8. CICS Transaktionsserver

8.1 CICS Übersicht

8.1.1 Implementierung eines Transaktionsmonitors

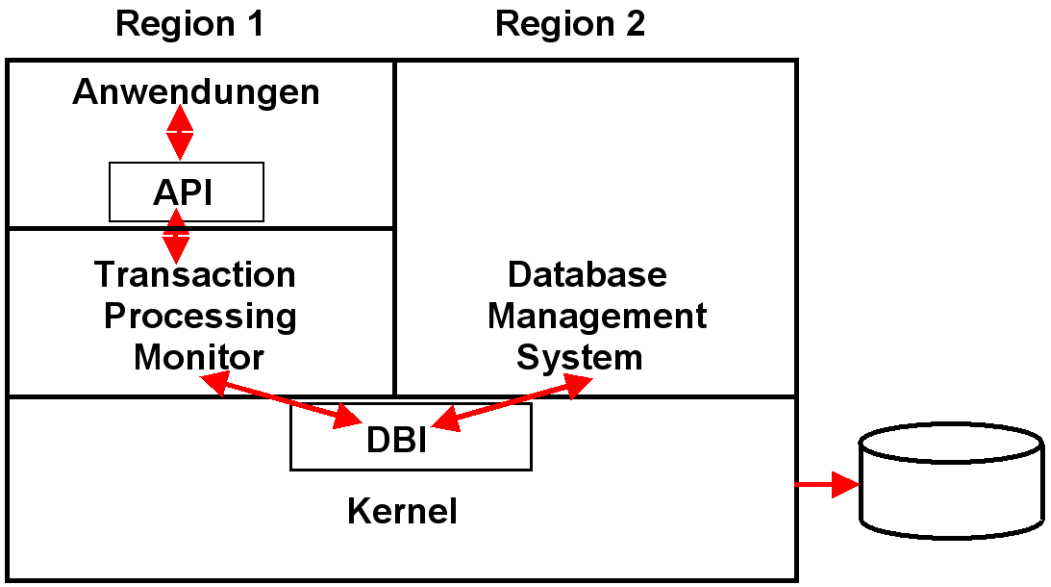


Abb. 8.1.1
Alternative 1: TP Monitor und Datenbanksystem sind getrennte Subsysteme

Unter z/OS laufen der Transaction Processing Monitor (TP Monitor, Transaction Server) und das Datenbank-System als unabhängige Anwendungsprozesse in getrennten virtuellen Adressenräumen unter dem Betriebssystem-Kernel. Der TP Monitor unterstützt sowohl Stapelverarbeitungs- als auch interaktive Transaktionen.

TP Monitor und Datenbanksystem kommunizieren über eine Kernel Schnittstelle (Datenbank Interface, DBI) miteinander.

Der Aufruf von Kernel Funktionen ist sehr aufwendig. Deshalb vermeidet der TP-Monitor nach Möglichkeit die Nutzung der Betriebssystem-Funktionen. Um Leistung und Durchsatz zu optimieren, kann er z.B. eigene Message Behandlungs- und Queuing-Einrichtungen enthalten.

8.1.2 Transaction Processing Facility

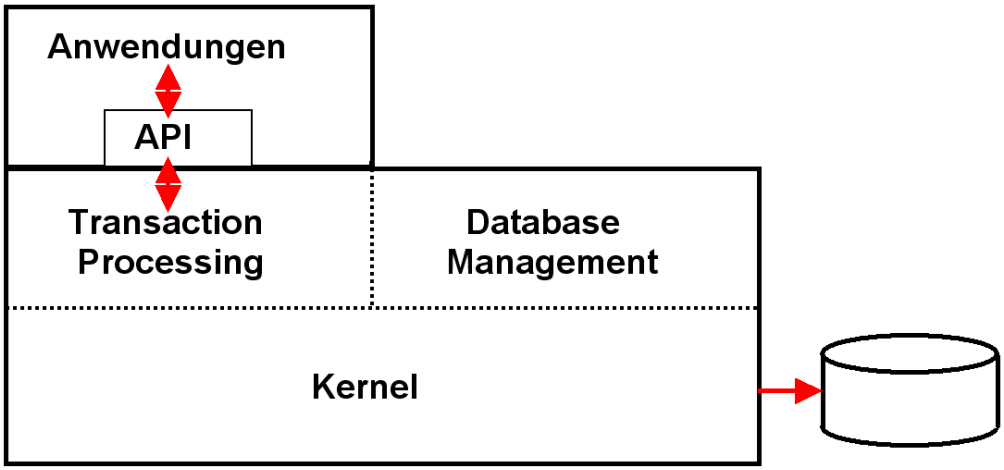


Abb. 8.1.2
Alternative 2: TP Monitor und Datenbanksystem sind Bestandteil des Kernels

Es wäre denkbar, die TP-Monitor- und Datenbank Funktionen in das Betriebssystem einzubauen. Dies ist z.B. beim Betriebssystem *Guardian* von HP/Compaq/Tandem und beim IBM-Betriebssystem TPF der Fall. TP Monitor und Datenbanksystem laufen im Kernel Status, evtl. auch die Anwendungen.

Dies bedingt sehr große Sorgfalt beim Schreiben des Codes. Durch Vermeidung des Wechsels zwischen User Status und Kernel Status (Problem and Supervisor State) ist ein signifikanter Leistungsgewinn möglich.

TPF *(Transaction Processing Facility)* ist ein eigenständiges Mainframe Betriebssystem, das nur eine einzige Anwendung erlaubt, nämlich die Transaktionsverarbeitung. Bei TPF laufen Anwendungen, Transaction-Monitor und Überwacher gemeinsam im Überwacher-Status.

Zu den Eigenschaften von TPF gehören:

- **Keine Einrichtungen für Softwareentwicklung (erfolgt auf einem anderen Rechner)**
- **Run-to-Completion (non-preemptive) Scheduler/Dispatcher**
- **Betriebssystem Aufruf erfordert ca. 500 Maschinenbefehle**
- **E/A Operation erfordert etwa 500 - 800 Maschinenbefehle**
- **>10 000 Transaktionen/s**

TPF wird eingesetzt, wenn besonders hohe Raten von relativ einfachen Transaktionen auftreten. Beispiele hiefür bilden Systeme für die Flugplatzreservierung, Geldausgabeautomaten und Kreditkartenverifizierung.

TPF ist aus dem Saber-Flugplatz-Reservierungssystem hervorgegangen, das 1959 gemeinsam von American Airlines und IBM entwickelt wurde. Später ist es in ACP und dann in TPF umbenannt worden.

Marriott Hotels, eine der großen internationalen Hotelketten, betreibt ein heterogenes Rechenzentrum. Es enthält ein zEnterprise 196 und ein System z10 Mainframe in dem primären Rechenzentrum und ein System z10 Mainframe-Backup-System in der remote Disaster Recovery Site. Darüber hinaus enthält die Marriott IT-Infrastruktur eine Mischung aus kleinen und Midrange-Systemen, die das gesamte Spektrum der Windows-, Linux-und UNIX-Betriebssysteme beinhalten. "Wir betreiben auch virtualisierte Ressourcen und evaluieren derzeitig die Installation einer zBX zEnterprise BladeCenter Extension".

"Unser zEnterprise System verwendet ein TPF-Betriebssystem, das wir im Laufe der Jahre für unsere Reservierungs-Verarbeitung optimiert haben. Wir glauben, dass dies uns einen strategischen Wettbewerbsvorteil" sichert“.

Mainframe Executive, Sept/Oct 2011.

8.1.3 Customer Information Control System (CICS)

CICS (ausgesprochen kiks, manchmal auch ziks) ist der am weitesten verbreitete, IBM proprietäre Transaktionsmonitor der Firma IBM. Die offizielle Bezeichnung ist:
CICS Transaction Server.

CICS ist unter den System z-Betriebssystemen z/OS und z/VSE, sowie in modifizierter Form (als Encina Erweiterung) unter AIX, HP-UX, Solaris sowie Windows Server verfügbar.

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.

Seit einigen Jahren benutzt IBM die offizielle Bezeichnung CICS Transaction Server oder abgekürzt CICS TS. In der Umgangssprache benutzt man weiterhin den Begriff CICS TP Monitor.

Derzeitig (2013) ist die Version CICS TS 5.1 verfügbar. Weit verbreitet ist noch Version CICS TS 3.2. Auf unserem Universitätsrechner ist beides installiert.

„TX Series for Multiplatforms“ ist eine CICS Implementierung für Unix, Linux und Windows, und ist weitestgehend mit der z/OS CICS Version kompatibel.

http://www.cedix.de/Literature/Textbooks/CicsIntro.pdf

Eine ganze Reihe von CICS Anwendungsprogrammen sind vor Jahrzehnten geschrieben worden und entsprechen in ihrer Struktur nicht mehr modernen Software Engineering Anforderungen.

Es hat immer wieder Überlegungen gegeben, diese in ihrer Struktur veralteten Programme umzuschreiben. Dies geschieht in der Praxis so gut wie gar nicht.

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

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

Frage:

Schätzen Sie bitte, wie groß die weltweiten Investitionen aller derzeitig in Betrieb befindlichen CICS Anwendungen sind

- **10 000 Mannjahre ?**
- **10 Millionen Mannjahre ?**
- **10 Milliarden Mannjahre ?**

Antwort:

10 Millionen Mannjahre dürften etwa richtig sein. Angenommen Entwicklungskosten von 100 000 $/Mannjahr ergibt eine Investition von 1 Billion $ in für Benutzer geschriebener CICS Anwendungssoftware.

