Zustands (state) des Systems. Das Log ist eine sequentielle Datei. Das vollständige Log enthält die historische Entwicklung aller Datenbankänderungen.

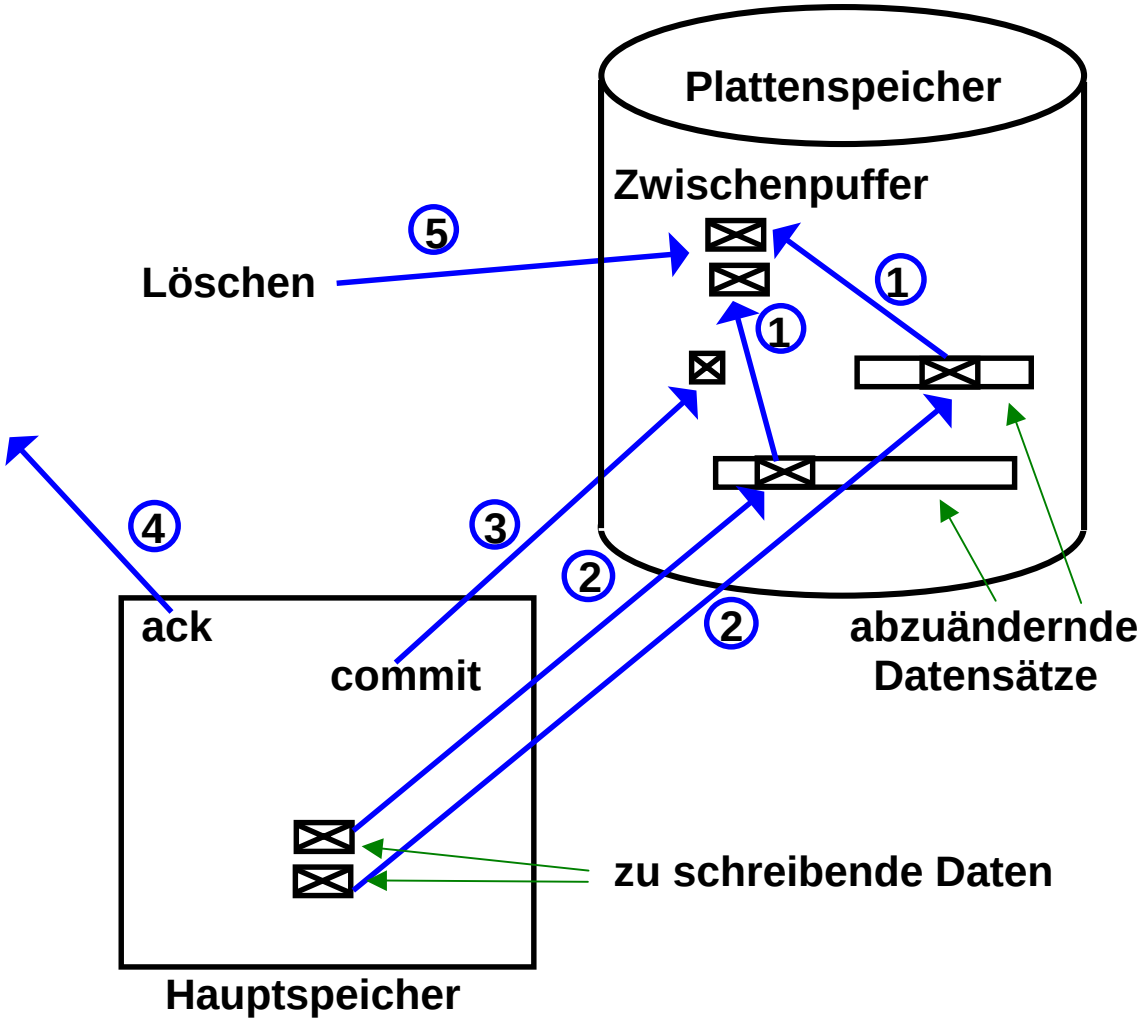
Backward Recovery

Die Recovery Manager Komponente des Transaktionsmonitors stellt sicher, daß im Fehlerfall der teilweise Ablauf einer Transaktion rückgängig gemacht wird, und daß alle abgeänderten Felder einer Datenbank wieder in ihren ursprünglichen Zustand zurückbersetzt werden.

Andere Bezeichnungen für Backward Recovery sind:

- **backout**
- **rollback**
- **abort**

Der Recovery Manager benutzt hierfür Daten, die entweder in temporären Zwischenpuffern auf der Festplatte oder in der Log file (oder beiden) festgehalten werden.



Backward Recovery

Andere Bezeichnungen sind : backout, roll back oder abort.

Zur Ermöglichung einer eventuellen Recovery finden bei jedem Write Vorgang die folgenden Schritte statt:

1. **abzuändernde Information in Puffer zwischenspeichern**
2. **Datensätze überschreiben**
3. **Commit Status festhalten. Damit ist es geschehen**
4. **erfolgreiches Commit dem Benutzer mitteilen**
5. **Zwischenpuffer löschen**

Backward Recovery

Recoverable Resources

Some data is designated a *recoverable resource* by a Transaction Monitor:

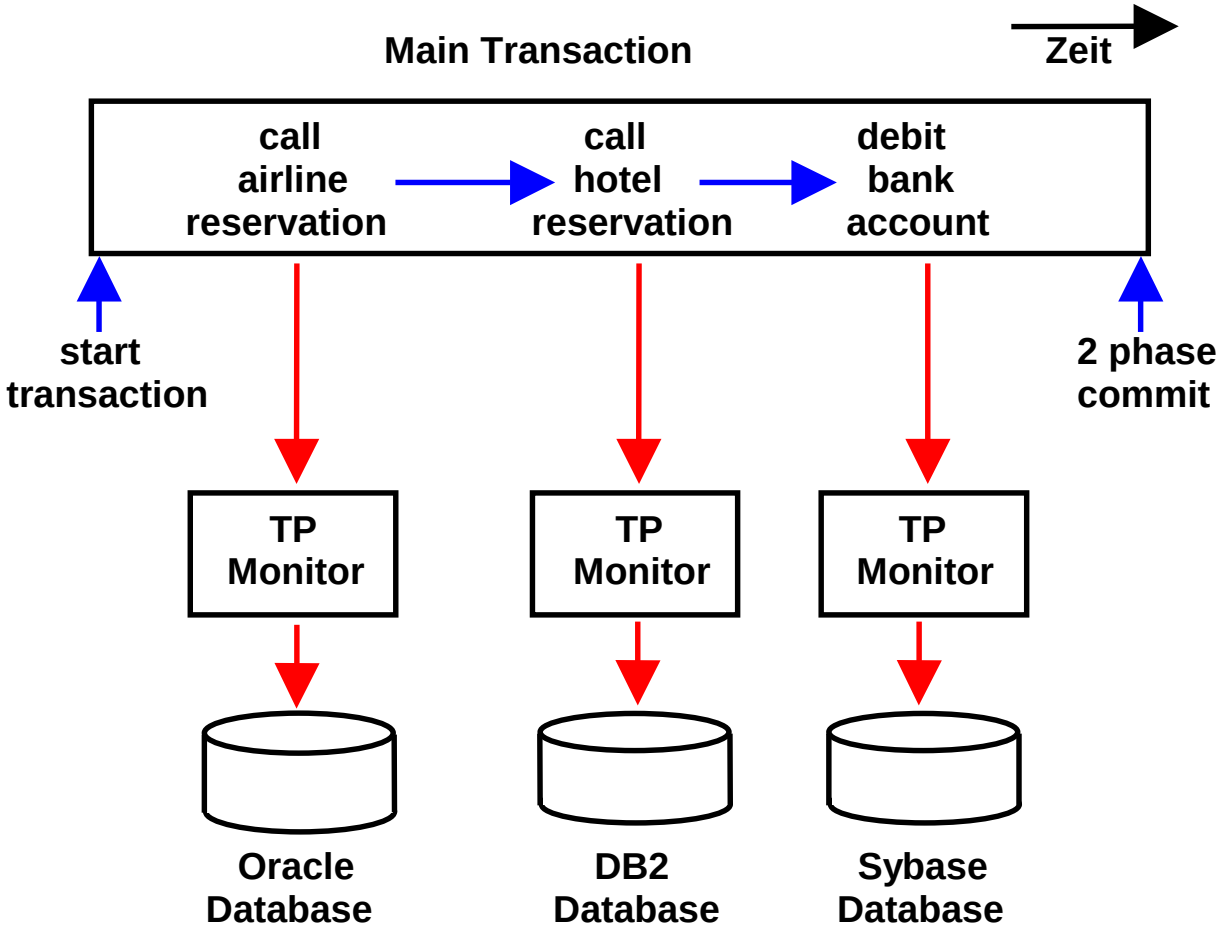
Recoverable resources can include:

- **Any designated file**
- **databases**
- **Certain transient data and temporary storage queues**

This data has recovery information logged or recorded by the transaction monitor in the system's dynamic Log. If a transaction involving a recoverable resource is successful, then the recovery information is deleted from the log.

However, if the task fails because of an error (whether in the application program, data access or transmission, or because of cancellation), the transaction monitor performs a dynamic transaction backout (DTB).

This reverses the pre-abend or pre-task failure updates and works backward from the last change before the failure occurred. DTB keeps corrupted data from being used by other tasks.



Distributed Flat Transaction

In dem gezeigten Beispiel beinhaltet die Transaktion für die Urlaubsreservierung (Main Transaction) Aufrufe für drei unterschiedliche TP Monitore und drei verschiedene Datenbanken.

Dieser Fall wäre mit einer nested Transaktion lösbar. Nested Transaktionen werden in der Praxis aber nicht eingesetzt.

An dessen Stelle tritt das 2-Pase Commit Protokoll, welches Bestandteil vieler TP Monitore ist, besonders auch der unter z/OS verfügbaren TP Monitore.

Die z/OS “Resource Recovery Services” beinhalten ebenfalls ein 2-Pase Commit Protokoll.

Two-phase Commit Protocol

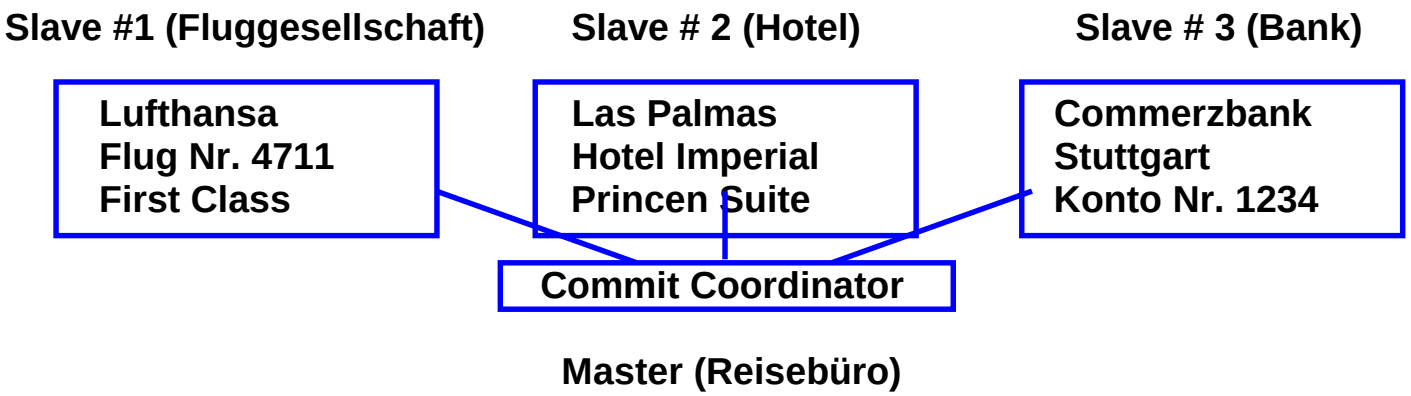
Das 2-Phase Commit Protokoll steuert die gleichzeitige Änderung mehrerer Datenbanken, z.B. bei der Urlaubsreise-Buchung die Änderung der Datenbanken der Fluggesellschaft, des Hotels und des Bankkontos von dem die Bezahlung der Reise abgebucht wird. Das Update der drei datenbanken erfolgt durch unabhängige TP Monitore.

Ein Problem tritt auf, wenn eines der Updates nicht erfolgen kann, z.B. weil das Hotel ausgebucht ist. Daher sind atomare Transaktionen erforderlich

Die Konsistenz wird erreicht durch einen „Master“ (Commit-Koordinator), der die Arbeit von „Slaves“ überwacht. Der TP Monitor des Reisebüros übernimmt die rolle des Masters, die TP Monitore der Fluggesellschaft, des Hotels und der Bank sind die Slaves.

Der Master sendet Nachrichten an die drei Slaves. Jeder Slave markiert die Buchung als tentativ und antwortet mit einer Bestätigung (Phase 1).

Wenn alle Slaves ok sagen sendet der Master eine Commit Nachricht, sonst eine Rollback Nachricht (Phase 2).



Master	Slave
Begin atomic action Send Request 1 ..... Send Request n Send „ Prepare to commit“	
	if action can be performed then begin Lock data Store initial state on disk Store requests on disk Send „OK“ end else Send „Failure“
Ende Phase 1	
if all slaves said „OK“ then send „Commit“ else send „Rollback“ Wait for acknowledgements	
	if master said commt then begin Do work Unlock data end Send „Acknowledgement“
Ende Phase 2	

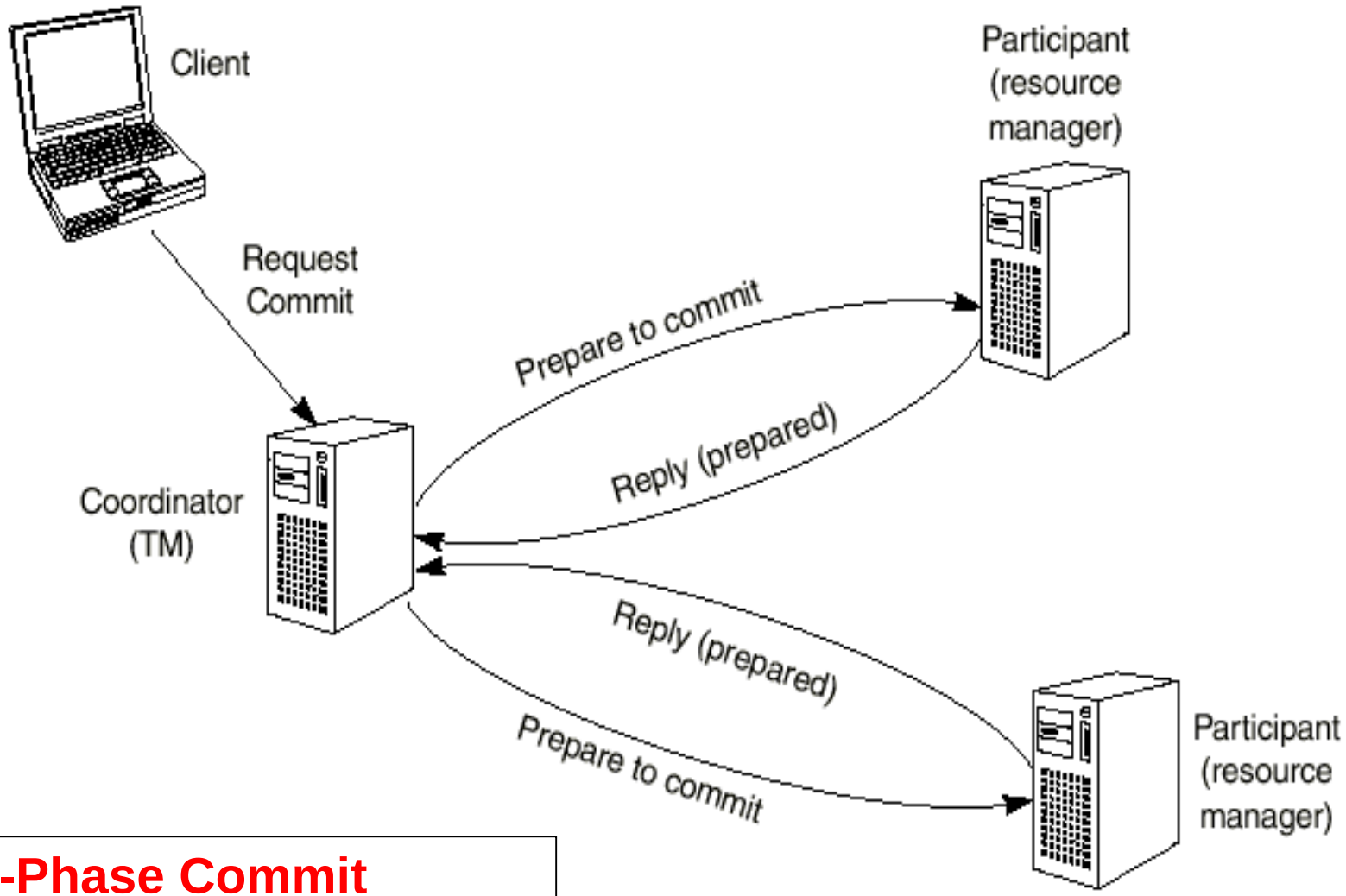
2-Phase Commit Protokoll

Der Master fragt bei allen beteiligten Slaves an, ob die gewünschte Aktion machbar ist. Jeder Slave markiert die Anforderung als tentativ und schickt eine Bestätigung an den Master Phase 1).