- **20 000 System z Servers haben durchschnittlich 1 Mill. Zeilen aktiven CICS Anwendungscode (zwischen 200 000 und 50 Millionen pro Server), kumulativ 20 Milliarden Lines of Code (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 $ (10^{12} $) in CICS Anwendungssoftware. Zum Vergleich, das USA 1999 GNP war 9 Billion $.**

10 Milliarden Mannjahre können nicht richtig sein. Bei 100 000 $ / Mannjahr wären das etwa 1000 Billionen $. So viel Geld gibt es auf dem Planeten nicht.

There are approximately 950 000 CICS programmers worldwide.

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

http://www.cedix.de/Literature/Textbooks/CicsIntro02.pdf

http://www.cedix.de/VorlesMirror/Band1/Unicom.pdf

8.1.4 Multiprogrammierte Verarbeitung von Transaktionen

Ein Hochleistungs-Transaktionssystem verarbeitet in jedem Augenblick Hunderte oder Tausende von Transaktionen gleichzeitig und parallel. Pro CPU ist typischerweise nur eine Transaktion laufend, die anderen sind ausführbar oder warten auf den Abschluss einer Ein-/Ausgabeoperation. Hunderte oder Tausende paralleler Transaktionen sind denkbar.

Im Regelfall setzt der TP-Monitor als ein getrennter Anwendungsprozess auf dem Betriebssystem auf.

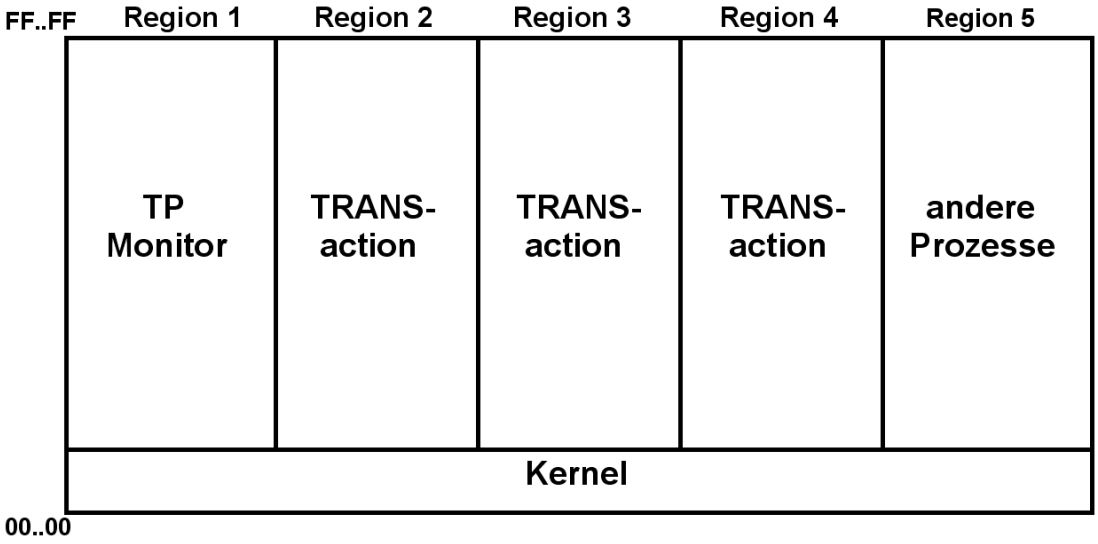


Abb. 8.1.3
Prozess-Ansatz

Für die Transaktions-Verarbeitung gibt es zwei Alternativen:

Beim Prozess-Ansatz läuft jede Transaktion als selbständiger Prozess in einem eigenen virtuellen Adressenraum (z/OS Region). Vorteil: hervorragende Isolation.

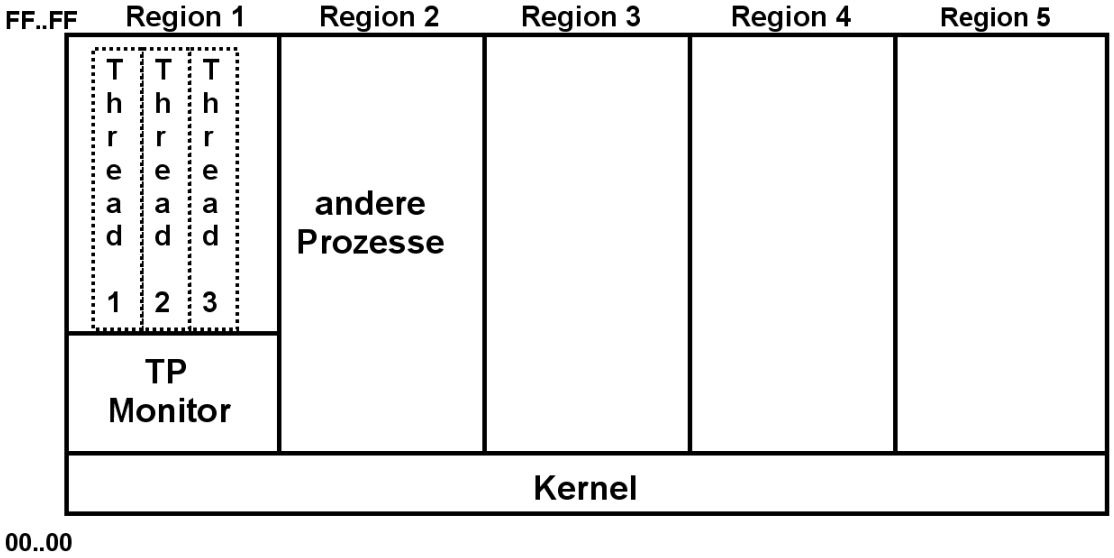


Abb. 8.1.4
Thread-Ansatz

Beim Thread-Ansatz laufen alle Transaktionen als Threads gemeinsam mit dem TP Monitor in einem einzigen virtuellen Adressenraum. Vorteil: Leistungsverhalten.

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

Die z/OS Version von CICS benutzt einen Tread-ähnlichen Ansatz. Der Transactions-Monitor und alle CICS Anwendungen laufen im gleichen virtuellen Adressenraum (CICS Region). CICS Struktur-Eigenschaften gewährleisen die Isolation der Threads untereinander und gegenüber dem TP Monitor.

Operationen über Adressraumgrenzen hinweg benötigen viele Mikrosekunden. Operationen im gleichen Adressraum benötigen Mikrosekunden-Bruchteile. Ein Faktor 100 Geschwindigkeitsunterschied ist möglich. Deshalb vermeidet der CICS TP-Monitor nach Möglichkeit die Nutzung der Betriebssystem-Funktionen.

Dazu läuft der vollständige CICS Transaktionsmonitor als eine z/OS Anwendung im Problemstatus. Gleichzeitig stellt CICS eine Laufzeitumgebung für zahlreiche Anwendungen (hier Transaktionen) zur Verfügung. Eine derzeitige Laufzeitumgebung wird als „Middleware“ bezeichnet. Andere Middleware Beispiele sind:

- **Web Application Server**
- **Message Based Queuing**
- **Corba**
- **RMI**
- **Remote Procedure Call Middleware, z.B. DCE**

8.1.5 Rolle der Exec CICS Kommandos

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

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

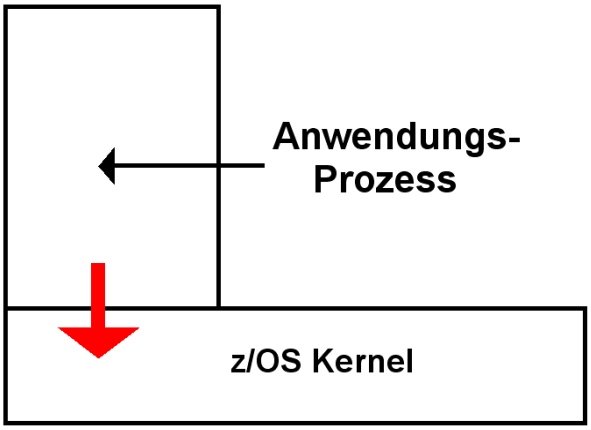


Abb. 8.1.5
System Calls

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

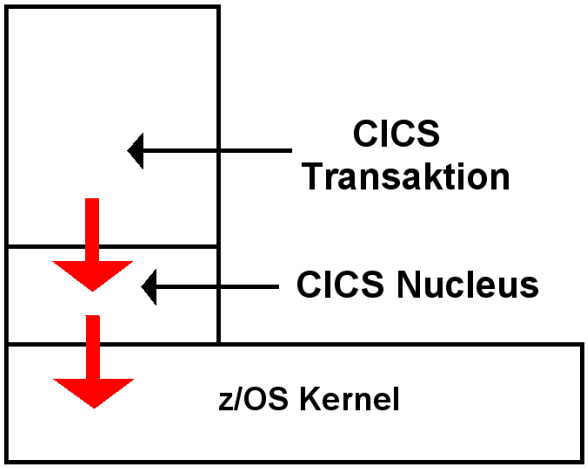


Abb. 8.1.6
EXEC CICS Kommandos

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.

Alle laufenden CICS Anwendungsprogramme befinden sich zusammen mit dem CICS Nucleus in einem einzigen virtuellen Adressenraum.

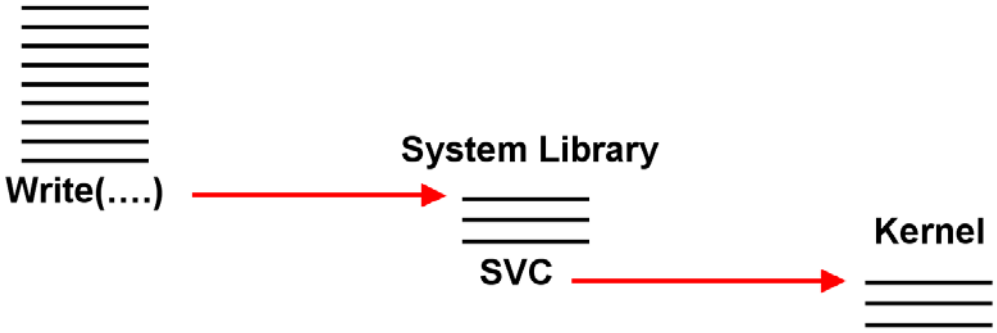


Abb. 8.1.7
Normale Schreiboperation

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

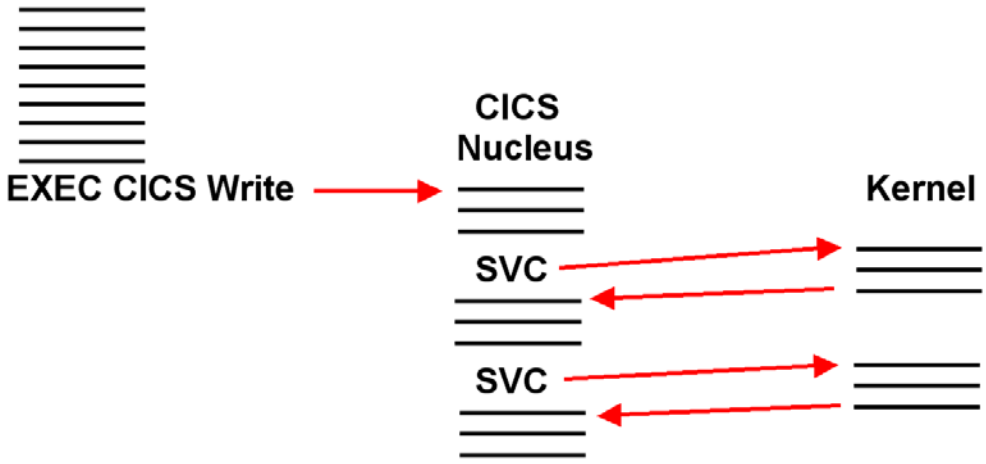


Abb. 8.1.8
CICS Schreiboperation

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, Voraussetzungen für ein evtl. Rollback geschaffen werden, usw.

Anders als reguläre Programme führen CICS Anwendungsprogramme bei transaktionskritischen Aktivitäten (Einhaltung der ACID Bedingungen) keine direkten Aufrufe des Betriebssystems aus. Stattdessen 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 unterhalb des tatsächlichen Betriebssystems um eine Umgebung für die Ausführung von Anwendungsprogrammen bereitzustellen.

Aufrufe des CICS Nucleus fangen immer mit der Sequenz EXEC CICS an, und hören mit dem Statement Delimiter der 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;
```

Ein Bildschirminhalt (eine Map) mit dem Namen label04, welche zu einer Gruppe ähnlicher Maps (einem Mapset mit dem Namens04set) gehört, wird an einen Klientenrechner gesendet.

Beispiel für einen Aufruf des CICS Nucleus 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

8.1.6 Überblick über die CICS Befehle

Bildschirm

SEND MAP
SEND CONTROL
SEND
RECEIVE MAP

Programmsteuerung

LINK
RETURN
XCTL
LOAD
RELEASE

Zwischenspeichern

WRITEQTS (Temporary Storage)
READQTS
DELETEQTS
WRITEQTD (Transient Data)
READQTD
DELETEQTD

Zeitsteuerung

ASKTIME
FORMATTIME
START
RETRieVE

Systemsteuerung

ADDRESS
ASSIGN

VSAMf

READ
WRITE
UNLOCK
DELETE
STARTBR
READNEXT
READPREV
RESETBR
ENDBR

Hauptspeichersteuerung

GETMAIN
FREEMAIN

Andere

END DEQ
SUSPEND
WRITEJOURNALNUM
HANDLE CONDITION
IGNORE CONDITION

Abb. 8.1.9
Überblick über die CICS Befehle

8.1.7 Beispiel für Embedded SQL

```
EXEC SQL INCLUDE SQLCA;
 EXEC SQL BEGIN DECLARE SECTION;
 char vname[20];
 char nname[20];
 EXEC SQL END DECLARE SECTION;

 main()
 {
   EXEC SQL DECLARE C1 CURSOR FOR
    SELECT VNAME,NNAME FROM PRAKT20.TAB020;
   EXEC SQL OPEN C1;
   EXEC SQL FETCH C1 INTO :vname, :nname;
   memcpy(map5020.map5020i.vnam1i,vname,20);
   memcpy(map5020.map5020i.nnam1i,nname,20);
   EXEC SQL FETCH C1 INTO :vname, :nname;
   memcpy(map5020.map5020i.vnam2i,vname,20);
   memcpy(map5020.map5020i.nnam2i,nname,20);
   EXEC SQL FETCH C1 INTO :vname, :nname;
   memcpy(map5020.map5020i.vnam3i,vname,20);
   memcpy(map5020.map5020i.nnam3i,nname,20);
   EXEC SQL FETCH C1 INTO :vname, :nname;
   memcpy(map5020.map5020i.vnam4i,vname,20);
   memcpy(map5020.map5020i.nnam4i,nname,20);
   EXEC SQL CLOSE C1;

   EXEC CICS SEND MAP("map5020") MAPSET("set5020") ERASE;