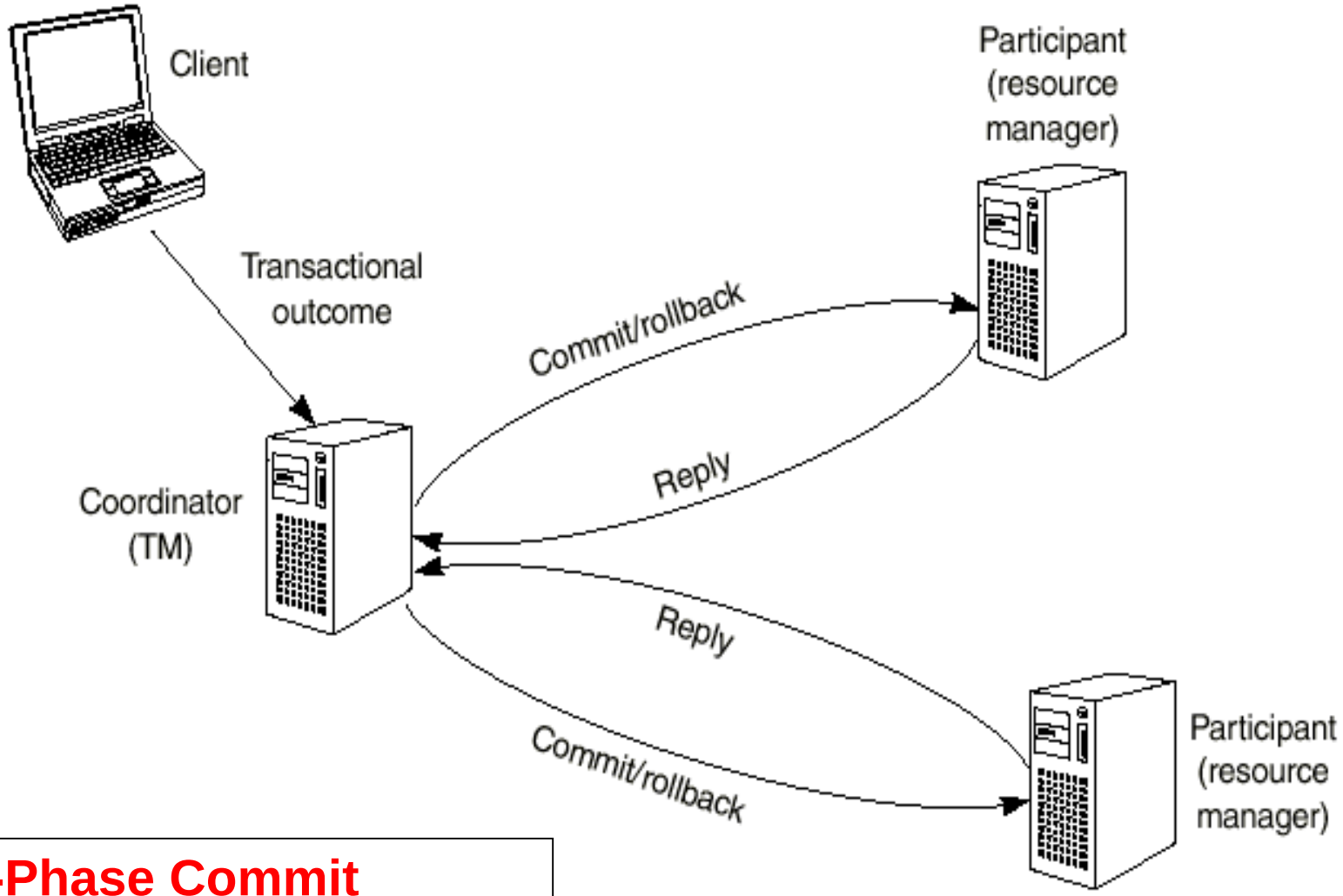
Wenn alle Slaves positiv andworten sendet der Master eine commit Aufforderung, worauf die Slaves die tentative Änderung permanent machen. Wenn einer der Slaves seine Transaktion nicht durchführen kann senst der Master an die Slaves eine Nachricht die tentative Änderung wieder rückgängig zu machen (Phase 2).

Alle Slaves bestätigen den Abschluss der 2-Phase Commit Transaktion.

Ende



2-Phase Commit
Grafische Darstellung Phase 1



2-Phase Commit
Grafische Darstellung Phase 2

Resource Recovery Services

Resource Recovery Services (RRS) provides a global syncpoint manager that any resource manager on z/OS can exploit. It enables transactions to update protected resources managed by many resource managers.

RRS is increasingly becoming a prerequisite for new resource managers, and for new capabilities in existing resource managers. Rather than having to implement their own two-phase commit protocol, these products can use the support provided by RRS.

Since transaction managers like CICS already offer many of the benefits of RRS for processing their own data, not many people rushed to exploit RRS when it was first introduced. However, as more transaction managers have become RRS resource managers, and as the complexity of the exchanges of transactional data increases, more and more systems and application programmers will need to use RRS.

http://www.redbooks.ibm.com/abstracts/sg246980.html

CICS
Customer Information Control System

CICS ist der am weitesten verbreitete, IBM proprietäre Transaktionsmonitor.

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

CICS hat eine Spitzenposition bezüglich Durchsatz, Zuverlässigkeit und Verfügbarkeit.

Mit CICS werden weltweit mehr als 30 Milliarden Transaktionen pro Tag verarbeitet.

Pro Tag wird mit CICS Transaktionen weltweit ein Geldbetrag von > 1 Billion (10^{12}) Dollar transferiert. Mehr als 30 Millionen Sachbearbeiter benutzen CICS bei ihrer täglichen Arbeit. 900 000 Benutzer können gleichzeitig auf einem CICS Rechner arbeiten.

Die folgende Aussage verdeutlicht die Bedeutung von CICS:

In 2001 war die Anzahl der weltweit ausgeführten CICS Transaktionen etwa so groß wie die Anzahl der Hits auf Seiten des World-Wide Web.

In 2001 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 .

Investition in Anwendungen

CICS Programme sind teilweise vor vielen Jahren oder Jahrzehnten geschrieben worden. Sie arbeiten zuverlässig und performant. Ihre Struktur entspricht aber nicht modernen Anforderungen. Es taucht deshalb immer wieder die Idee auf, die z.T. sehr alten CICS Programme neu zu schreiben und bei der Gelegenheit moderne Software Engineering Techniken einzusetzen.

Eine gute Gelegenheit hierzu bot sich bei der „Year 2000“ Umstellung, die allen Unternehmen viel Geld gekostet hat. Es ist dennoch fast nirgendwo geschehen. Hierfür existieren zwei Gründe.

1. Die erforderlichen Investitttionen sind viel zu hoch. Dies ist eine Überschlagsrechnung mit den folgenden Annahmen:

- **20 000 zSeries und S/390 Servers haben durchschnittlich 1 Mill. Zeilen aktiven Anwendungscode (zwischen 200 000 und 50 Millionen pro Server), kumulativ 20 Milliarden LOC.**
- **Bei einer Produktivität von 2 000 LOC/Mannjahr ergibt sich eine Investition von 10 Millionen Mannjahren.**
- **Angenommen 100 000 $/Mannjahr ergibt eine Investition von 1 Billion $ in S/390 Anwendungssoftware. Zum Vergleich, das USA 1999 GNP war 9 Billion $.**

http://www-3.ibm.com/developer/solutionsevent/pdfs/spector_lunchtime_keynote.pdf

I

In der Weltwirtschaft ist weder das Geld und erst recht nicht die erforderliche Anzahl an Programmierern vonhanden um existierende Anwendungen ohne Not umzuschreiben. Vorhandene Ressourcen werden dringend für anstehende Aufgaben benötigt.

2. Die vorhandenen Anwendungen arbeiten zuverlässig und performant. In vielen Fällen fehlen Glaube und Überzeugung, dass das Neuschreiben existierender Anwendungen eine messbare Verbesserung bringen würde.

CICS Transaktionsmonitor

Bei allen transaktionskritischen Funktionen rufen CICS Anwendungsprogramme den Betriebssystem-Kernel nie direkt auf. Statt dessen führt ein CICS Anwendungsprogramm spezielle *EXEC CICS* Kommandos aus, um Betriebssystem-Funktionen zu bewirken wie:

- **Dateizugriffe,**
- **Klienten-Kommunikation**
- **Prozesswechsel oder**
- **Funktionen, die System Resourcen erfordern.**

EXEC CICS Kommandos werden vom „CICS Nucleus“ im User Status ausgeführt. Beispielsweise kann ein EXEC CICS WRITE Kommando die Funktion mehrerer System Calls beinhalten:

- **Zugriff auf die Lock Datenbank**
- **Zugriff auf die Log Datenbank**
- **Speicherung der Daten**

Der CICS TS Transaktionsmonitor verhält sich wie ein Mini-Betriebssystem unterhalb des tatsächlichen Betriebssystems. CICS stellt damit eine Laufzeitumgebung für den Ablauf von CICS Transaktionen zur Verfügung.

Anders als reguläre Programme führen CICS Anwendungsprogramme keine direkten Aufrufe des Betriebssystems aus. Statt dessen enthalten CICS Anwendungsprogramme Aufrufe des CICS Nucleus um Funktionen wie Terminal I/O, File I/O, Program Control usw. auszuführen, welche Kernel Funktionen erfordern.

CICS TS verhält sich wie ein Mini-Betriebssystemkernel innerhalb des tatsächlichen Betriebssystems um eine Umgebung für die Ausführung von Anwendungsprogrammen bereitzustellen.

Aufrufe des CICS Nucleus fangen immer mit der Seuuenz EXEC CICS an, und hören mit dem Statement Delimiterder Programmiersprache auf, in der das ECEC CICS Statement eingebettet ist, z. B ; in C++ oder END in Cobol.

Beispiel für einen Aufruf des CICS Nucleus innerhalb eines C++-Programms:

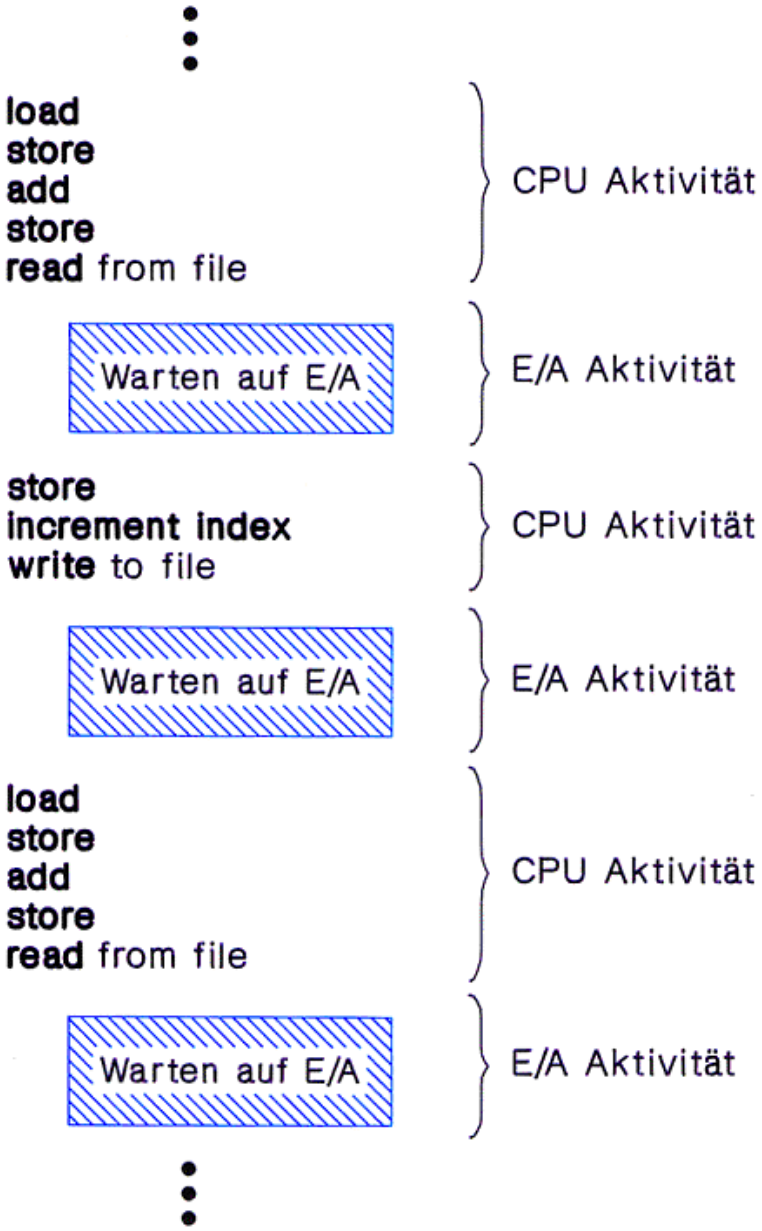
```
EXEC CICS SEND MAP("label04") MAPSET("s04set") ERASE;
```

Neue CICS Anwendungen werden häufig in C oder Java geschrieben. Daneben hat Cobol nach wie vor eine überragende Bedeutung.

Beispiel für ein CICS-Statement innerhalb eines COBOL-Programms

```
EXEC CICS
WRITEQ TS QUEUE(‘ACCTLOG’) FROM(ACCTDTLO)
LENGTH(DTL-LNG)
END EXEC
```

Ein existierender Datensatz „ACCTDTLO“ wird in eine temporäre Warteschlange ACCTLOG geschrieben, die als Log zur Datensicherung dient



Struktur einer Transaktion

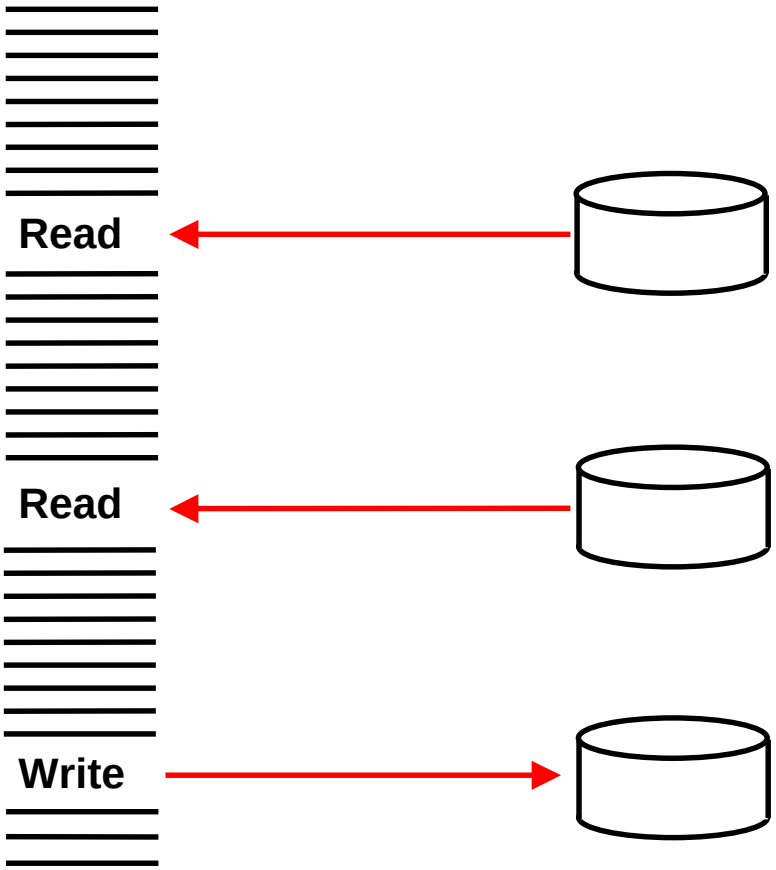
Eine Transaktion (und auch die meisten nicht-transaktionalen Anwendungsprogramme) besteht aus einer Folge von

- **Gruppen von Maschinenbefehlen**
- **I/O Operationen (Zugriff auf Plattenspeicher-Daten).**

Die Programmausführung ist eine abwechselnde folge von CPU und I/O Aktivitäten.