 }
```

Abb. 8.1.10
Code Beispiel in C++

Der hier wiedergegebene Code in C++ enthält eine Reihe von EXEC SQL Statements für Zugriffe auf eine Datenbank sowie ein einziges EXEC CICS Statement, mit dem eine Nachricht an einen Terminal gesendet wird.

Das EXEC CICS SEND MAP Statement sendet die mittels der SQL Statements aus der Datenbank ausgelesenen Daten an den Bildschirm.

8.1.8 Erstellen einer CICS - DB2 Anwendung

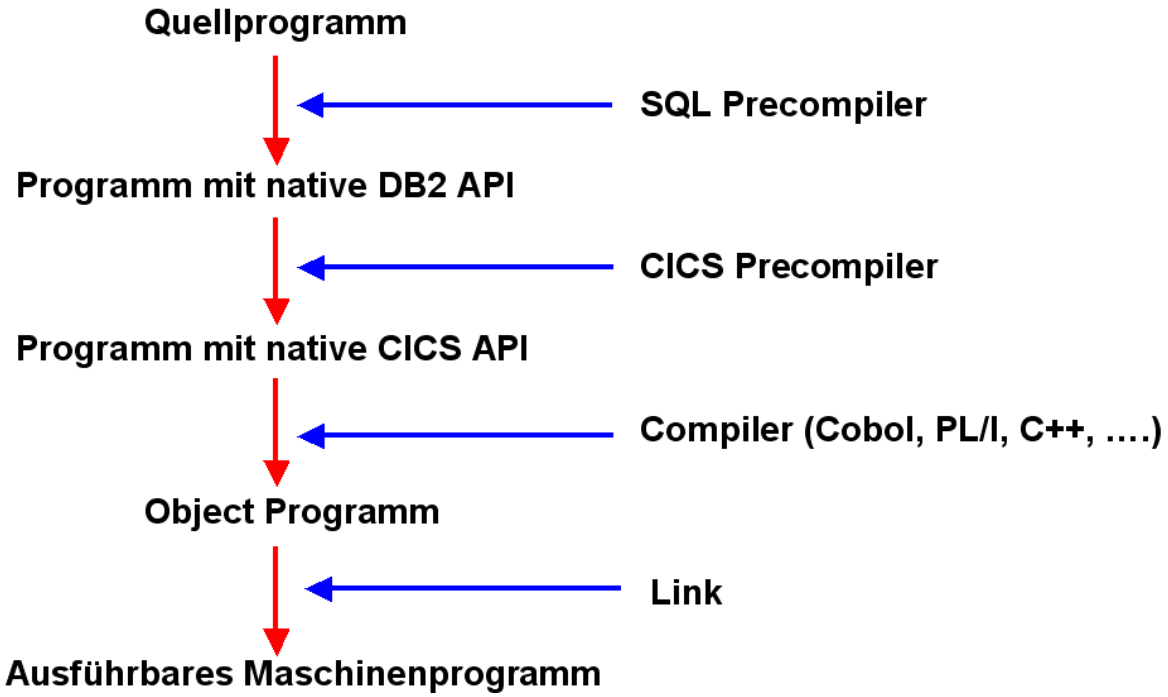


Abb. 8.1.11
Übersetzung des Quellprograms

Das in Abb. 8.1.10 dargestellte Quellprogramm wird vor der Übersetzung zunächst von einem SQL Precompiler verarbeitet, welcher die EXEC SQL Statements durch Quellcode Makros oder Bibliotheksroutine-Aufrufe ersetzt, die der angesprochenen Datenbank entsprechen. Der Code in diesen Makros würden z.B. bei einer Oracle Datenbank anders aussehen als bei einer DB2 Datenbank.

Anschließend bewirkt der CICS Precompiler eine ähnliche Vorgehensweise für alle EXEC CICS Statements.

Danach wird der so entstandene Quellcode durch einen regulären Compiler übersetzt.

8.2 Ausführungsbeispiel einer CICS Transaktion