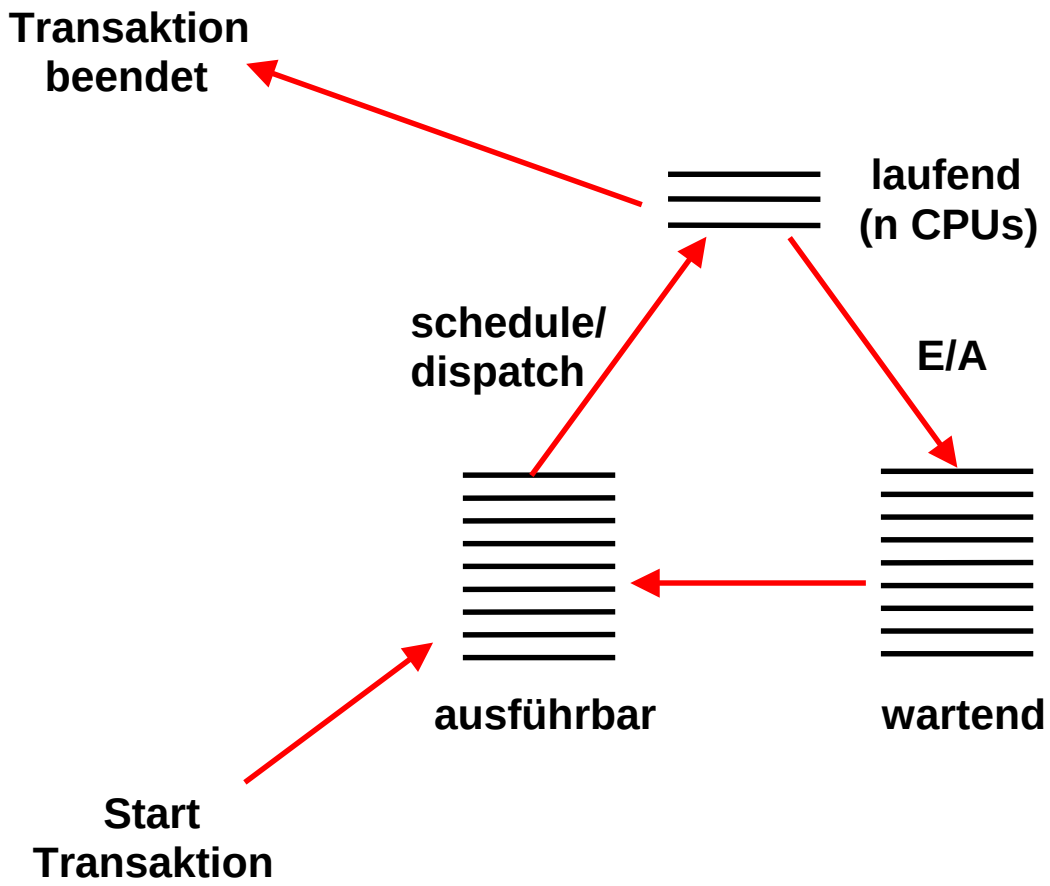
Beim Start einer I/O Operation muss das Anwendungsprogramm in der Regel warten, bis die I/O Operation abgeschlossen ist (z.B. die vom Plattenspeicher gelesenen Daten im Hauptspeicher eingetroffen sind).

Es wäre unökonomisch, wenn die CPU während der I/O Aktivität warten müsste.



Ablauf einer CICS Transaktion

Das Programm für die Ausführung einer Transaktion besteht aus Folgen von Maschinenbefehlen, in die in unregelmäßigen Abständen Bibliotheksaufrufe eingebaut sind, z.B. für die Ausführung von I/O Opeerationen. Während des Zeitabschnittes in der die Read/Write Operation durchgeführt wird verarbeitet die CPU eine andere Transaktion. Dies geschieht deshalb, weil eine Befehlsausführung etwa 1 ns benötigt, während ein Plattenspeicherzugriff etwa 10 ms braucht.



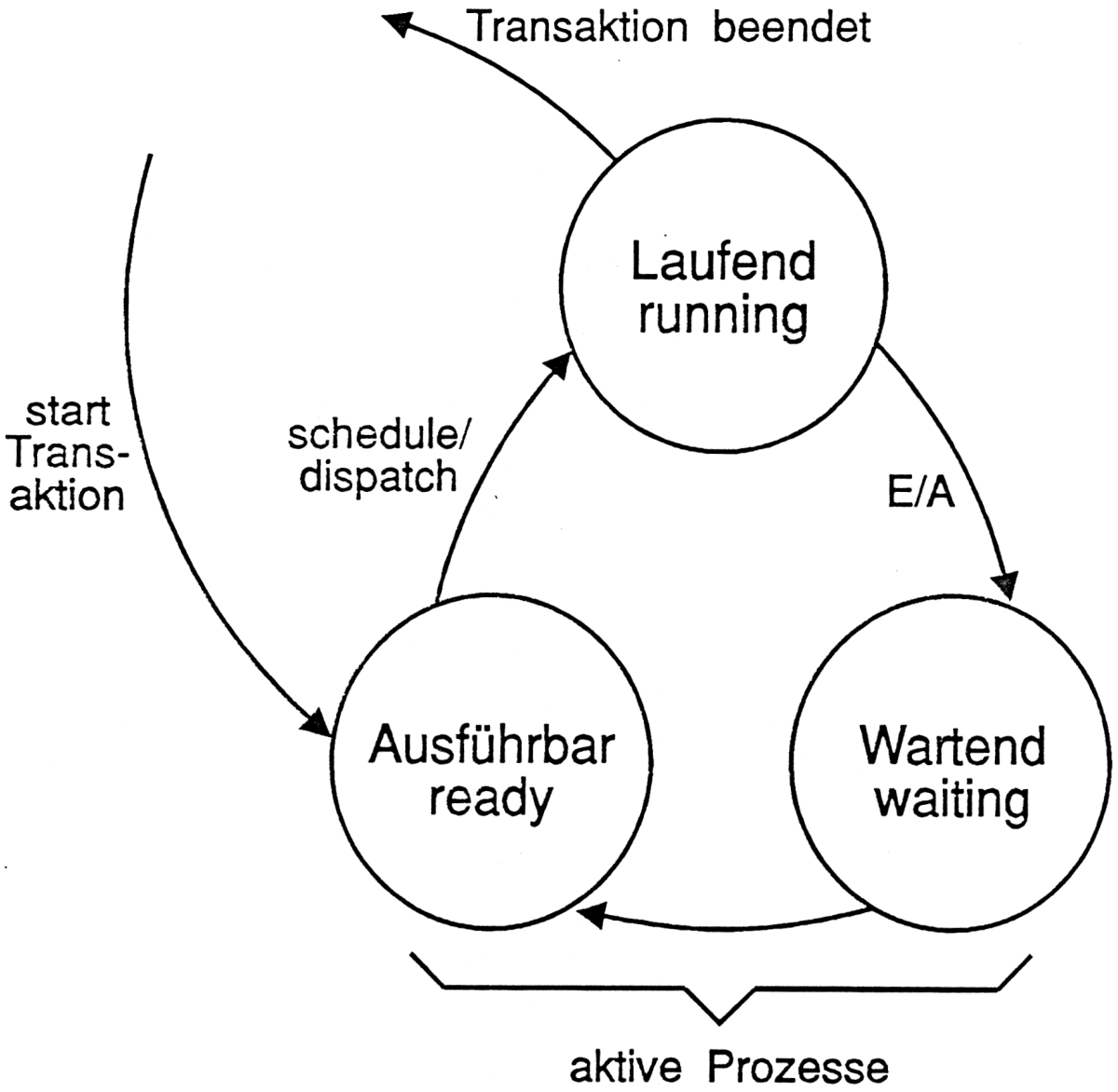
Die Prozesse befinden sich immer in einem von drei Zuständen.

Ein Prozess befindet sich im Zustand laufend, wenn er von einer der verfügbaren CPUs ausgeführt wird. (Die allermeisten Mainframes sind Mehrfachrechner und verfügen über mehrere CPUs, bis zu 64 bei einem z10 EC).

Ein Prozess befindet sich im Zustand wartend, wenn er auf den Abschluss einer I/O Operation wartet.

Ein Prozess befindet sich im Zustand ausführbar, wenn er darauf wartet, dass die Scheduler/Dispatcher Komponente ihn auf einer frei geworden CPU in den Zustand laufend versetzt.

Zustand von Prozessen



Eine Transaktion (Prozess) wechselt während ihrer Ausführung ständig zwischen den Zuständen laufend, wartend und ausführbar.

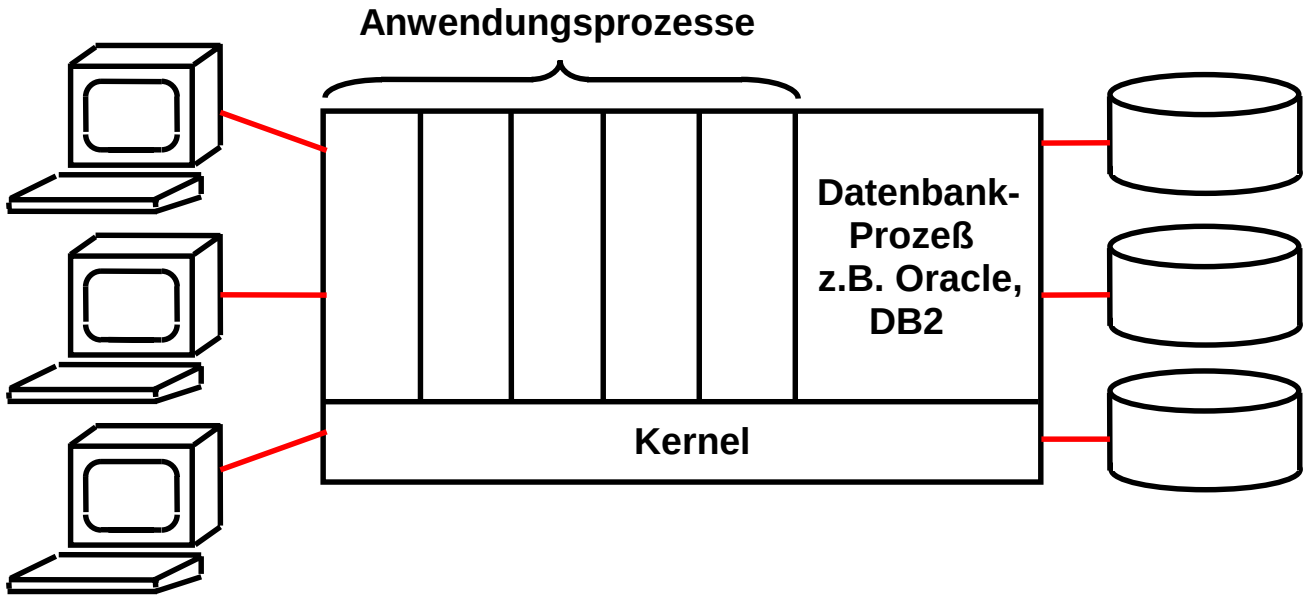
Die TCBs (Task Control Blöcke) der einzelnen aktiven Prozesse werden in drei Queues für laufende, wartende und ausführbare Prozesse verwaltet.

In jedem Augenblick können Tausende von Transaktionen gleichzeitig aktiv sein. Sie wechseln ständig zwischen den drei Zuständen.

Eine neu hinzukommende Transaktion wird zunächst in den Zustand Ausführbar versetzt.

Wenn eine Transaktion beendet wurde, verlässt sie die Menge der aktiven Prozesse.

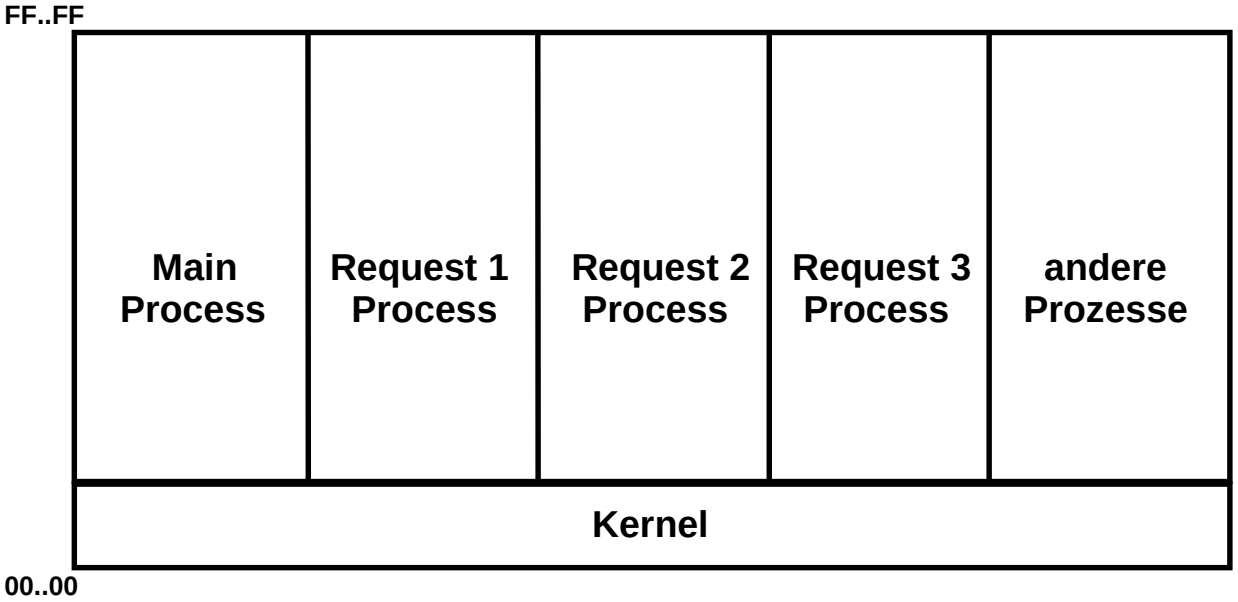
Ausführungsmodell für Transaktionen



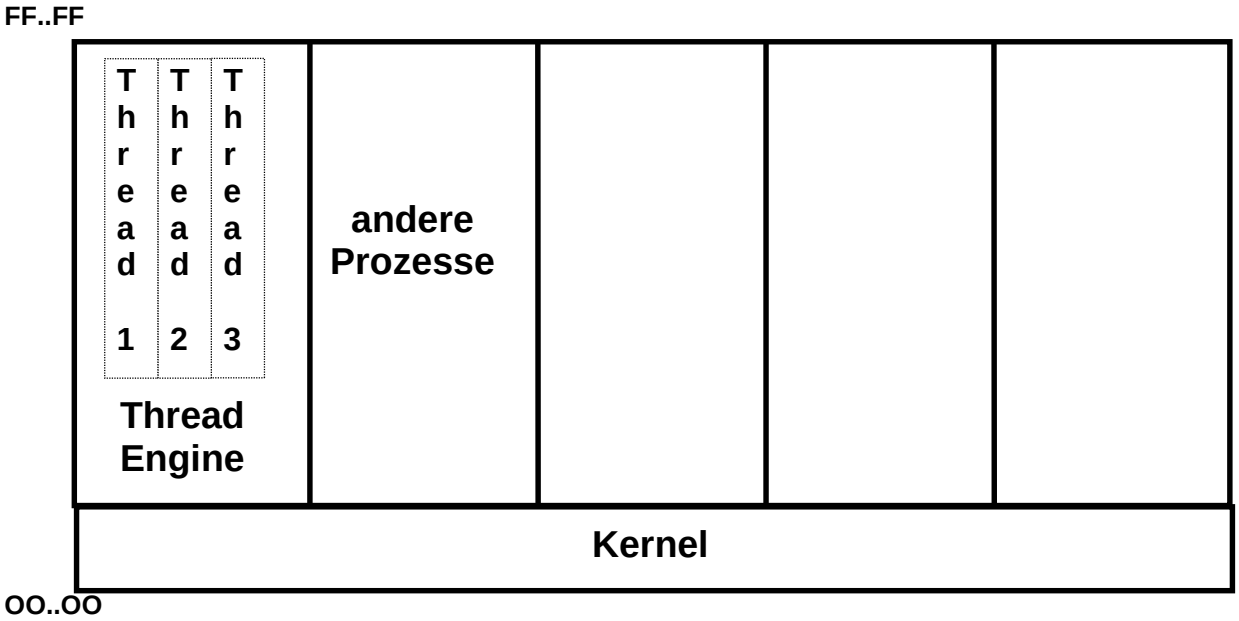
Multiprogrammierte Verarbeitung von Transaktionen

In jedem Augenblick verarbeitet der Rechner gleichzeitig viele Transaktionen. Pro CPU ist nur eine Transaktion aktiv, die anderen sind ausführbar oder warten auf den Abschluss einer Ein-/Ausgabeoperation. Hunderte oder Tausende paralleler Transaktionen sind denkbar.

Es muss verhindert werden, dass mehrere Transaktionen gleichzeitig auf die gleichen Daten zugreifen. Hierzu werden Leseberechtigungen und Schreibberechtigungen mit Hilfe von Locks (Sperren) verwaltet.



Ansatz mit mehreren Prozes



Thread Ansatz

Ein Hochleistungs-Transaktions-system verarbeitet in jedem Augenblick Hunderte oder Tausende von Transaktionen gleichzeitig und parallel. Für die Verarbeitung gibt es zwei Alternativen:

- **1 Prozess pro Transaktion**
- **1 Thread pro Transaktion)**

1 Prozess pro Transaktion bedeutet, dass jede Transaktion in einem eigenen virtuellen Adressenraum (z/OS Region) läuft. Vorteilhaft ist, dass die Isolation der Transaktionen untereinander (das i in ACID) optimal gewährleistet ist. Nachteilig ist der höhere Verarbeitungsaufwand im Vergleich zum Thread Ansatz.

CICS Process Model

1. z/OS and OS/390 [Mainframe]

Multiple applications and users using a single operating system address space and operating system task concurrently

CICS manages its own execution units within a single operating system task

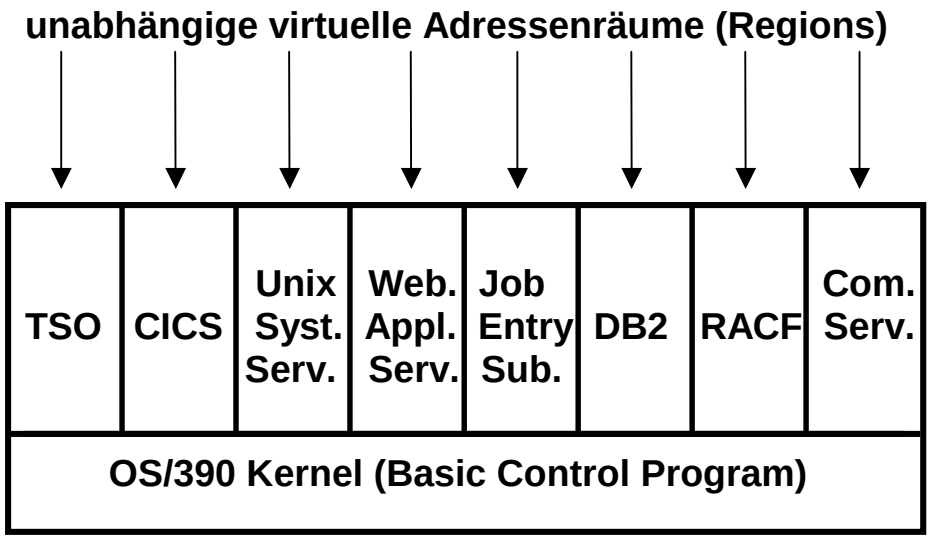
"Quasi Reentrant" - an application can lose control when executing CICS API requests only

2. Other platforms

Multiple applications and users using multiple operating system processes

One application running at a time in a process

"Preemptive" - application can lose control at any point

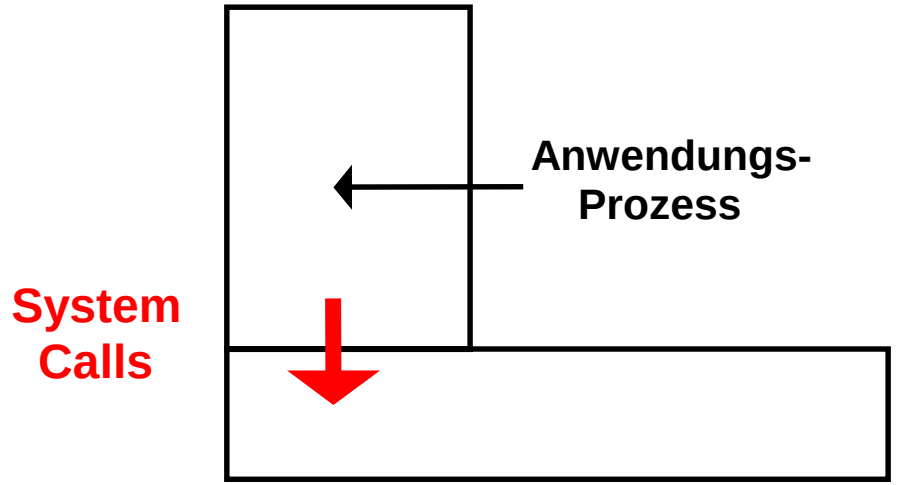


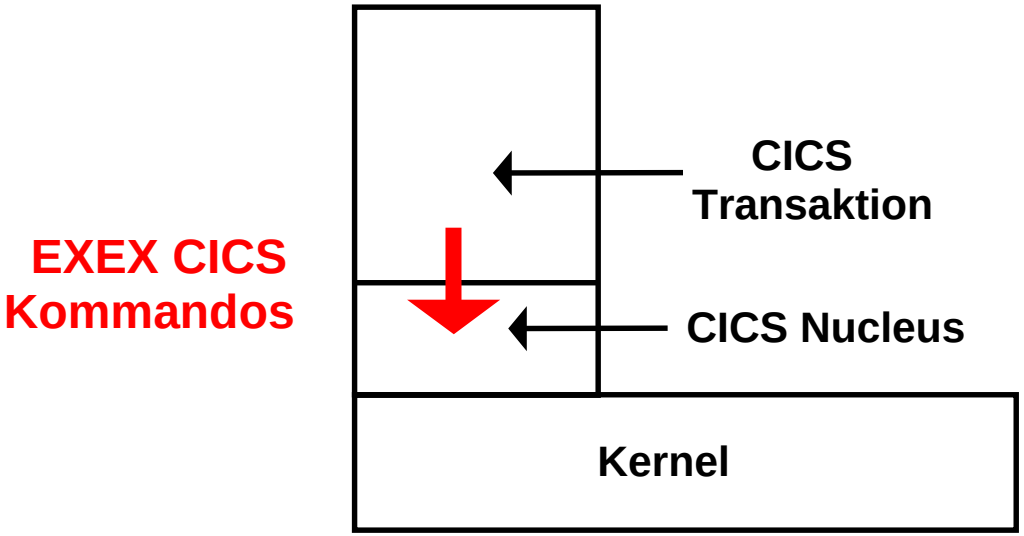
z/OS Grundstruktur

Der z/90 Kernel unterstützt eine Vielzahl von virtuellen Adressenräumen, die im OS/390 Jargon als Regions bezeichnet werden.

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

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

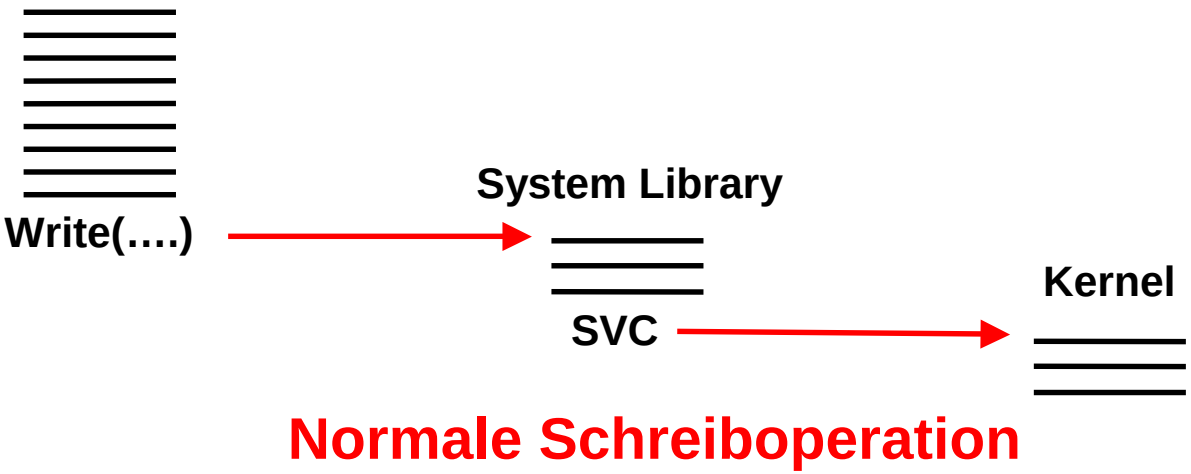




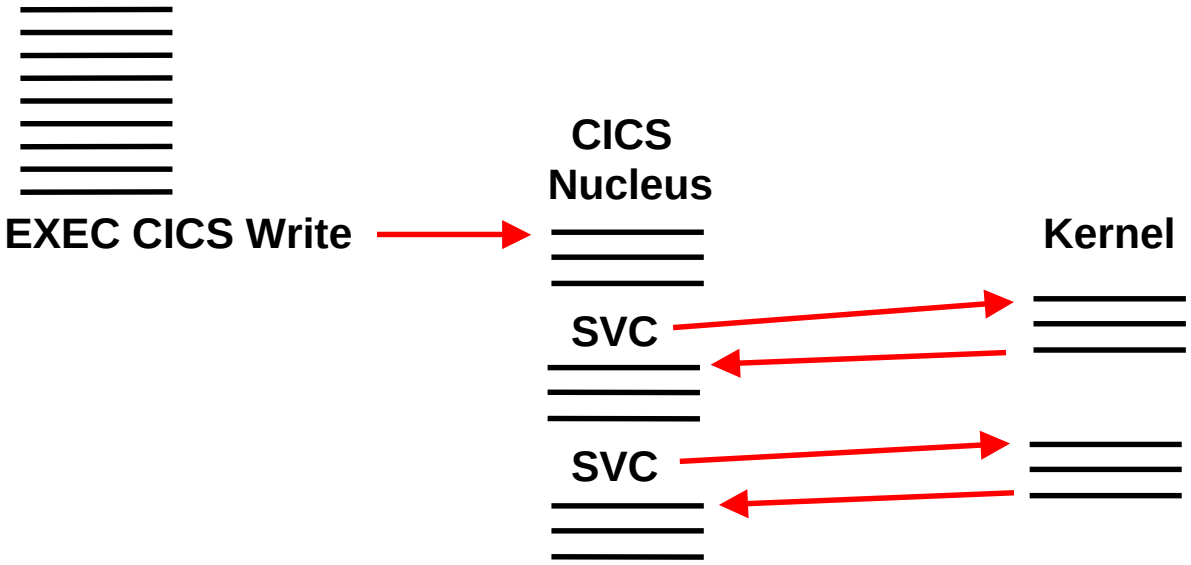
Ein normaler Anwendungsprozess nimmt Dienstleistungen des Kernels über System Calls wie z.B. OPEN oder READ in Anspruch.

Eine CICS Transaktion verwendet statt dessen EXEC CICS Kommandos, die Dienstleistungen einer speziellen CICS Komponente (CICS Nucleus) in Anspruch nehmen.

Der CICS Nucleus ruft bei Bedarf den Kernel über normale System Calls auf.



Normale Schreiboperation



CICS Schreiboperation

Beispiel: Write Operation

Eine normale Write Operation führt zum Aufruf einer Library Routine, die in den Kernel mittels eines SVC Maschinenbefehls verzweigt und diesen ausführt.

CICS ersetzt die Write Operation durch eine EXEC CICS Write Operation. Diese führt zum Aufruf einer Routine des CICS Nucleus, welche die ACID Eigenschaften sicherstellt, indem u.A. Einträge in eine LOG Datei und eine Lock Datei erfolgen, Voraus-setzungen für ein evtl Rollback geschaffen werden usw.

CICS Nucleus

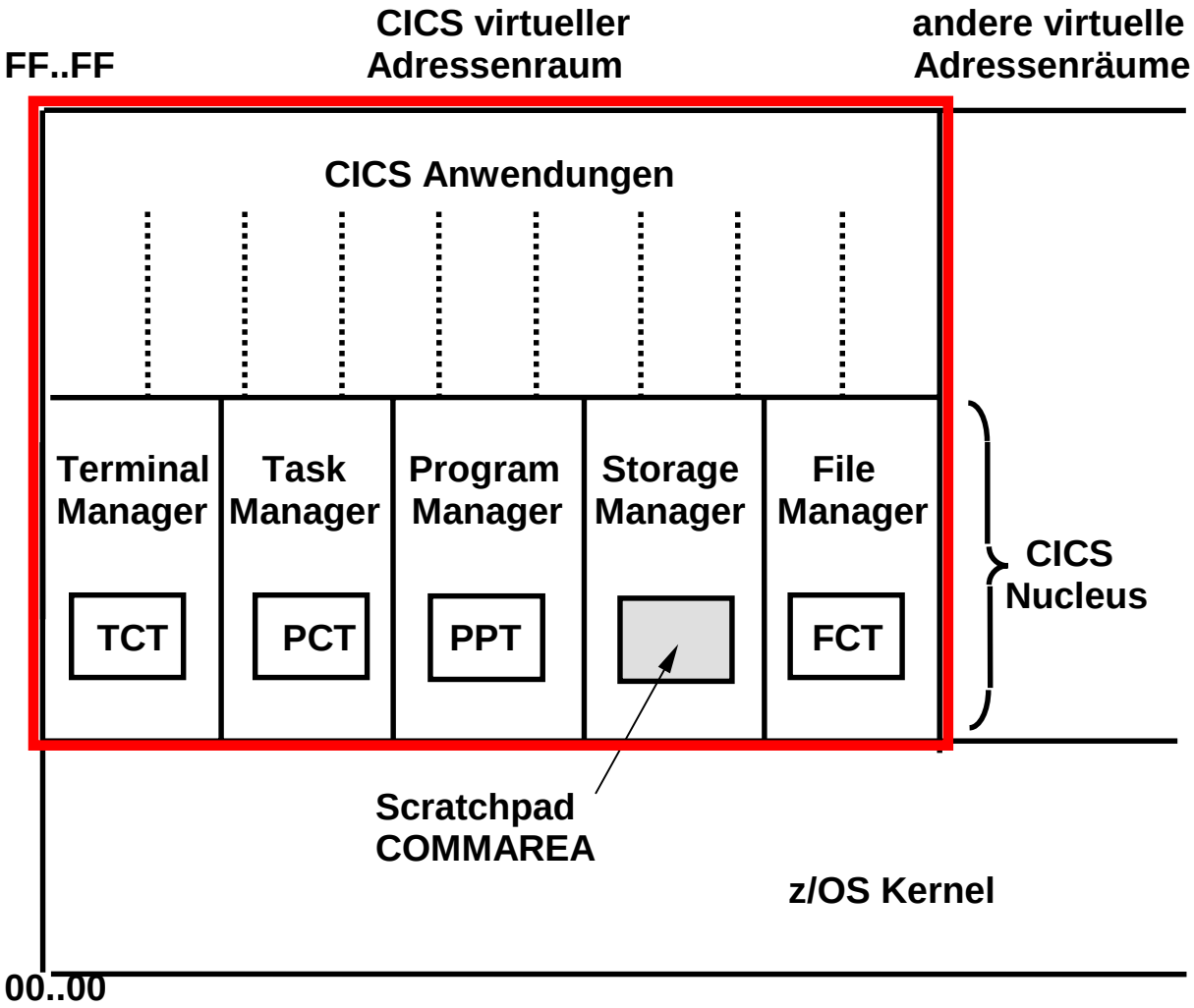
CICS verhält sich wie ein Mini-Betriebssystem unterhalb des eigentlichen Betriebssystem und stellt eine Umgebung für die Ausführung von CICS Anwendungsprogrammen (Transaktionen) zur Verfügung.

Unter z/OS läuft CICS in einem eigenen virtuellen Adressenraum (Region). Alle CICS Transaktionen laufen in der CICS Region unter Kontrolle von CICS. CICS Anwendungsprogramme benutzen CICS für alle Schnittstellen (interfaces). CICS wiederum greift auf das z/OS Betriebssystem zu.

CICS ist die Schnittstelle zwischen Anwendungs-programmen, Datenbanken und Teleprocessing (Communication) Zugriffsmethoden.

Aus Sicht des Betriebssystems existiert nur eine einzige Anwendung und nur ein Prozess – der CICS Transaktionsmonitor.

"Quasi Reentrant" - eine CICS-Anwendung wird nur bei der Ausführung einer CICS API (EXEC CICS ...) unterbrochen.



CICS Komponenten

CICS läuft als lang laufender Stapelverarbeitungsjob in einem einzigen virtuellen Adressenraum (Region in OS/390 Terminologie). CICS Anwendungsprogramme laufen „run to completion“; Interactivität wird programmtechnisch gewährleistet, indem ihre maximale Ausführungszeit eine vorgegebene Grenze nicht überschreitet.

Die CICS Nucleus Komponenten (Terminal Manager, Task Manager, Program Manager, Storage Manager and File Manager) 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 Nucleus Komponenten

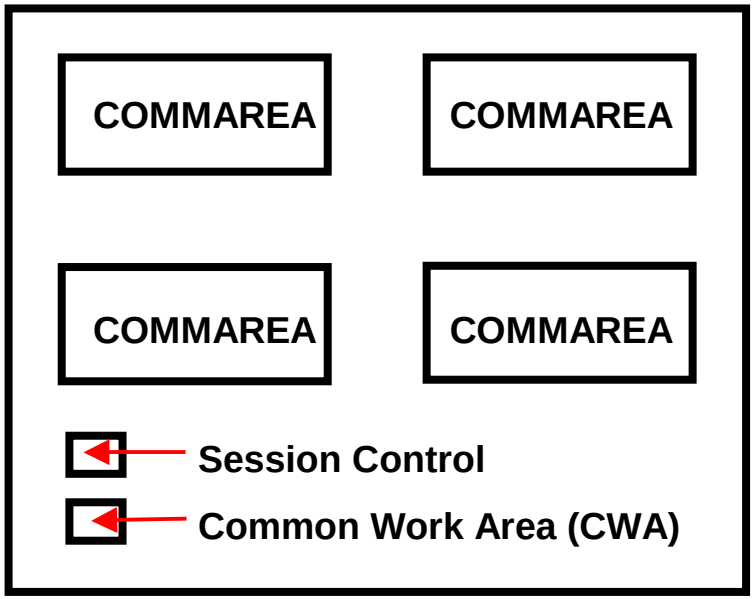
Die CICS Nucleus Komponenten (Terminal Control, Task Control, Program Control, Storage Control and File Control) nutzen den gleichen virtuellen Adressenraum wie alle Anwendungen. Jede Nucleus Komponente hat eine zugeordnete Tabelle: TCT, PCT, PPT, FCT. Über COMMAREA werden Sessions eingerichtet: Der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

Die CICS Nucleus Komponenten laufen in *Domains*. Domains enthalten Programme, Tabellen und Steuerblöcke. Die Überwachung und Steuerung der Transaktions-verarbeitung erfolgt vor allem durch drei Komponenten:

- **Task Control (andere Bezeichnung: Transaction Manager,XM) ist für den Empfang von *Transaction Requests* zuständig, sowie für die Erstellung und Steuerung von *Tasks*, welche die Transaction Requests verarbeiten.**

- **Program Control (andere Bezeichnung: Program Manager, PG) ist zuständig für das Laden von Anwendungsprogrammen in den Hauptspeicher sowie deren anschließende Ausführung. Auch wenn ein Programm von mehreren gleichzeitig laufenden Transaktionen benutzt wird, befindet sich nur eine einzige Kopie des Programms im Hauptspeicher.**

- **Storage Control (andere Bezeichnung: Storage Manager, SM) ist zuständig für die Zuordnung von (virtuellem) Speicherplatz, der für die Transaktionsverarbeitung benötigt wird.**



Scratchpad

Der Scratchpad-Bereich innerhalb des virtuellen Speichers wird von der Storage Manager Komponente des CICS Nucleus für interne Verarbeitungsabläufe benutzt.