8.2.1 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 wurden. Die Kundendatei ist als VSAM Datei implementiert, 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. (VSAM Index) identifiziert, die dem Kunden von Hand zugeordnet wurde. In der Praxis würde sie nicht von Hand sondern automatisch erzeugt werden; diese Komponente fehlt, um die Anwendung einfach zu halten.

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

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

enthalten, welches leider vergriffen, aber als Kindle Ausgabe bei Amazon erhältlich 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.cedix.de/Literature/Textbooks/Cobol/CicsCobPrimer.pdf

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.

http://www.cedix.de/DiplArb/Busse04.pdf

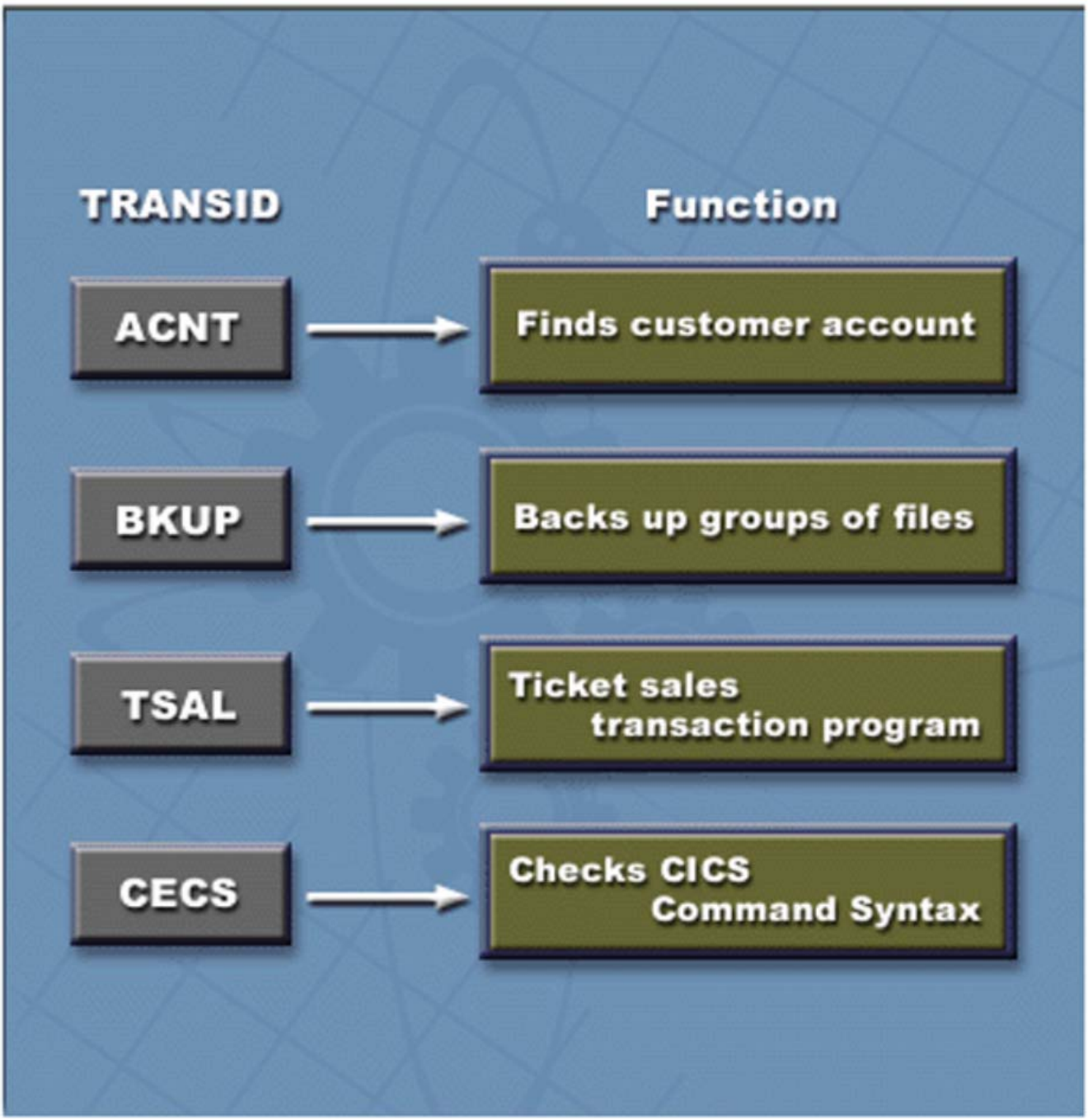
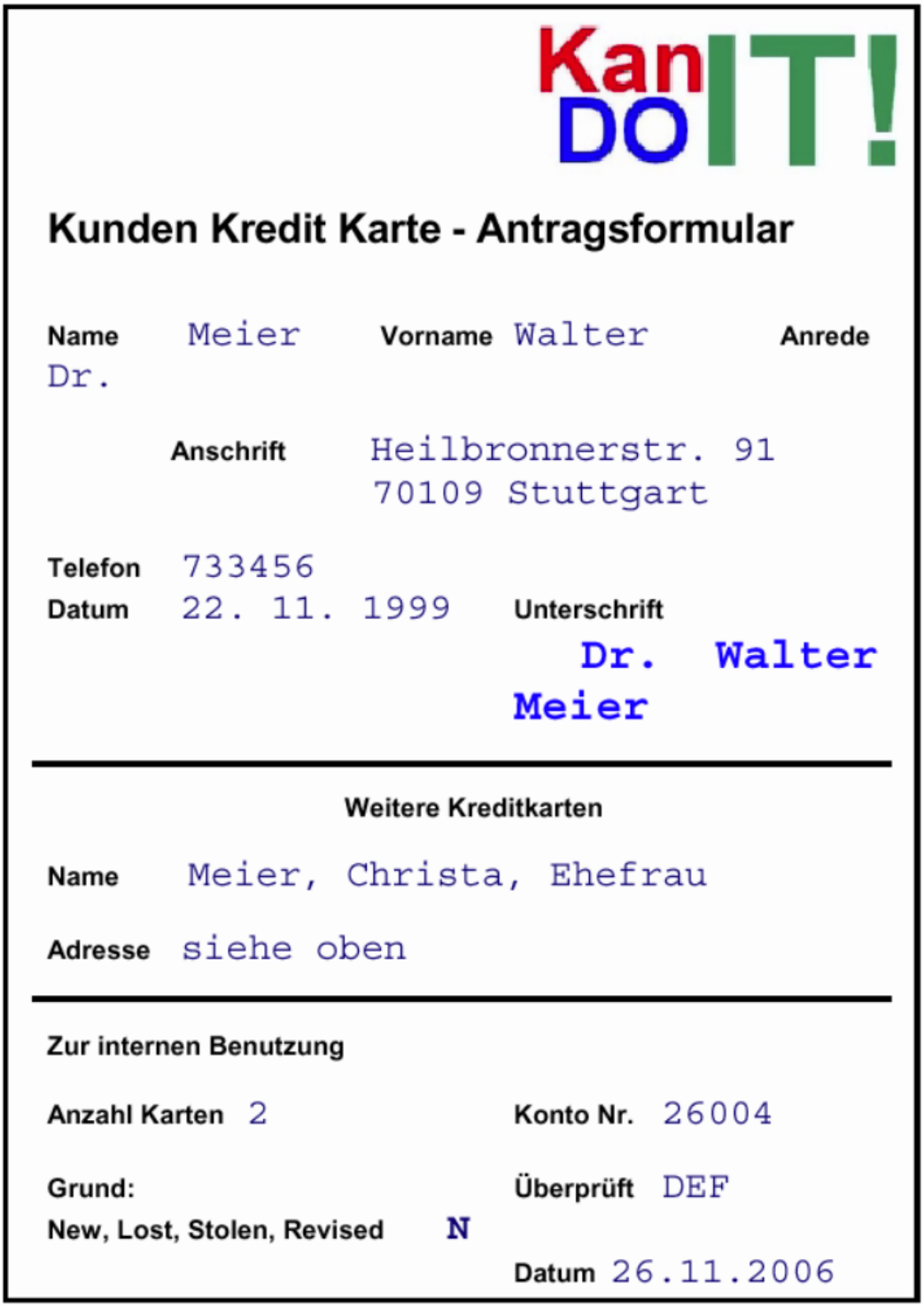


Abb. 8.2.1
Ein bestimmtes Transaktionsprogramm wir mittels einer TRID aufgerufen

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

CICS TRIDs sind grundsätzlich immer 4 Zeichen lang.

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



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

Abb. 8.2.2