Innerhalb des Scratchpads wird für jeden aktiven Klienten eine COMMAREA eingerichtet.

Scratchpad

Der Scratchpad-Speicherbereich innerhalb des Hauptspeichers wird von der Storage Manager Komponente des CICS Nucleus für interne Verarbeitungsabläufe benutzt.

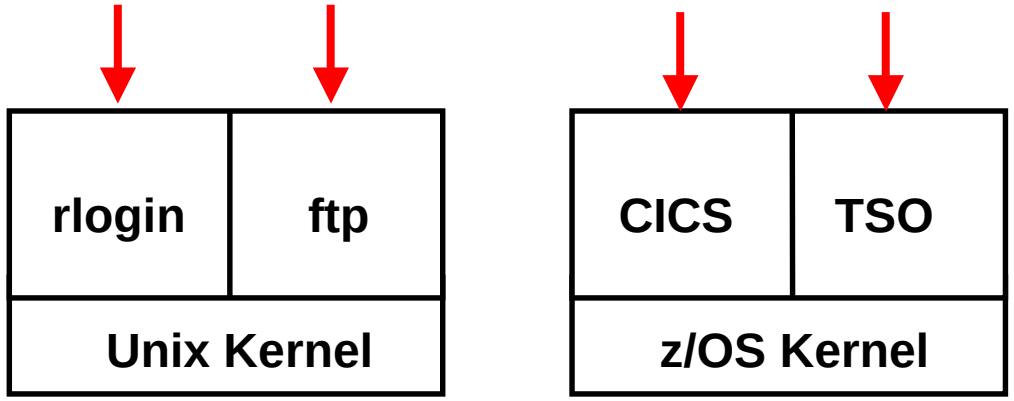
Innerhalb des Scratchpads wird für jeden aktiven Klienten eine COMMAREA eingerichtet.

Der Begriff COMMAREA ist doppelt belegt. Zum einen werden individuelle Ein-/Ausgabe Puffer innerhalb des Scratchpads als COMMAREA bezeichnet. Ein Scratchpad kann mehrere COMMAREA Ein-/Ausgabepuffer enthalten, daneben aber aich zahlreiche anderen Informationen.

Zum anderen wird zweideutig der Scratchpad-Speicherbereich innerhalb des Hauptspeichers manchmal ebenfalls als COMMAREA bezeichnet.

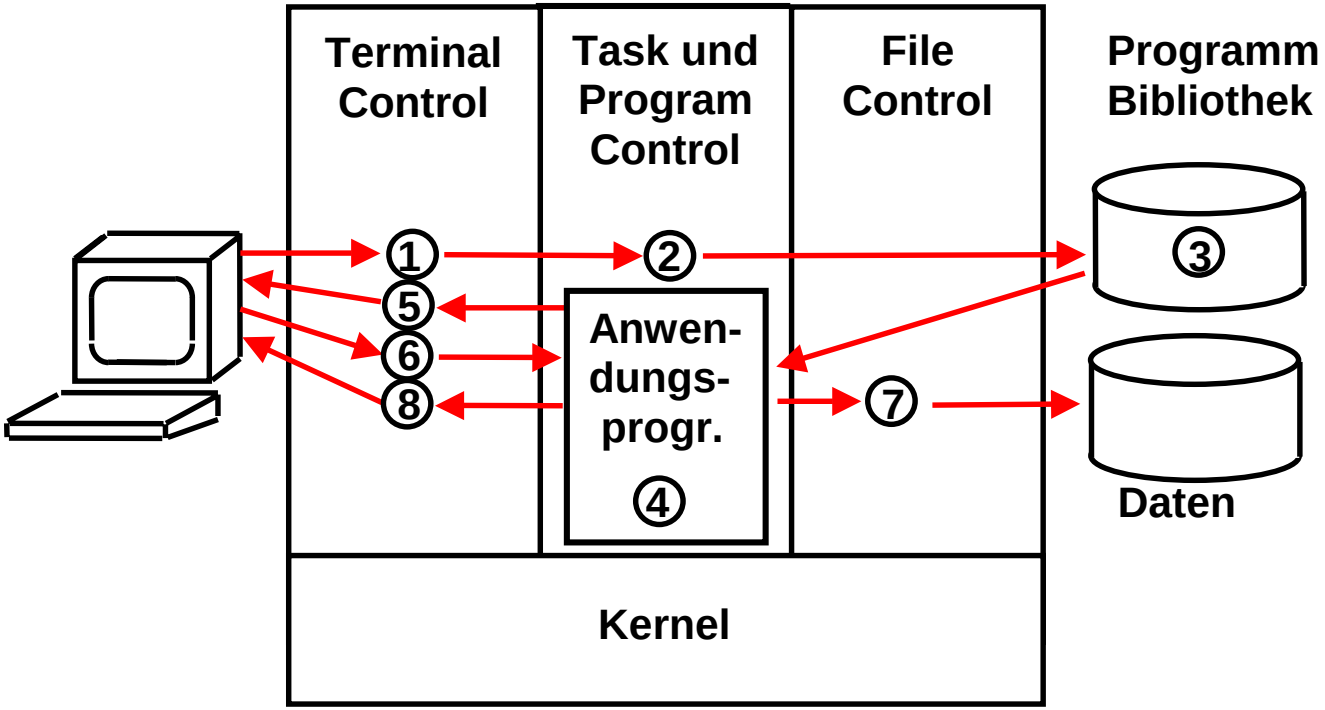
In einem Präsentationslogik-Anwendungs-Programm (z.B. in Java) bezeichnet der Begriff COMMAREA fast immer einen Ein-/Ausgabe Puffer.

Der Inhalt eines COMMAREA Ein-/Ausgabepuffers wird häufig als Record, Ein-/Ausgaberecord oder Unit Record bezeichnet.



Aufbau einer Sitzung

Klient logged sich in das CICS Subsystem ein. (Unter z/OS stehen mehrere Subsysteme für ein remote Login zur Verfügung, z.B. TSO).



Ablaufsteuerung einer CICS Transaktion

Ablaufsteuerung einer CICS Transaktion

1. **Klient logged sich in das CICS Subsystem ein. Eine neue Transaktion ruft CICS mit Hilfe ihrer Transaction ID (TRID) auf. Terminal Control übernimmt die Eingabe und speichert sie ab**
2. **CICS interpretiert die Nachricht als Transaktion und ruft das entsprechende Anwendungsprogramm auf.**
3. **Das Anwendungsprogramm befindet sich entweder schon im Arbeitsspeicher oder wird vom Program Manager aus der Programmbibliothek geladen**
4. **Ein CICS Prozess (Task) wird erzeugt der die Anwendung ausführt**
5. **Terminal Control baut ein Bildschirm Menü auf (z.B. mit BMS oder mit Java Präsentationslogik) welche dem Benutzer eine Spezifikation der durchzuführenden Aktivität ermöglicht**
6. **Weitere Eingaben werden von Terminal Control entgegengenommen und zur Verarbeitung weitergereicht**
7. **File Control liest gewünschte Daten aus einer Datenbank**
8. **Die gelesenen Daten (Unit Record) werden von Terminal Control aufbereitet und auf dem Bildschirm angezeigt**

Ablaufsteuerung einer CICS Transaktion

1. **Ein Klient logged sich in das CICS Subsystem ein. Die Nachricht enthält die Adresse des Klienten. Eine neue Transaktion ruft CICS mit Hilfe ihrer Transaction ID (TRID) auf. CICS Terminal Control prüft, ob eine Session mit dem Klienten besteht. Wenn nein, werden die ersten 4 Bytes der Nachrichten als TRID interpretiert. Terminal Control übernimmt die Eingabe und speichert sie ab**

2. **Die Nachricht mit der TRID wird an CICS Task Control weitergegeben. Ein CICS Prozess (Task) wird erzeugt der die Anwendung ausführt. Information über alle laufenden Transaktionen ist im TRID Table festgehalten. Das Anwendungsprogramm befindet sich entweder schon im Arbeitsspeicher oder wird vom Program Manager aus der Programmbibliothek geladen. Der Processing Program Table PPT enthält Information über alle CICS interne und alle Benutzer geschriebenen Anwendungen.**

3. **CICS Program Control lädt bei Bedarf das Anwendungsprogramm.**

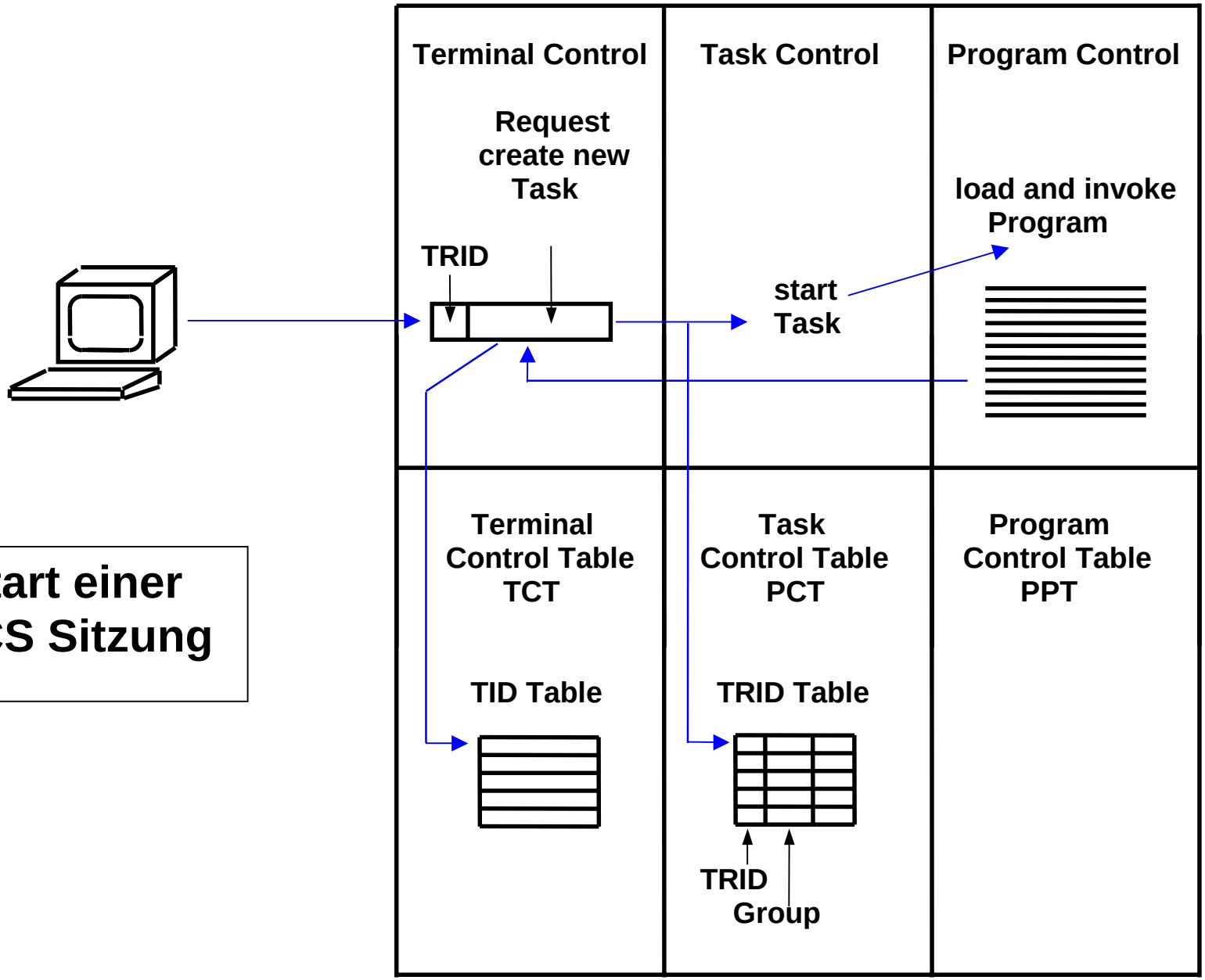
4. **Task Control liest aus seinem TRID Table die zu der TRID gehörige Group aus, darunter Referenz auf Mapset und Anwendungsprogramm. Das Anwendungsprogramm liest die Nachricht des Klienten.**

5. **File Control liest gewünschte Daten aus einer Datenbank.**

6. **Die gelesenen Daten (Unit Record) werden von Terminal Control aufbereitet Terminal Control baut ein Bildschirm Menü auf (z.B. mit BMS oder mit Java Präsentationslogik) und auf dem Bildschirm angezeigt.**

7. **Wenn vom Klienten die nächste Nachricht eintrifft, erinnert sich CICS Terminal Control, daß eine Sitzung bereits besteht. Weitere Eingaben werden von Terminal Control entgegengenommen und zur Verarbeitung Task Control weitergereicht Die Nachricht wird unmittelbar an CICS Task Control weitergereicht.**

Start einer CICS Sitzung



Transaction processing - Steps

These steps are involved in processing a transaction:

1) Entry – A transaction ID (TRANSID) enters the CICS system.

2) Task creation – CICS creates a task to process the transaction. The task is now ready to be run.

3) Dispatch – CICS determines which of the ready tasks should be run next, and dispatches that task to be started.

4) Execution – The task invokes the appropriate CICS program and runs.

5) Processing – When the invoked program calls CICS to perform a service on its behalf, the task gives up control of the CPU and waits for the requested service to be completed.

6) Redispatch – After the requested service has been completed, the task is ready to run again, and CICS dispatches it again.

7) Return – When all work required to process the TRANSID is done, the program issues a RETURN command to return control to CICS.

8) Termination – CICS removes the task from the system.

Nachname	Schmitz
Vorname	Stefan
Per. Nr.	34567
Straße	Herdweg. 92
PLZ	71032
Wohnort	Böblingen

Nachname	Müller
Vorname	Fritz
Per. Nr.	12345
Straße	Ahornstr. 29
PLZ	70178
Wohnort	Stuttgart

Nachname	Meier
Vorname	Boris
Per. Nr.	23456
Straße	Marienstr. 72
PLZ	72076
Wohnort	Tübingen

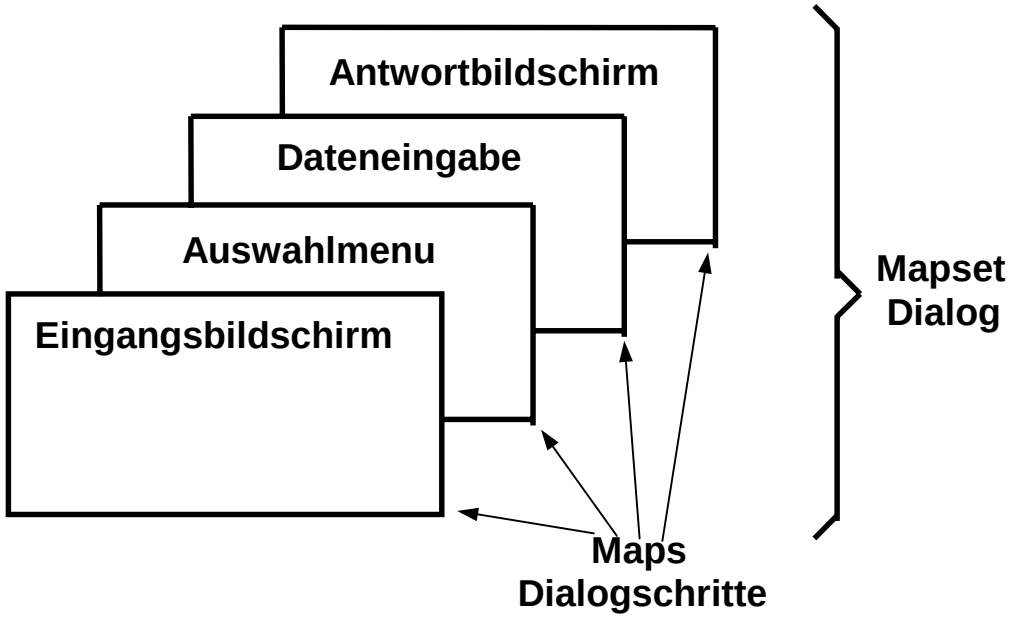
Dargestellt sind drei Bildschirmwiedergaben, welche die gleiche Map benutzen.

Die in schwarzen Buchstaben dargestellte Information ist Bestandteil der Map. Sie dient CICS als Schablone für die auszugebende Information.

Die in roten Buchstaben dargestellte Information wird von CICS bei unterschiedlichen anfragen erzeugt und in Felder eingestellt, die innerhalb der M

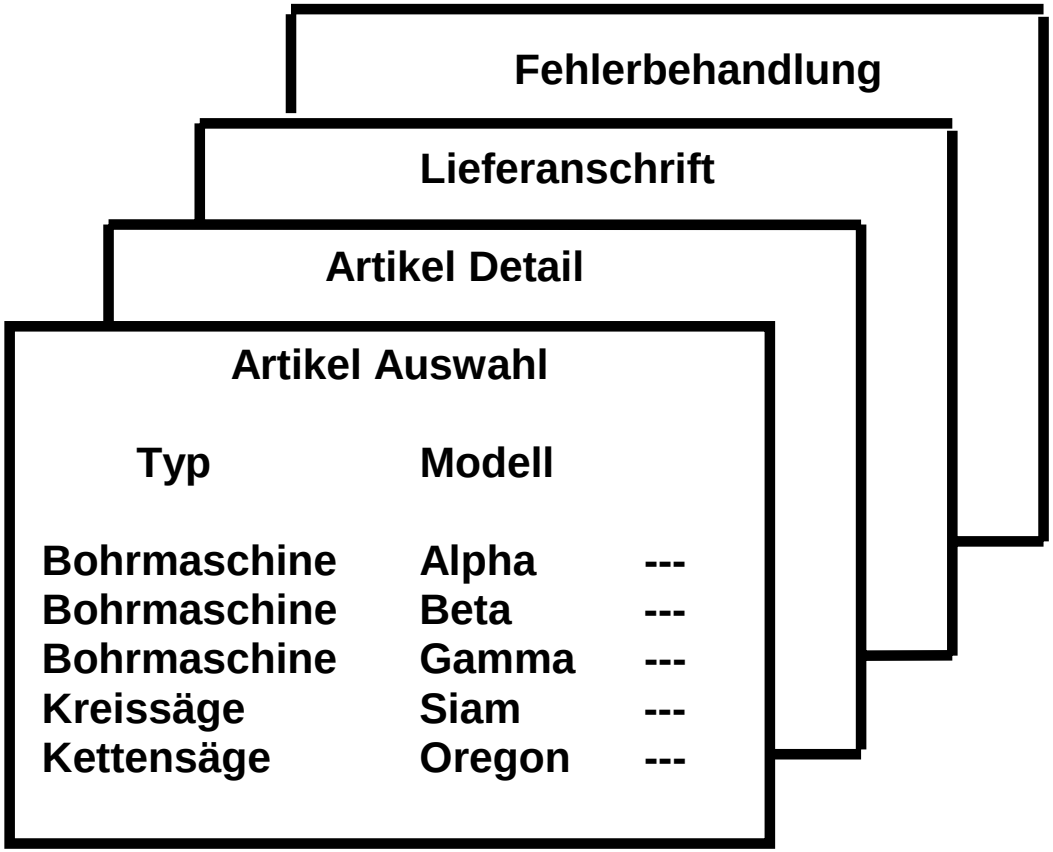
Für den Benutzer am Bildschirm besteht eine Transaktion in der Regel aus mehreren Schritten.

Der Benutzer ruft beispielsweise CICS auf, identifiziert sich und gibt eine TRID (einen Transaktionstyp) ein, trifft eine Auswahl zwischen mehreren Alternativen aus einem Auswahlmenu, und erhält schließlich eine Antwort.



Der statische Inhalt eines Bildschirms wird als „Map“ bezeichnet (Dialogschritte bei SAP R/3). Eine Transaktion wird in der Regel mehrere unterschiedliche Bildschirme (Maps) wiedergeben.

Eine Map enthält ein Gerüst generischer Information Dieses wird während der Transaktionsausführung mit spezifischer Information angereichert. Alle zu einer Transaktion gehörigen Maps werden als Mapset bezeichnet (Dialog bei SAP R/3).



Mapset

Mapset für einen Lieferauftrag besteht aus 4 Maps

Die NACT Transaktion

Im Folgenden schlüpfen wir in die Rolle eines Sachbearbeiters eines Unternehmens, der von seinem Arbetsplatzrechner eine Transaktion aufruft und durchführt.

Die Transaktion hat den Namen NACT. Sie ist auf dem z/OS Rechner der Universität Leipzig http://jedi.informatik.uni-leipzig.de installiert, und ist Bestandteil eines dort verfügbaren Vorrats an Übungen.

NACT wird in einer fiktiven Firma, einem Kaufhaus mit dem Namen KanDoIT eingesetzt. KanDoIT hat tausende von Kunden, an welche Kreditkarten des Kaufhauses ausgegeben werden. Die Kundendatei ist als VSAM Datei implentiert mit je einem Record pro Kunde. Jeder Record enthält Namen, Adresse, Tel. Nr. Kreditgrenze, derzeitiger Kontostand, usw. Jeder Kunde ist durch eine 5-stellige Kontonr. identifiziert, die ihm von Hand zugeordnet wurde. Letztere würde in einer echten Umgebung nicht von Hand sondern automatisch erzeugt werden; diese Komponente fehlt um die Anwendung einfach zu halten.

Literatur

Eine detaillierte Beschreibung der NACT Transaktion ist in einem ausgezeichneten Lahrbuch:

J. Horswill: “Designing & Programming CICS Applications”. O´Reilly, 2000, ISBN 1-56592-676-5

entthalten, welches leider vergriffen ist. Die Business Logik der Anwendung ist in Cobol geschrieben und nahezu unverändert von einer älteren Transaktion ACCT übernommen. Für diese existiert eine hervorragende Dokumentation in dem CICS Application Programming Primer:

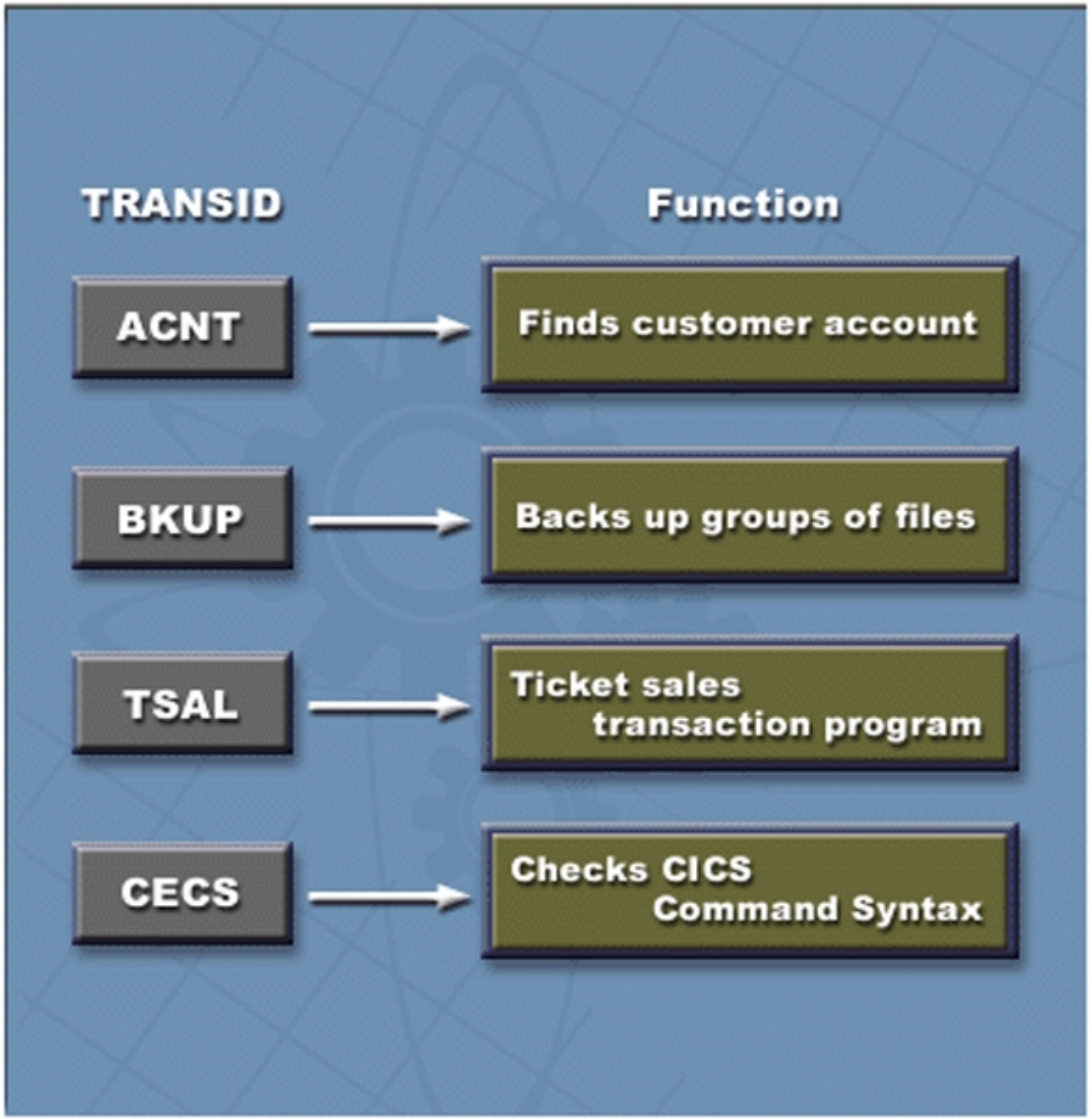
http://www-01.ibm.com/support/docview.wss?uid=pub1sc33067401,

alternativ verfügbar unter http://139.18.4.35/pubs/Lehre/ACCTappl.pdf oder CICS Application Programming Primer.

Der CICS Application Programming Primer enthält eine sehr detaillierte – Codezeile für Codezeile – Beschreibung der NACT bzw. ACCT Cobol Business Logik.

Eine ebenfalls sehr gute Beschreibung ist in einer Diplomarbeit der Universität Leipzig zu finden:

Tobias Busse: Generation of a Java front end for a standalone CICS application accessed through MQSeries.



NACT ist eine *Transaction ID* (TRID). Mit einer TRID wird eine (von vielen) Anwendungen aufgerufen, die CICS als Transaktionen ausführt. Die TRID beschreibt die Funktion der Transaktion. Ein normaler Benutzer ruft einen Dienst eines CICS Servers auf, indem er eine TRID eingibt.

CICS TRID's sind grundsätzlich immer 4 Zeichen lang.

Einige Transaktionen und Ihre TRIDs sind Bestandteil des CICS TP Monitors. Z. B. die CECS Transaktion implementiert eine Kommandozeilen-Shell.

CICS Transactions

In CICS, an instance of a particular transaction request by a computer operator or user is called a *task*. When a user invokes a transaction, CICS begins a task for that request. CICS also loads any application programs required for the transaction.

For example, when a student registration request comes into the CICS system, CICS represents and keeps track of that request and its associated work by starting a unique task for it. CICS then loads the application programs that are required for executing that task.

CICS provides concurrent transaction processing, which means that many users can enter and process requests at the same time. In order to allow for many users while ensuring swift response times, CICS employs multitasking methods.

Under CICS, all users share application programs and data files. This means that if one transaction is being processed and another user make a similar request, CICS does not reload the application program. Instead, CICS starts a new task for the second request, using the same program or data file. CICS runs each task individually, briefly giving each task control of the CPU.

KanDOIT!

Kunden Kredit Karte - Antragsformular

Name Meier Vorname Walter Anrede Dr.

Anschrift Heilbronnerstr. 91
70109 Stuttgart

Telefon 733456

Datum 22. 11. 1999 Unterschrift Dr. Walter Meier

Weitere Kreditkarten

Name Meier, Christa, Ehefrau

Adresse siehe oben

Zur internen Benutzung

Anzahl Karten 2 Konto Nr. 26004

Grund: Überprüft DEF

New, Lost, Stolen, Revised N

Datum 26.11.2006

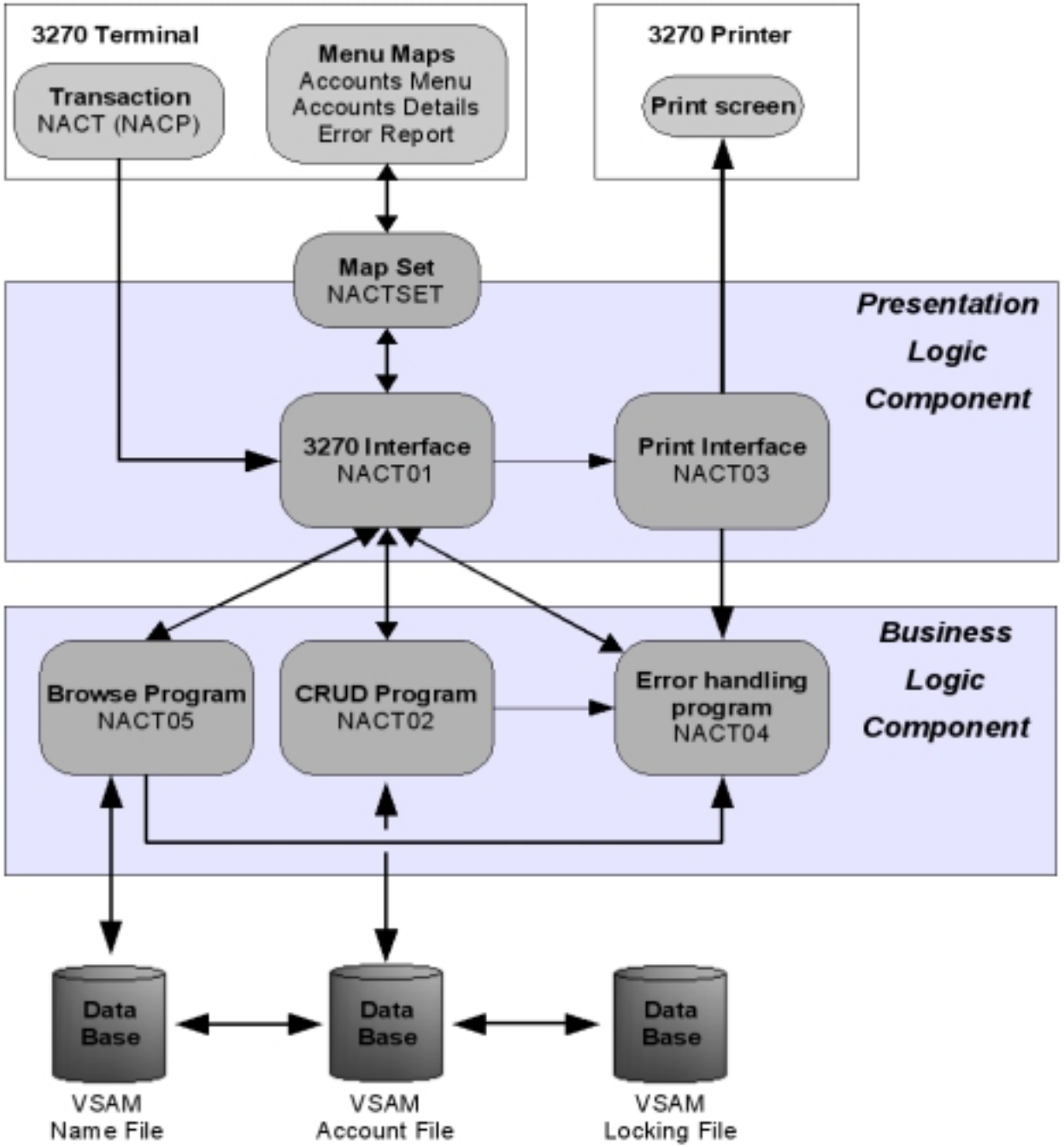
In unserem Beispiel besucht ein Neukunde, Dr. Walter Meier, das Kreditbüro des Großkaufhauses KanDoIT und beantragr eine Kreditkarte. Der Sachbearbeiter der Firma KanDoIT läßt ihn das nebenstehende Formular ausfüllen.

Der Sachbearbeiter ordnet dem Kunden die Konto Nr. 26004 zu und signiert das Antragsformular mit dem Kürzel seines Namens DEF. Als Grund für die Kreditkartenausgabe wird N (neu) angegeben.

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		

Beispiel
KanDolt Großkaufhaus

In der Kunden Kreditkartenverwaltung ist die Kundendatei als key-sequenzed VSAM Datei angelegt. Die Struktur eines Records ist nebenstehend abgebildet.