Antragsformular für eine Kreditkarte

In unserem Beispiel besucht ein Neukunde, Dr. Walter Meier, das Kreditbüro der Firma KanDoIT und beantragt eine Kreditkarte. Der Sachbearbeiter der Firma KanDoIT lässt ihn das in Abb. 8.2.2 dargestellte 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		

Abb. 8.2.3
Kundenrecord des VSAM Datasets

In der Kunden Kreditkartenverwaltung ist die Kundendatei als Key-sequenced VSAM Datei angelegt. Die Struktur eines Records ist in Abb. 8.2.3 abgebildet.

8.2.2 Komponenten der NACT Transaktion

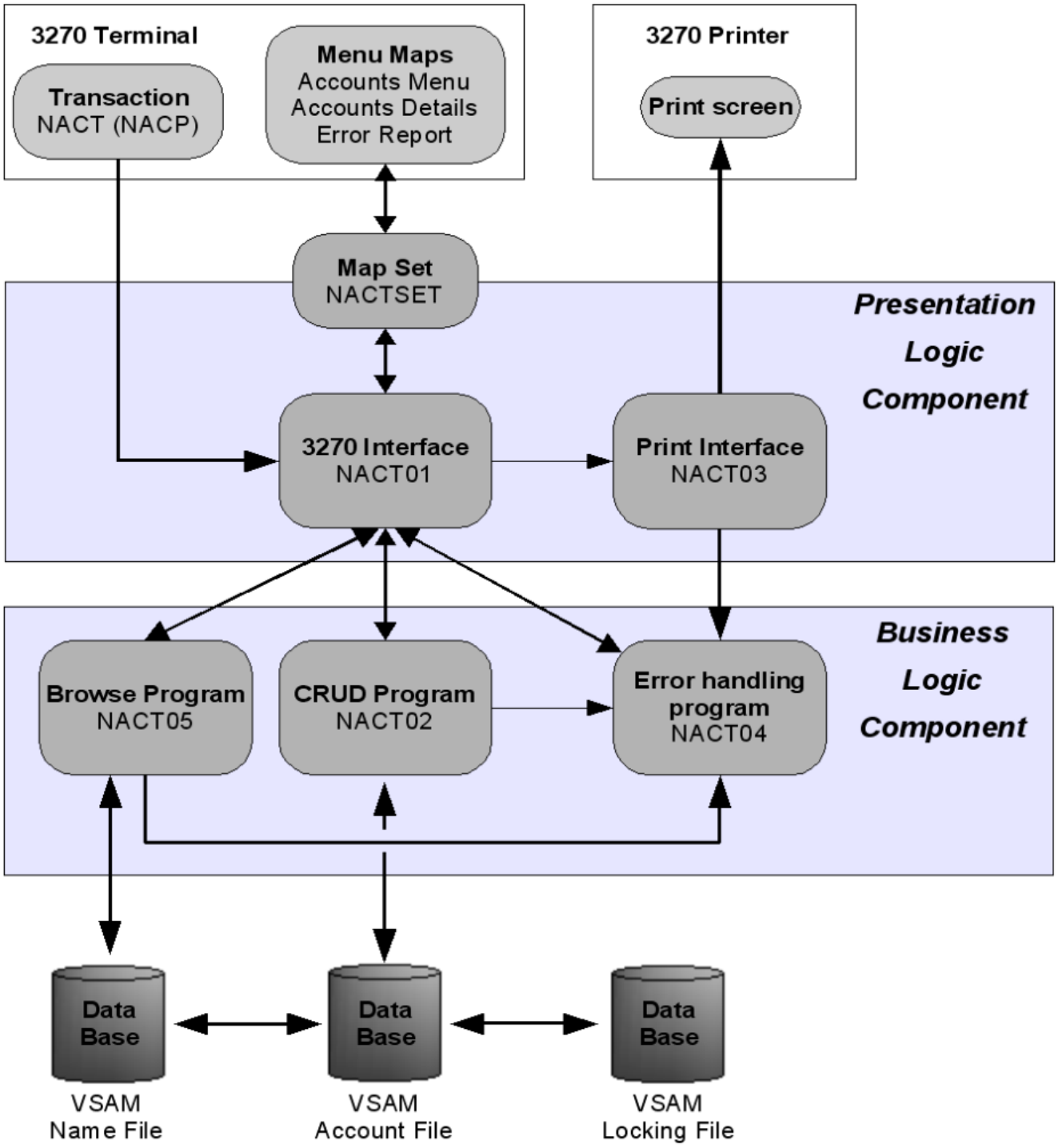


Abb. 8.2.4
Komponenten der NACT Transaktion

Die Struktur der NACT Transaktion besteht aus einem Kliententeil und einem CICS Server Teil. Der Klienten Teil besteht aus einem 3270 Emulator. 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.

8.2.3 Ausführungsbeispiel

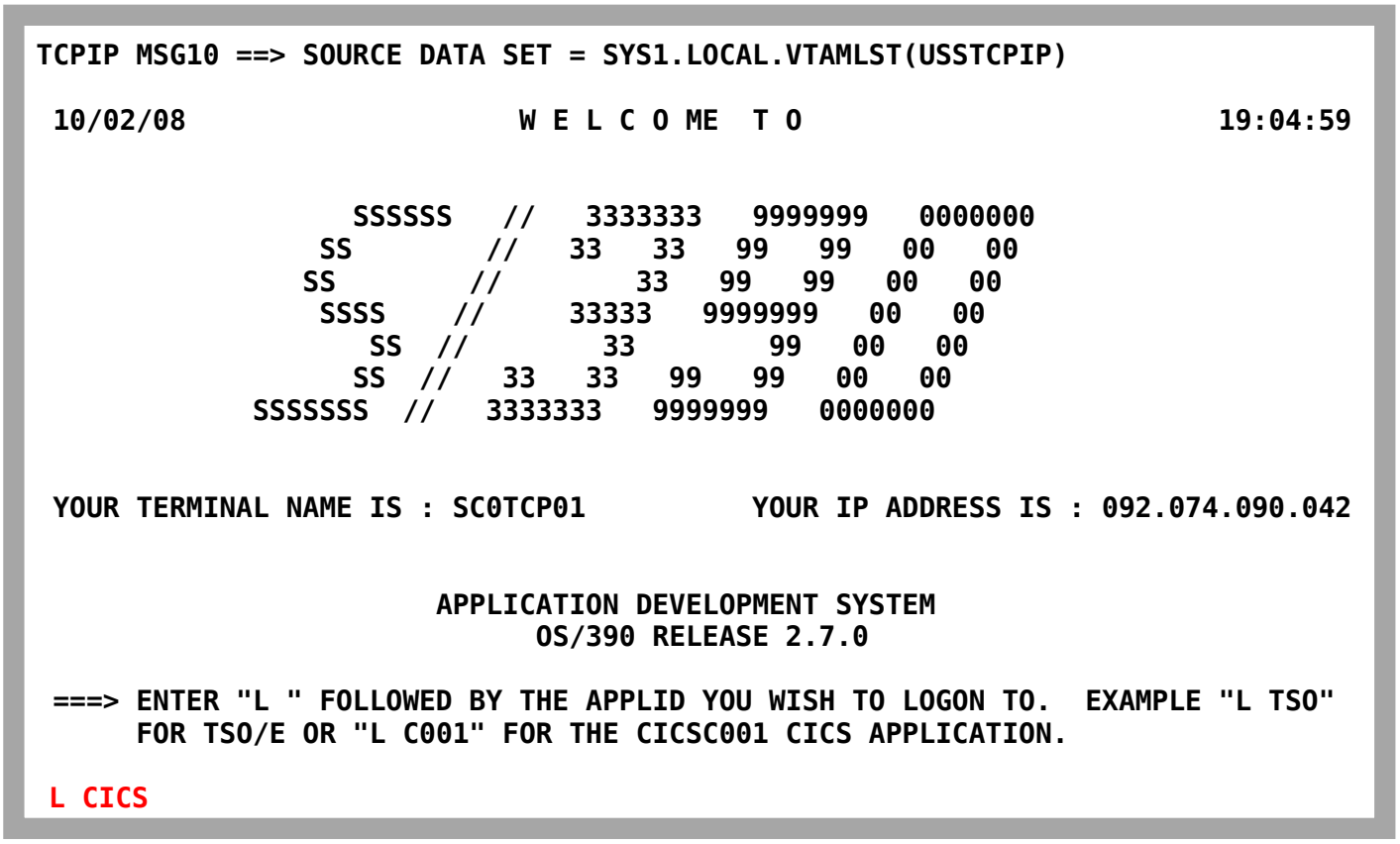


Abb. 8.2.5
Welcome Screen

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 z/OS Rechners wird das CICS Subsystem mit dem Kommando L(ogon) CICS aufgerufen. An Stelle von CICS könnten hier auch andere Subsysteme aufgerufen werden, z.B. IMS oder TSO.

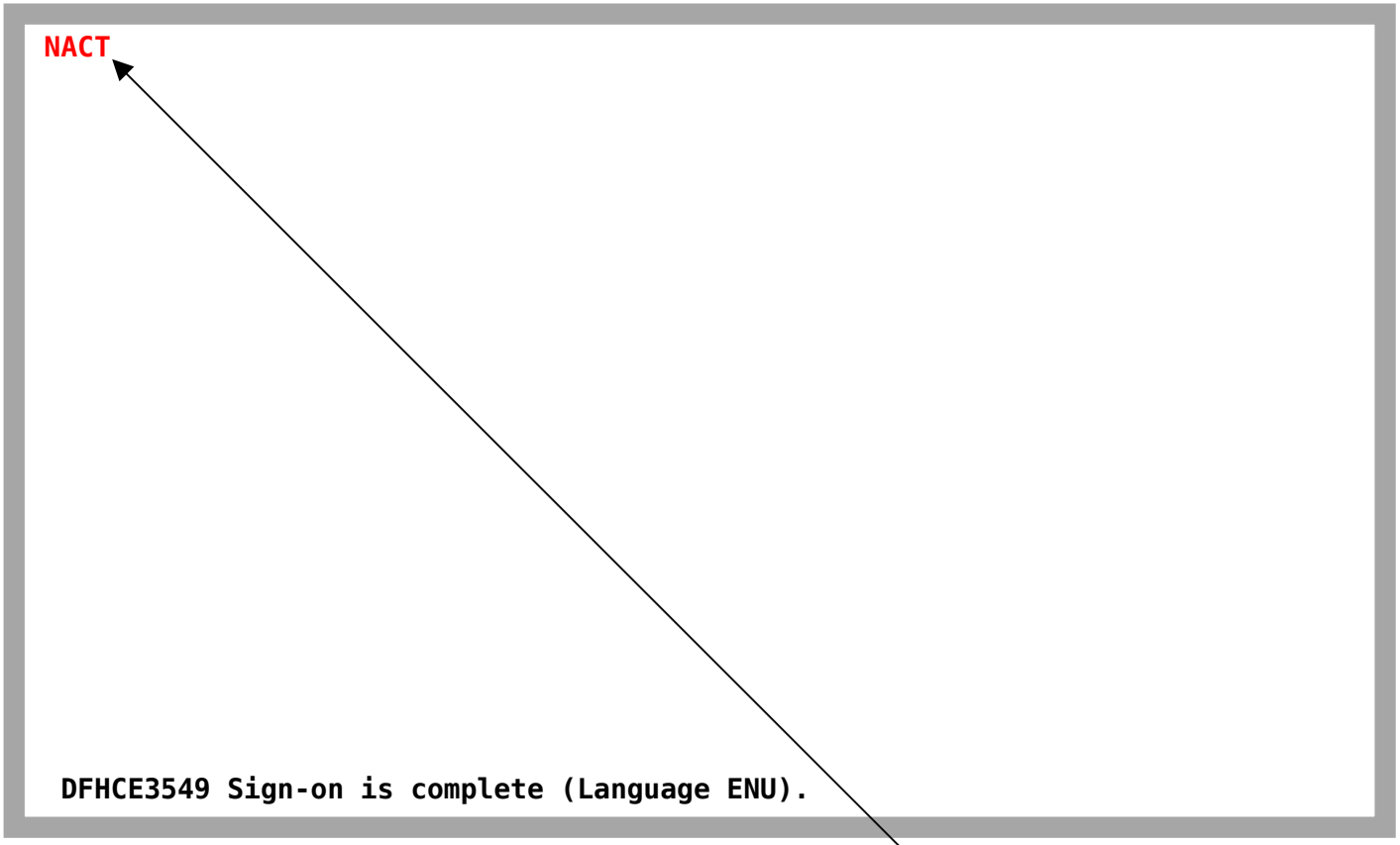


Abb. 8.2.6
TRID Selektion

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

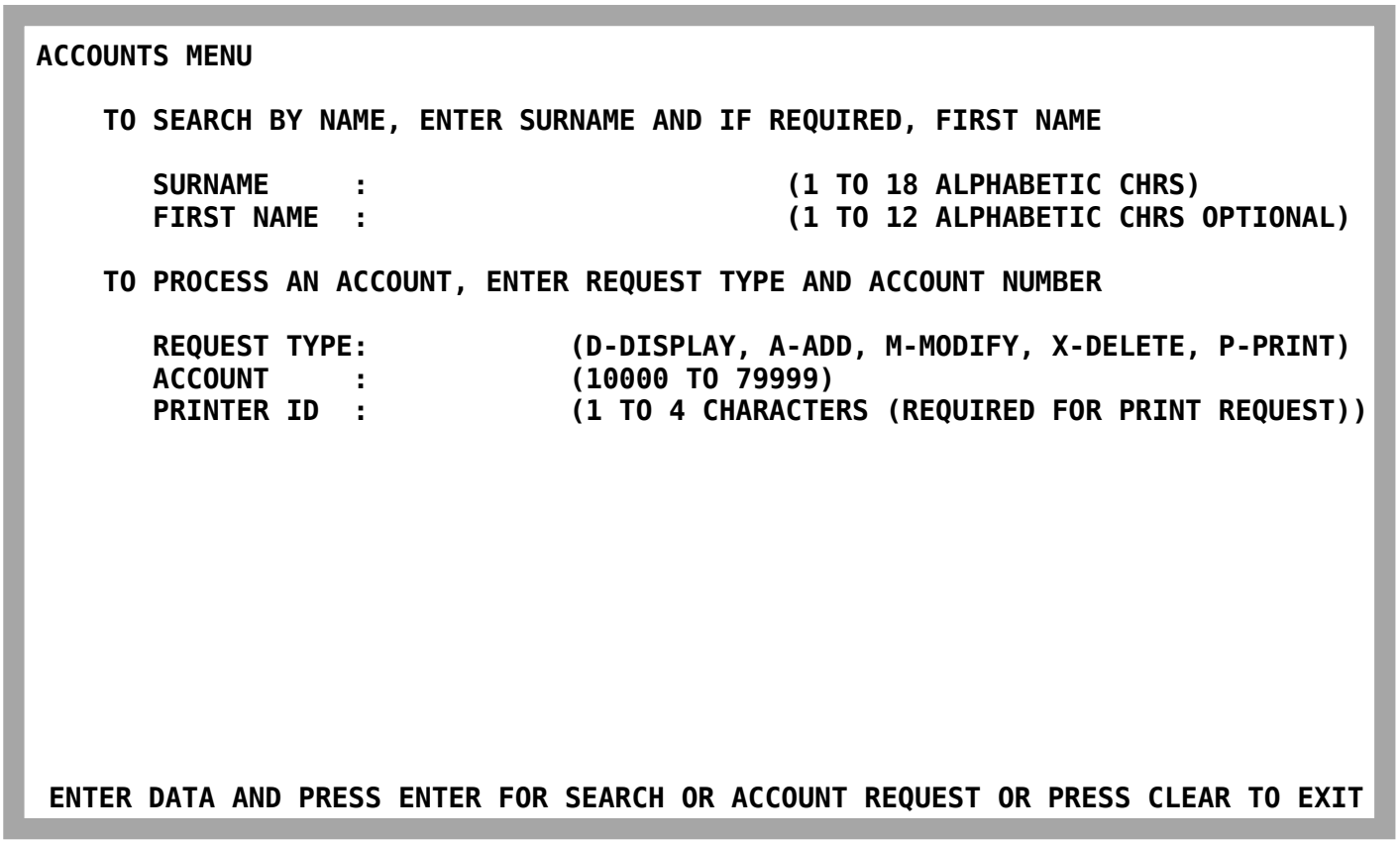


Abb. 8.2.7
Leere Accounts Menu MAP

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

Die NACT Transaktion verwendet zwei unterschiedliche Maps, mit den Namen „Accounts Menu“ bzw. „Accounts“. Der Name wird jeweils links oben in den Bildschirm angegeben.

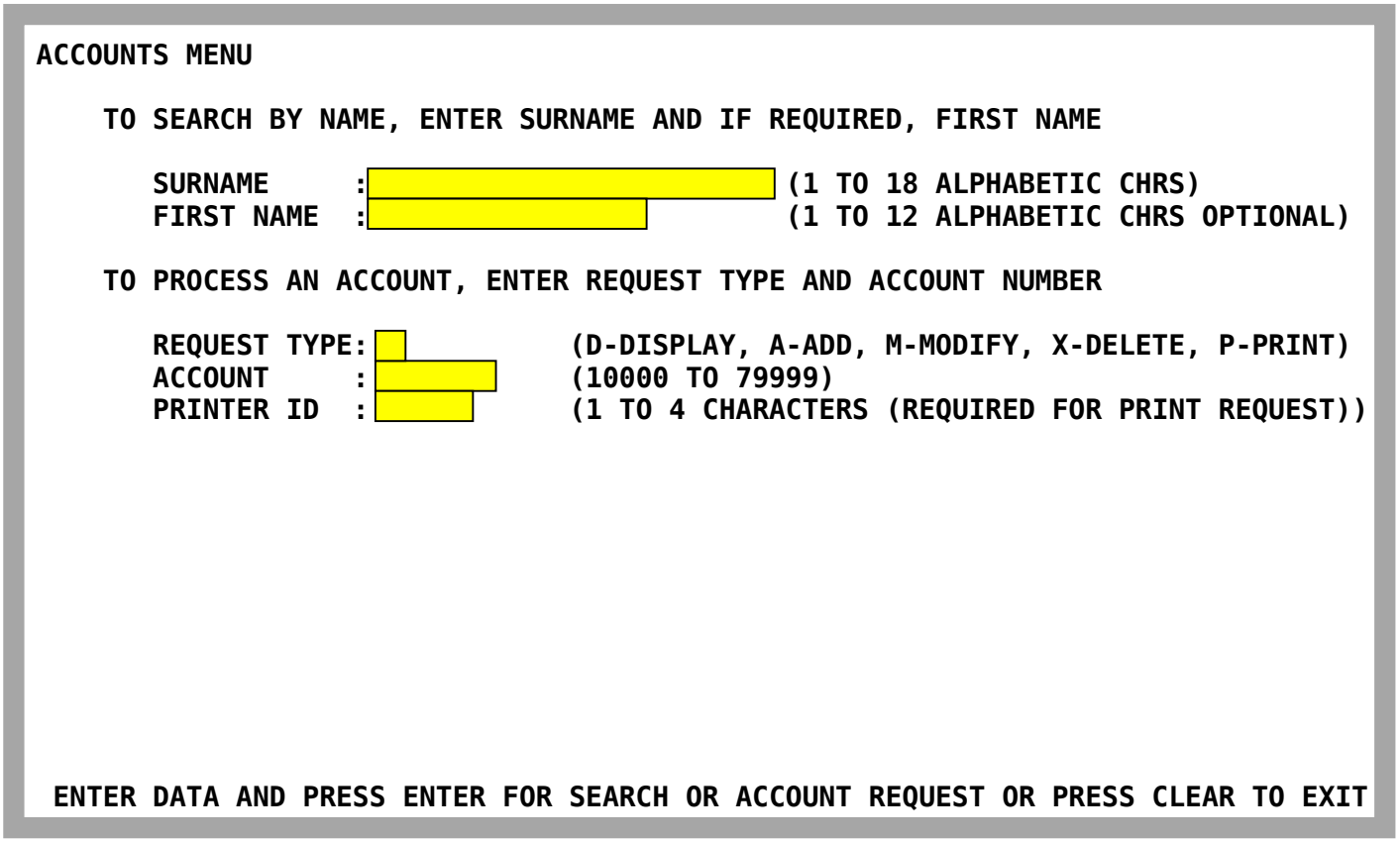


Abb. 8.2.8
Eingabefelder der Accounts Menu Map

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 weiß.

Bitte Zurückhaltung mit Ihren Verbesserungsvorschlägen.

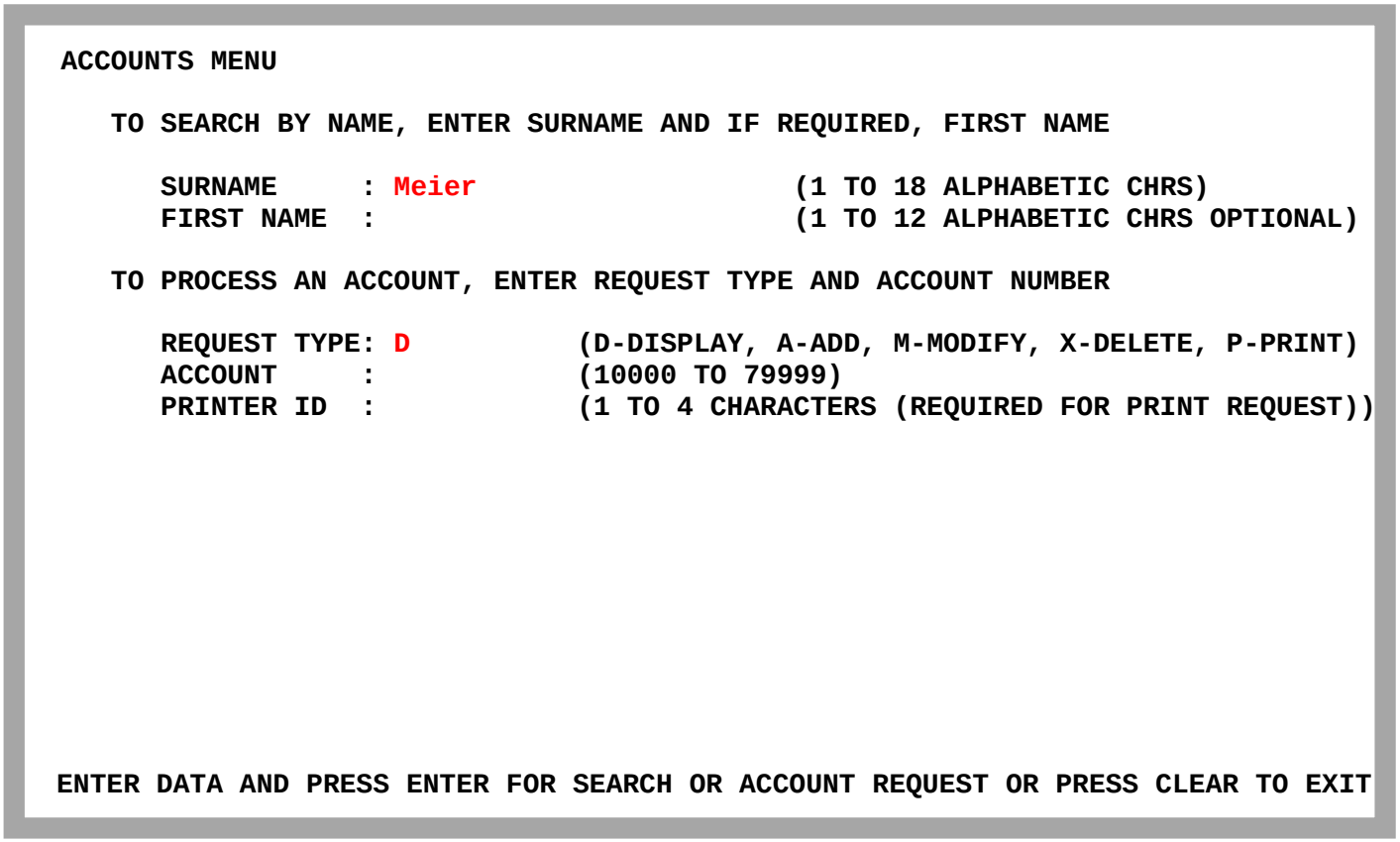


Abb. 8.2.9
Dateneingabe in die Accounts Menu Map

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.

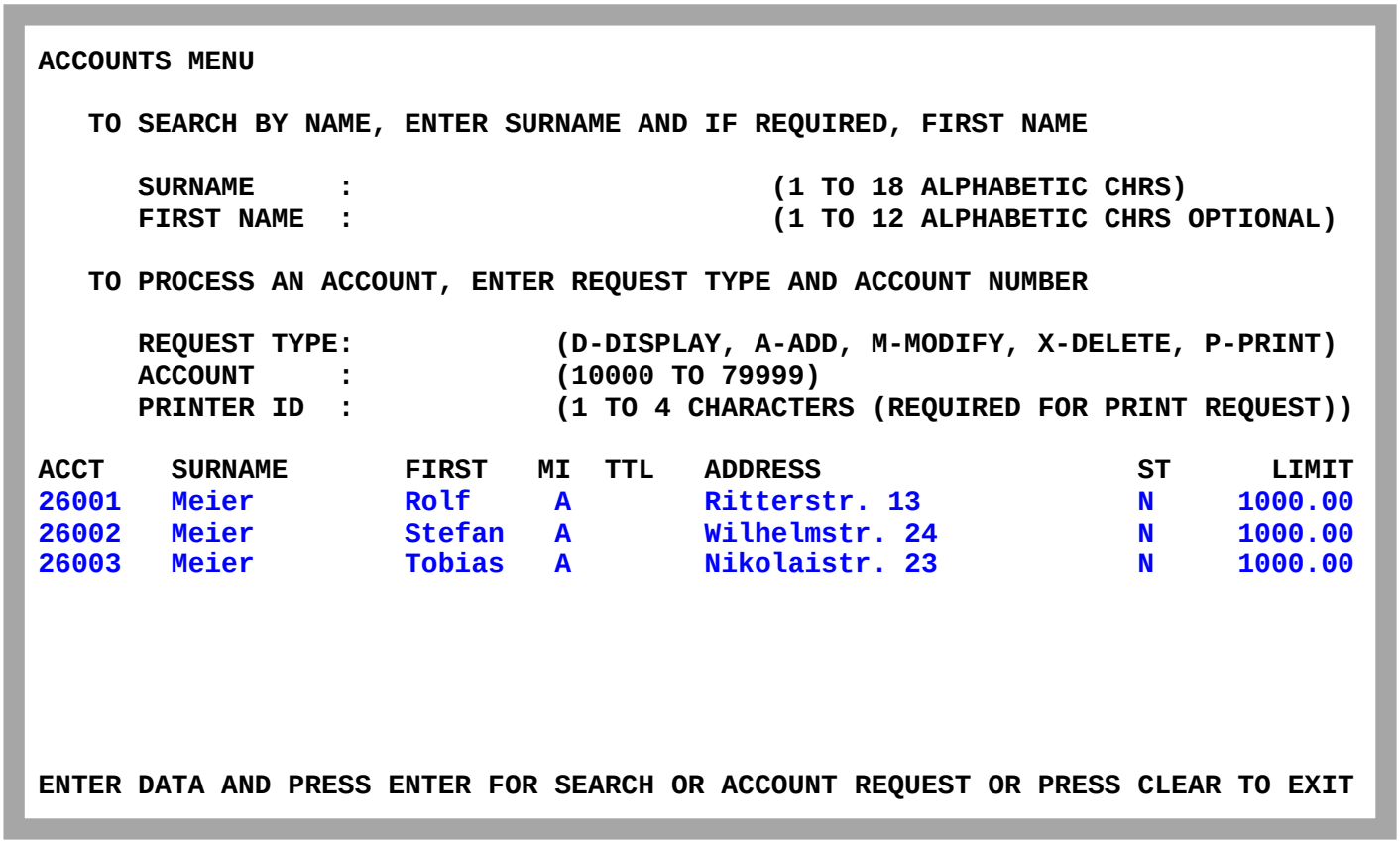


Abb. 8.2.10
Antwort von CICS

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

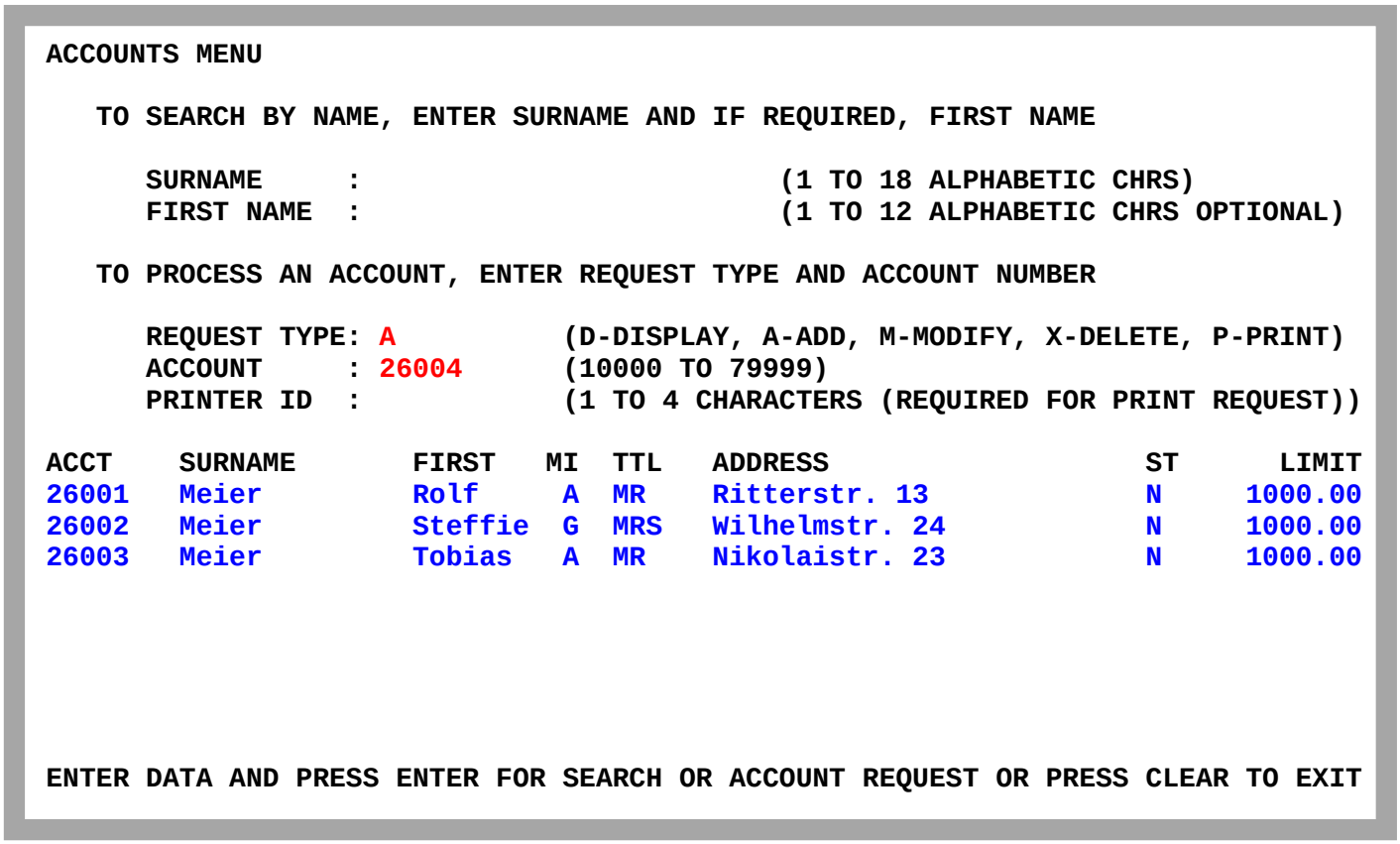


Abb. 8.2.11
Eingabe der Aufforderung „Add“ in die Accounts Menu Map

Der Sachbearbeiter gibt als Request Type A (für add) ein. Damit soll ein neuer Record für Herrn 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
```