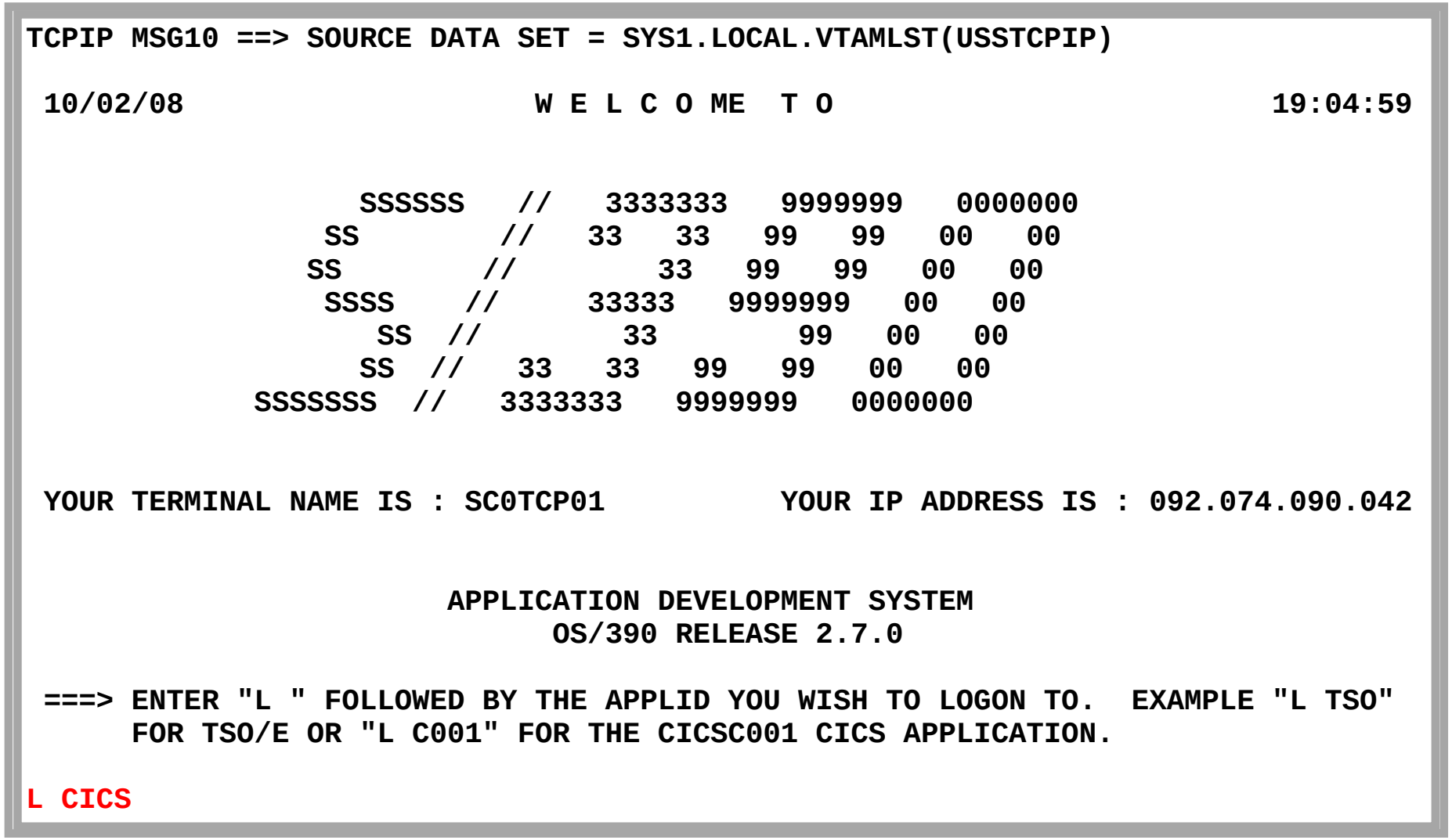


KanDoIt
NACT Transaktion

Die Struktur der NACT Transaktion ist nebenstehend abgebildet. Sie besteht aus einem Kliententeil und einem CICS Server Teil. Der Klienten Teil besteht aus einem 3270 Emulatur. Zusätzlich ist ein Arbeitsplatzdrucker vorgesehen, der in der Implementierung auf dem z/OS Rechner der Universität Leipzig fehlt.

Der Server Teil besteht aus einer Presentation Logik (in BMS implementiert), und einer Business Logik (in Cobol implementiert).

CRUD (Create, Read, Update, Delete) ist das Kernelement der Business Logic.

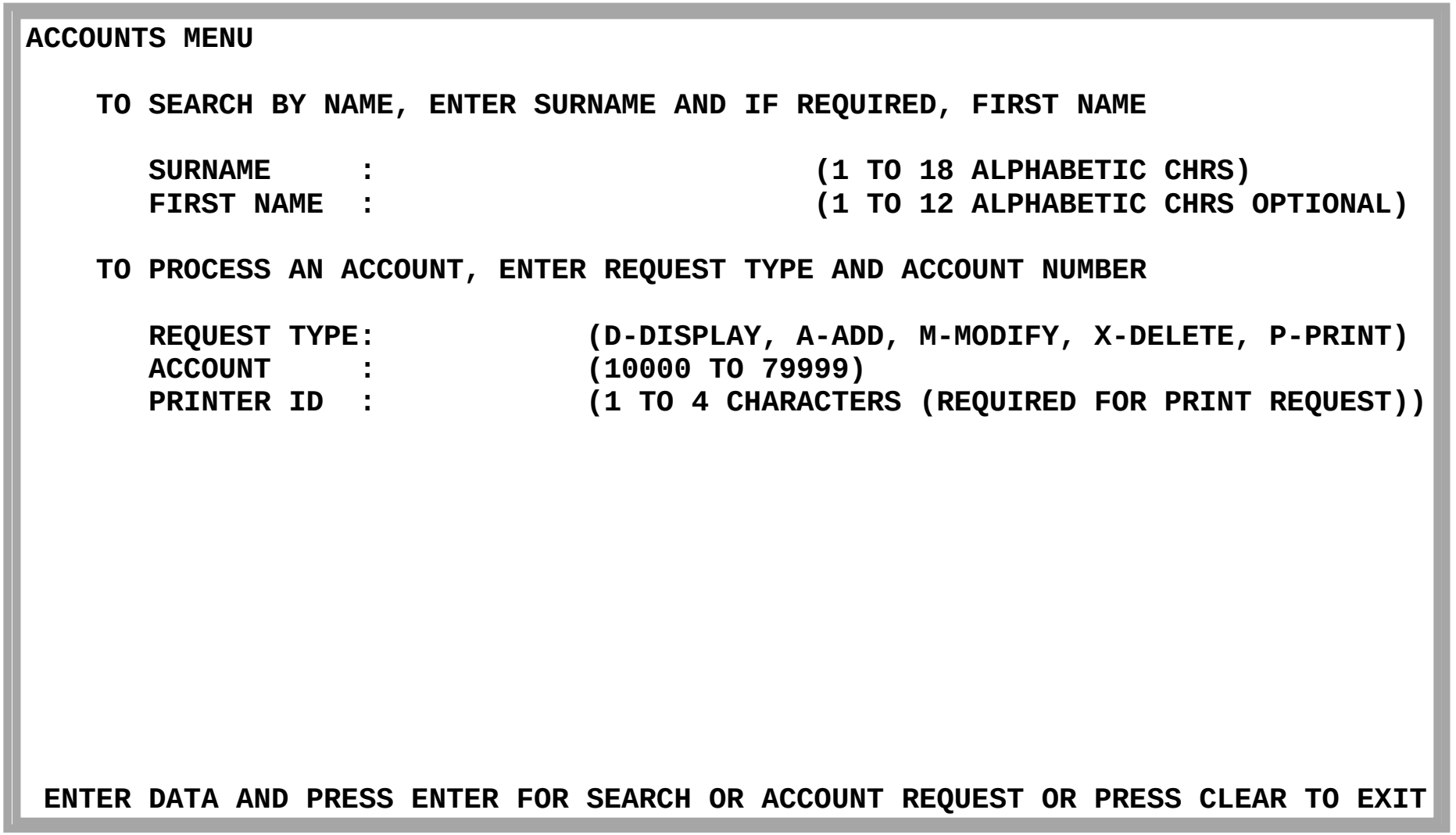


Die folgende Bildschirmfolge stellt den Ablauf einer Transaktion dar, in der Dr. Walter Meier eine neue Kreditkarte beim Großkaufhaus KanDoIT beantragt.

Auf dem Eingangsbildschirm des OS/390 Rechners wird das CICS Subsystem mit dem Kommando L(ogin) CICS aufgerufen.

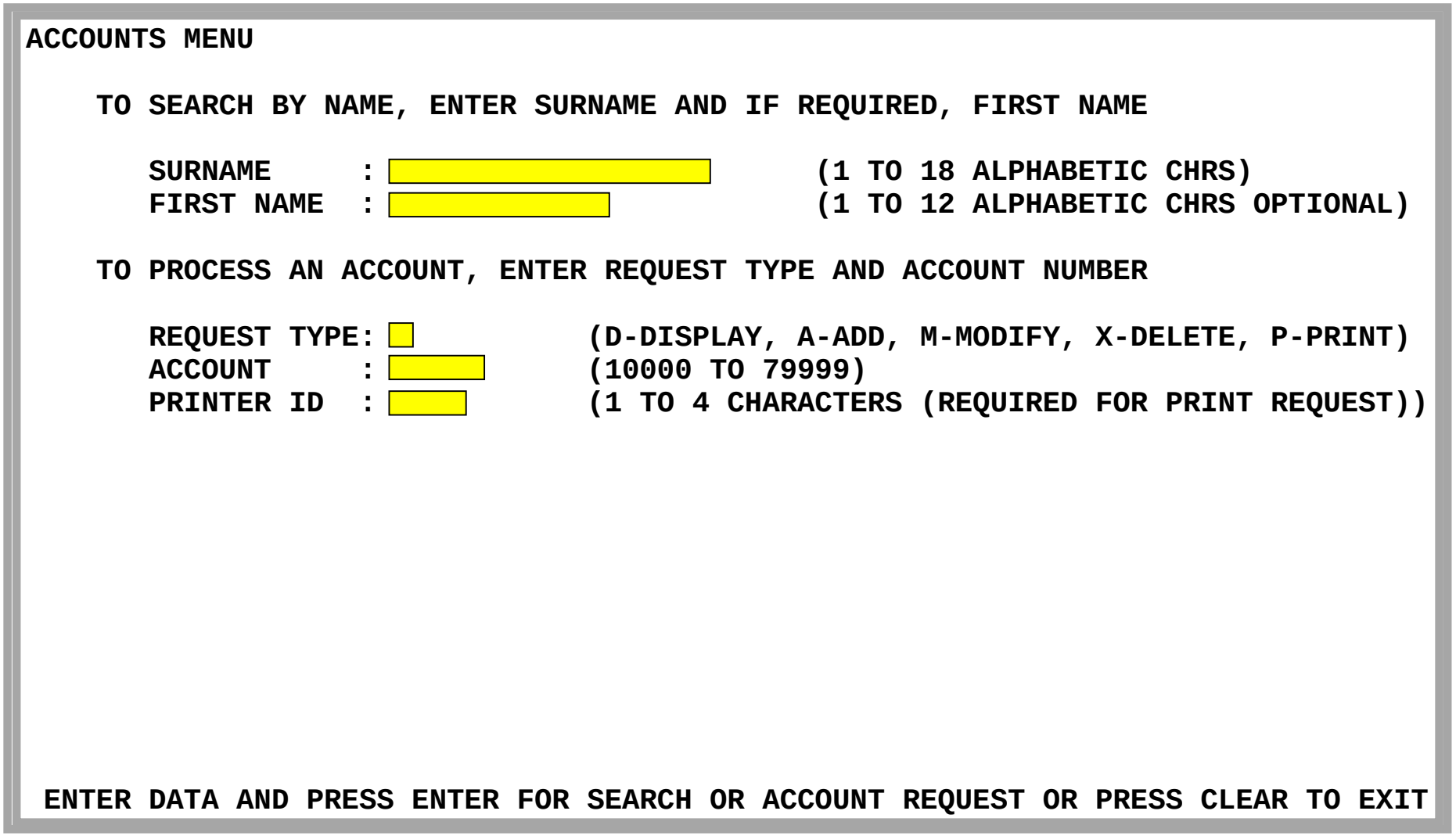
```
NACT


DFHCE3549 Sign-on is complete (Language ENU).
```

Auf eine sehr kryprische Weise fordert CICS den Benutzer auf, die TRID der Transaktion einzugeben, hier NACT.



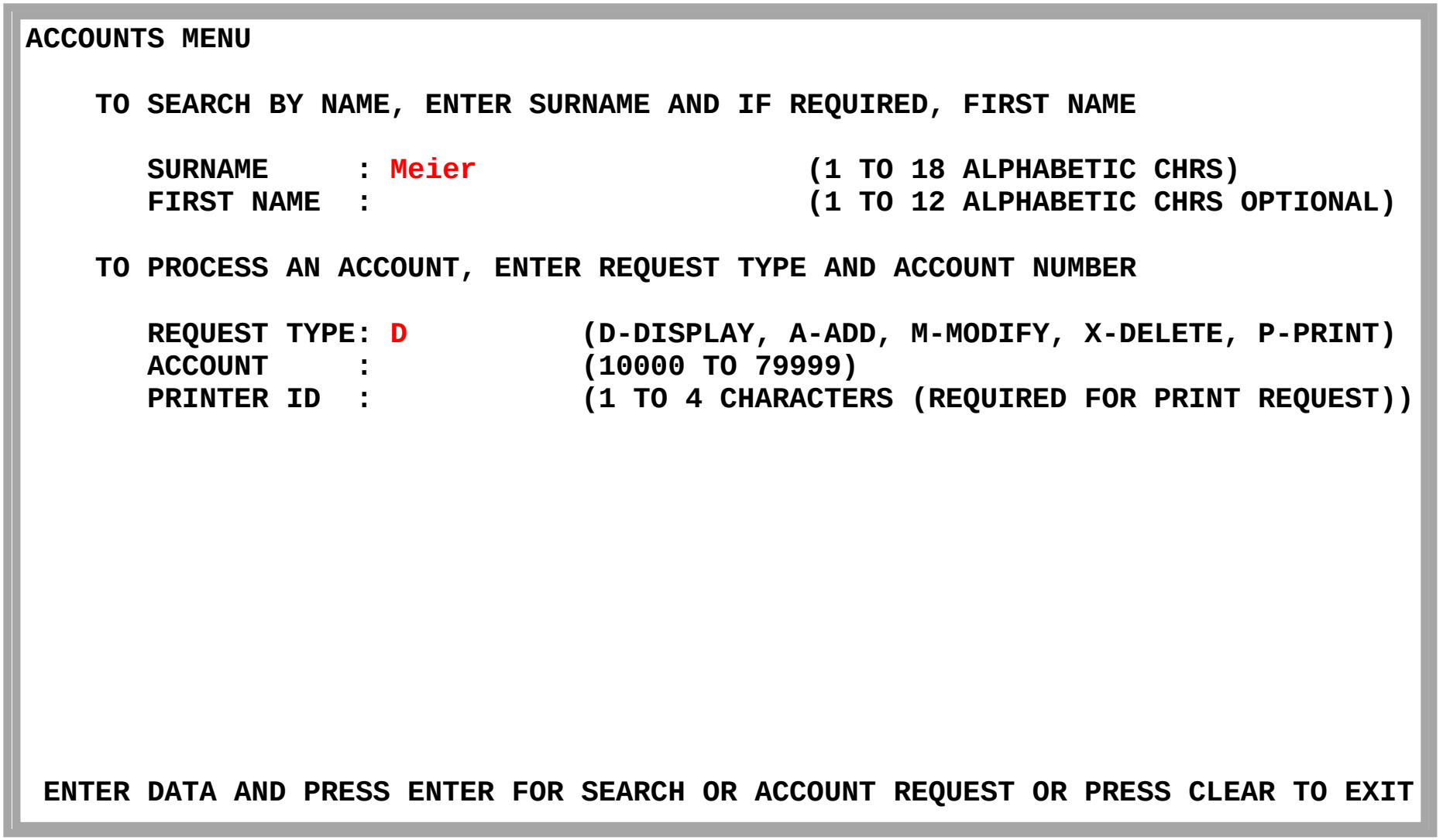
Dies ist der Eingabescreen der NACT Transaktion. Er enthält vorgefertigten Text sowie 5 Felder (hier unsichtbar), in die der Benutzer an seinem Terminal Daten eingeben kann.

Der hier dargestellte Screen wird von CICS als „MAP“ , von anderen Transaktionsmonitoren auch als View oder Screen bezeichnet. Zu einer Transaktion gehören in der Regel mehrere unterschiedliche MAPs.



Dies ist der gleiche Eingabescreen der NACT Transaktion. Zum Unterschied gegenüber der vorherigen Darstellung sind hier die möglichen (an sich unsichtbaren) 5 Eingabefelder gelb dargestellt. Die Felder haben eine Länge von jeweils 18, 12 , 1, 5 und 4 Zeichen. Vom Benutzer wird erwartet, dass er dies weis.

Bitte Zurückhaltung mit Ihren Verbesserungsvorschlägen.

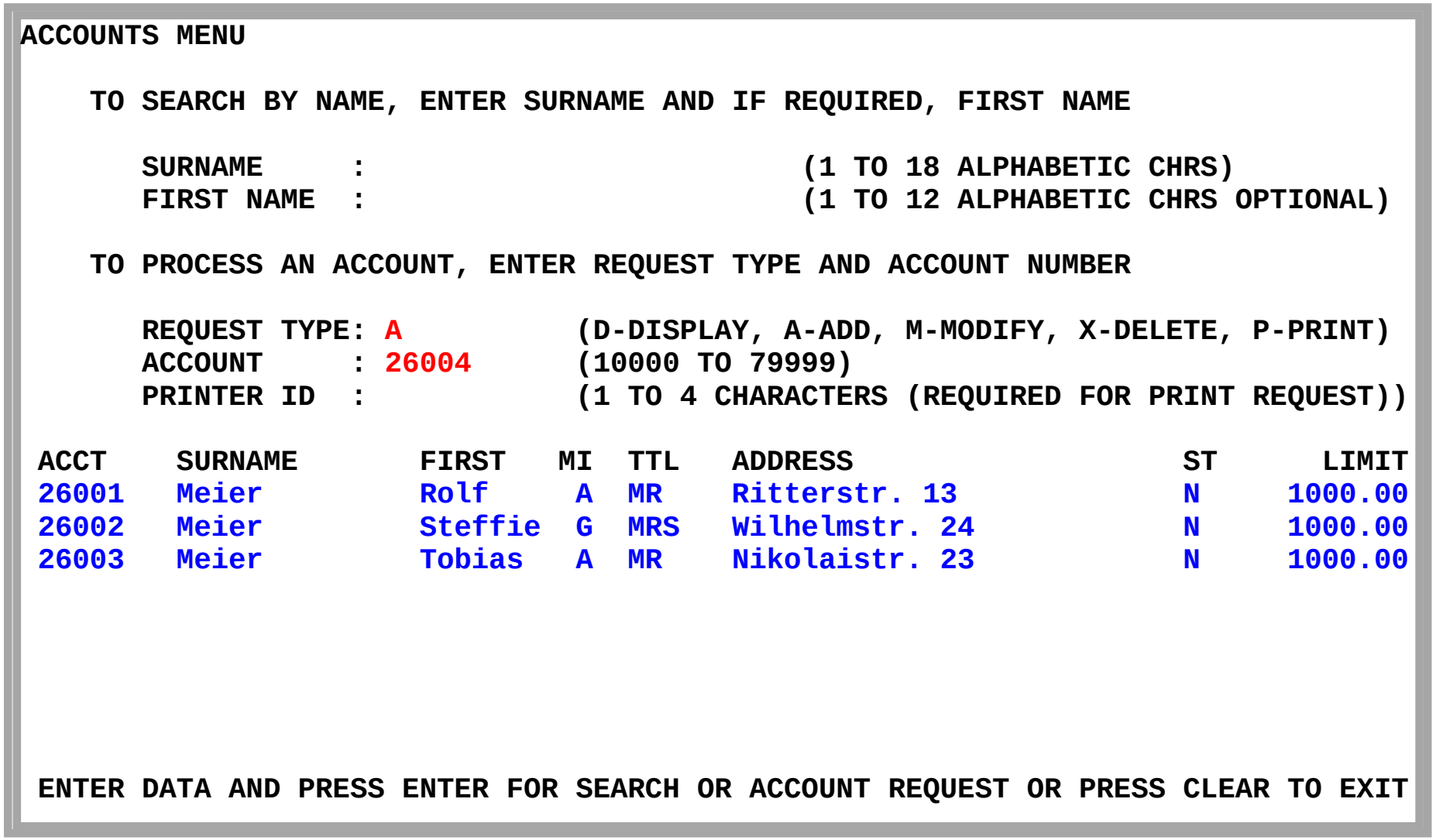


Der Sachbearbeiter gibt den Namen Meier und den Request Type D (für Display) ein. Er will überprüfen, ob der Name Walter Meier schon vorhanden ist.

In der hier gewählten Darstellung werden die Eingaben des Sachbearbeiters in rot und die Antworten von CICS in blau dargestellt.

```
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:            (D-DISPLAY, A-ADD, M-MODIFY, X-DELETE, P-PRINT)
       ACCOUNT     :            (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        Ritterstr. 13             N     1000.00
 26002   Meier         Stefan   A        Wilhelmstr. 24            N     1000.00
 26003   Meier         Tobias   A        Nikolaistr. 23            N     1000.00









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

CICS findet in seiner Kundendatei drei Einträge mit dem Namen Meier, allerdings keinen Dr. Walter Meier.



Der Sachbearbeiter gibt als Request Type A (für add) ein. Damit soll ein neuer Record für Herr Dr. Walter Meier angelegt werden.

Gleichzeitig ordnet er dem neuen Kunden die Konto Nr. 26004 zu.

```
ACCOUNTS                  ADD ACCOUNT NUMBER 26004

 SURNAME      :                    (18 CHRS) TITLE      :      (4 CHRS OPTIONAL)
 FIRST NAME   :                    (12 CHRS) MIDDLE INIT:      (1 CHR  OPTIONAL)
 TELEPHONE    :                    (10 DIGS)
 ADDRESS LINE1:                         (24 CHRS)
         LINE2:                         (24 CHRS)
         LINE3:                         (24 CHRS OPTIONAL)

 CARDS ISSUED :           (1 TO 9)          CARD CODE  :      (1 CHR)
 DATE ISSUED  :           (MM DD YY)        REASON CODE:      (N,L,S,R)
 APPROVED BY  :           (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
```

CICS stellt eine andere MAP auf den Bildschirm. In diesem Beispiel verwenden wir nur diese zwei MAPs, die meisten Transaktionen haben eine sehr viel größere Anzahl von MAPs.

Die Eingabefelder sind wiederum unsichtbar.

```
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 : 2          (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
```

Der Sachbearbeiter überträgt die Daten aus dem Antragsformular und betätigt die Entertaste. Dies bewirkt, dass die eingegebenen Daten in den neuen VSAM Record für Herrn Dr. Walter Meier übernommen werden..

```
ACCOUNTS MENU

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

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









 ACCOUNT NUMBER 26004 ADDED

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

Es erscheint wieder die erste Map. Der Sachbearbeiter ruft nochmals die D (display) Funktion für alle Meier's in der Kundendatei auf.

```
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:            (D-DISPLAY, A-ADD, M-MODIFY, X-DELETE, P-PRINT)
       ACCOUNT     :            (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
 26004   Meier         Walter   R  DR    Heilbronnerstr. 91        N     1000.00




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

CICS zeigt an, dass es nun auch einen Kunden Record für Herrn Dr. Walter Meier gibt.

Das war es.

Moderne Oberflächen

In dem hier gezeigten Beispiel benutzen alle Screens das 1971 entstandene „3270 Protokoll“ und die BMS (Basic Mapping Support) Präsentationslogik, welche Bestandteil von jedem CICS TP Monitor ist.

Eine Alternative sind moderne Benutzeroberflächen, die in der großen Mehrzahl der Fälle mit Hilfe von Java programmiert werden. Ein einfaches Beispiel ist unten wiedergegeben.

In Wirtschaft und Verwaltung wurden in den letzten Jahren für die allermeisten CICS Anwendungen hiermit ausgestattet. Dies geschah fast immer als Alternative (und nicht als Ersatz) zu der existierenden BMS Präsentationslogik, die heute immer noch sehr gebräuchlich ist.

KanDoIT Account Enquiry Client

Actions Help

Enter account number: 26004

KanDOIT!

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

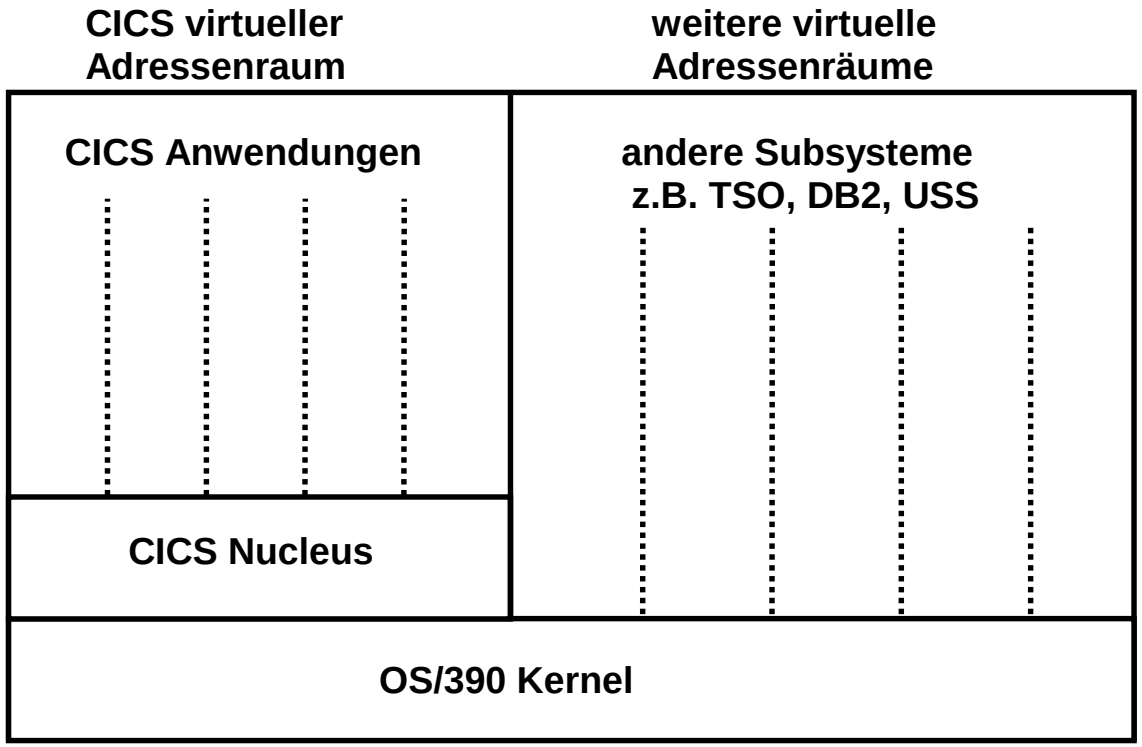
Telephone: 0000733456

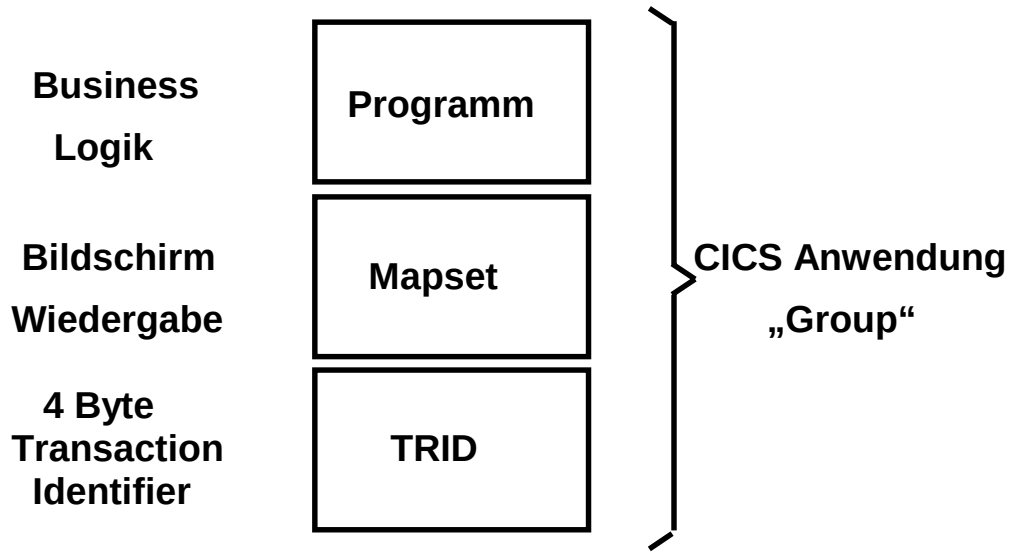
Others Who May Charge:

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





Anwendungen, die auf einem Server laufen, werden normalerweise in einer Entwicklungsumgebung erstellt, die nicht auf dem Server läuft.

Eine CICS Anwendung wird ebenfalls außerhalb von CICS entwickelt. Alle Komponenten der Anwendung werden als „Group“ zusammengefasst. Die Group erhält einen Namen, der CICS bekannt gegeben wird.

Anschließend wird die Group in der CICS Programm Bibliothek installiert. Eine einfache Anwendung besteht aus 3 Teilen: Programm, Mapset und Transaction ID.

Installation einer neuen CICS Anwendung

Alle unter CICS installierten Anwendungen existieren in der Form von Gruppen (Group). Eine Group enthält alle Komponenten, aus denen eine CICS Anwendung besteht. Als Minimum besteht eine Group aus einem ausführbaren CICS Anwendungsprogramm, einer TRID sowie falls die CICS interne Präsentationslogik Componente benutzt wird, aus einem Mapset.

Die Installation einer neuen Anwendung unter CICS besteht aus den folgenden Phasen:

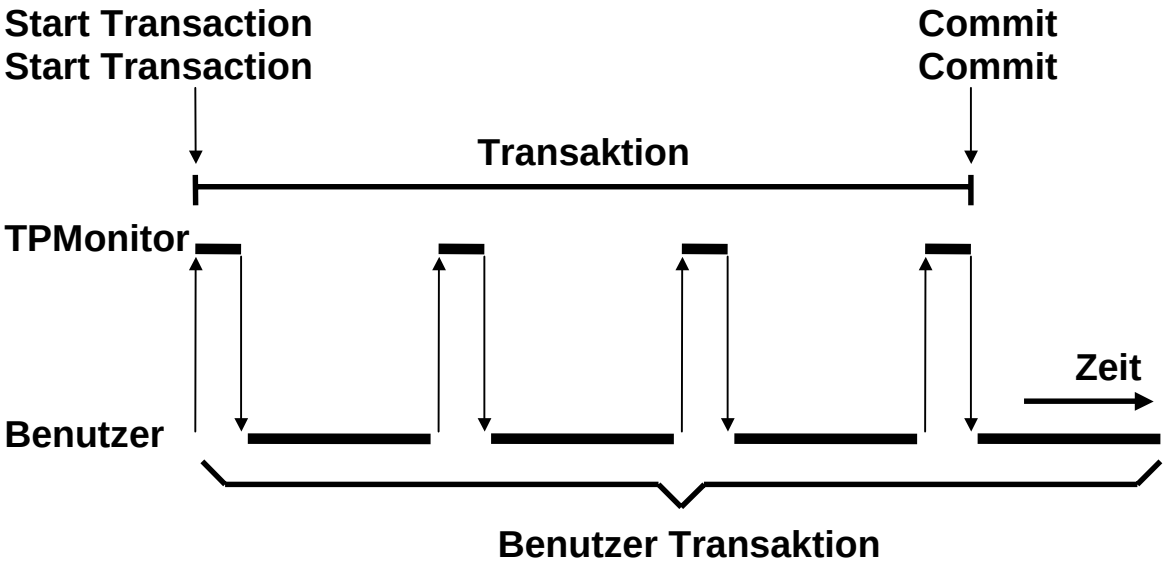
Mit Hilfe der Definition (define) wird CICS mitgeteilt, dass eine neue Group mit dem Namen xxxx besteht. Zu dieser Group gehören die folgenden Komponenten:

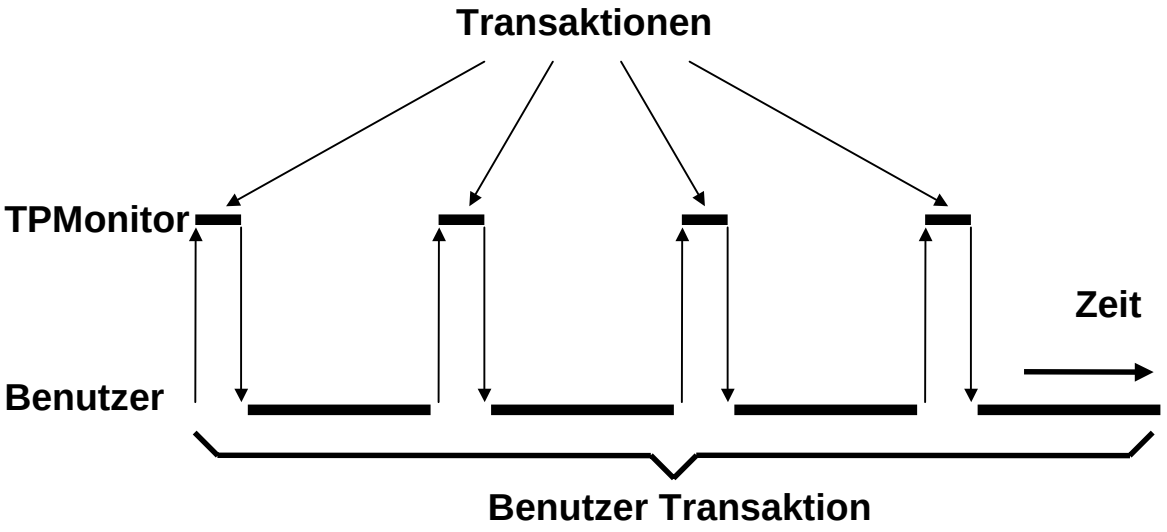
Anwendungsprogramm	**yyyy**
Mapset	**zzzz**
TRID	**aaaa**

Bei der eigentlichen Installation werden

die einzelnen Referenzen werden aufgebaut, z.B. neue TRID in TRID Table einfügen
Laden der Komponenten, z.B. laden des Anwendungsprogramms in die CICS Programmbibliothek.

Diese Schritte sind für jede neue CICS-Anwendung erforderlich. In einer nachfolgenden Übung werden sie diese Schritte vornehmen.





Conversational Transaction

Eine Conversational Transaction bindet Resourcen über einen längeren Zeitraum

Pseudo-Conversational Transaction

Pseudo-Conversational Transaktionen sind der bevorzugte Programmierstiel. Zwischen den Transaktionsschritten können Ressourcen von anderen Klienten (Transaktionen) genutzt werden. Konzept des Sessionmanagements.