Abb. 8.2.12
Leere Accounts MAP

CICS stellt eine andere MAP auf den Bildschirm, die Accounts Map. 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.

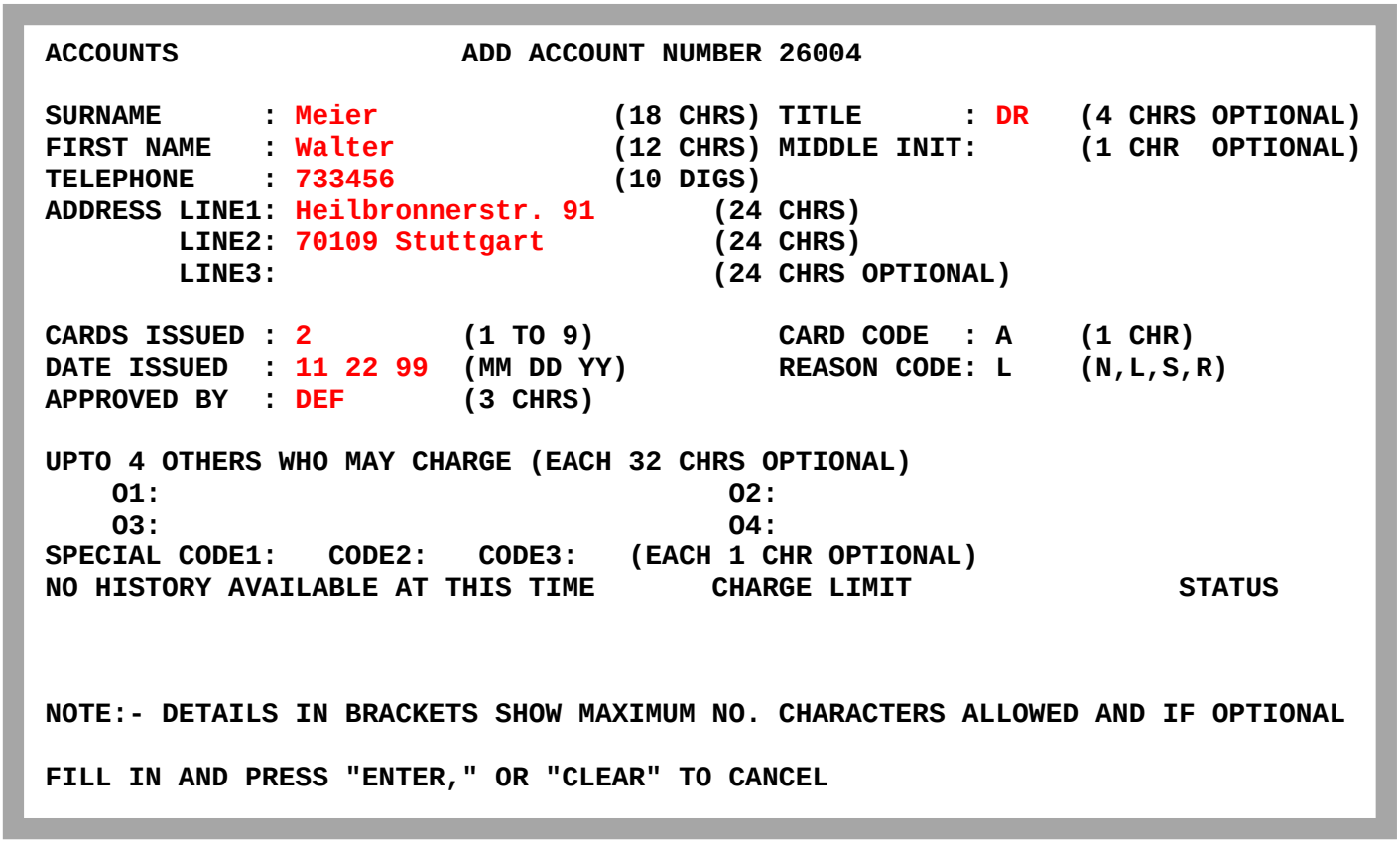


Abb. 8.2.13
Daten Eingabe in die Accounts Map

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.

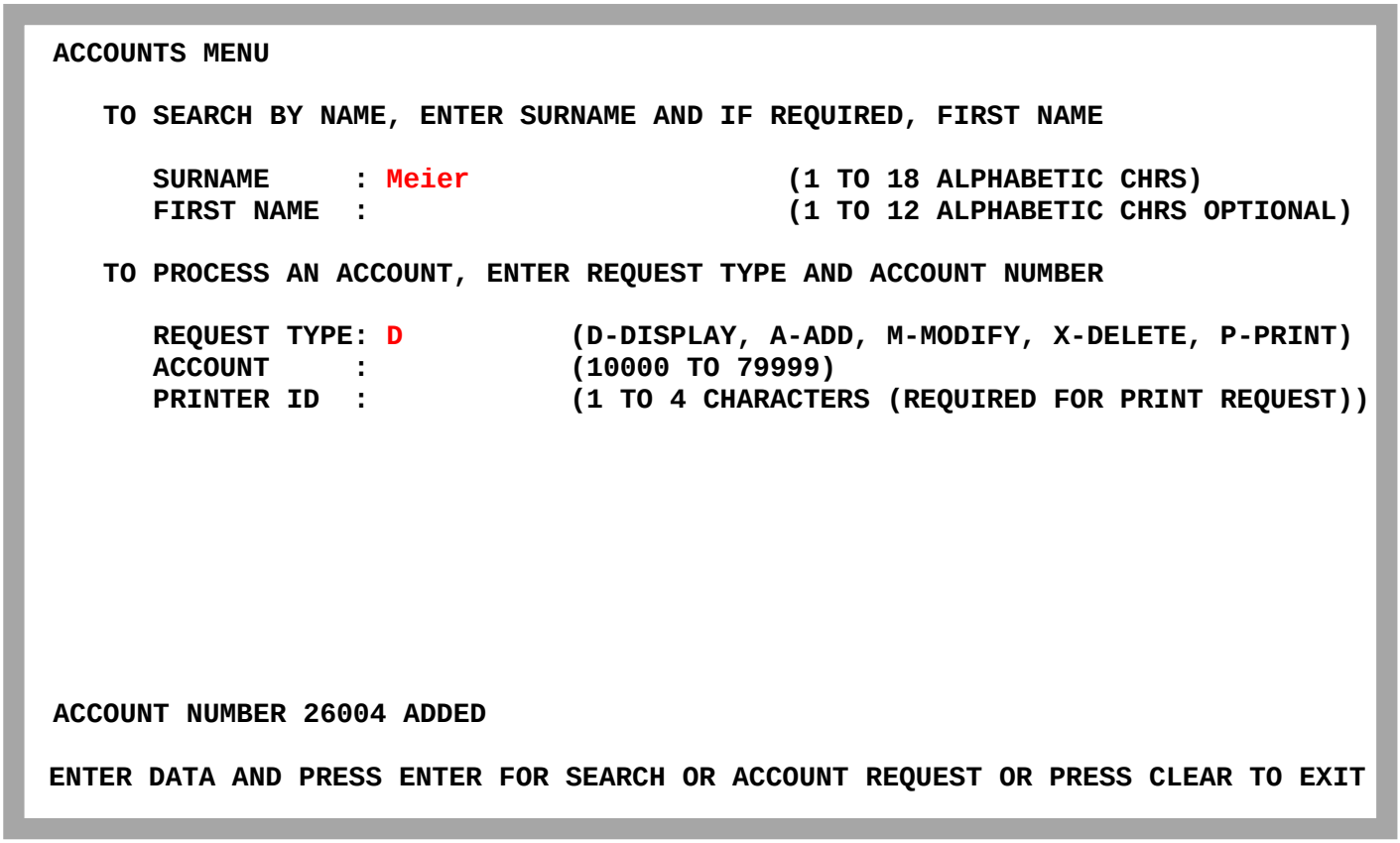


Abb. 8.2.14
Zurück zur Accounts Menu Map

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

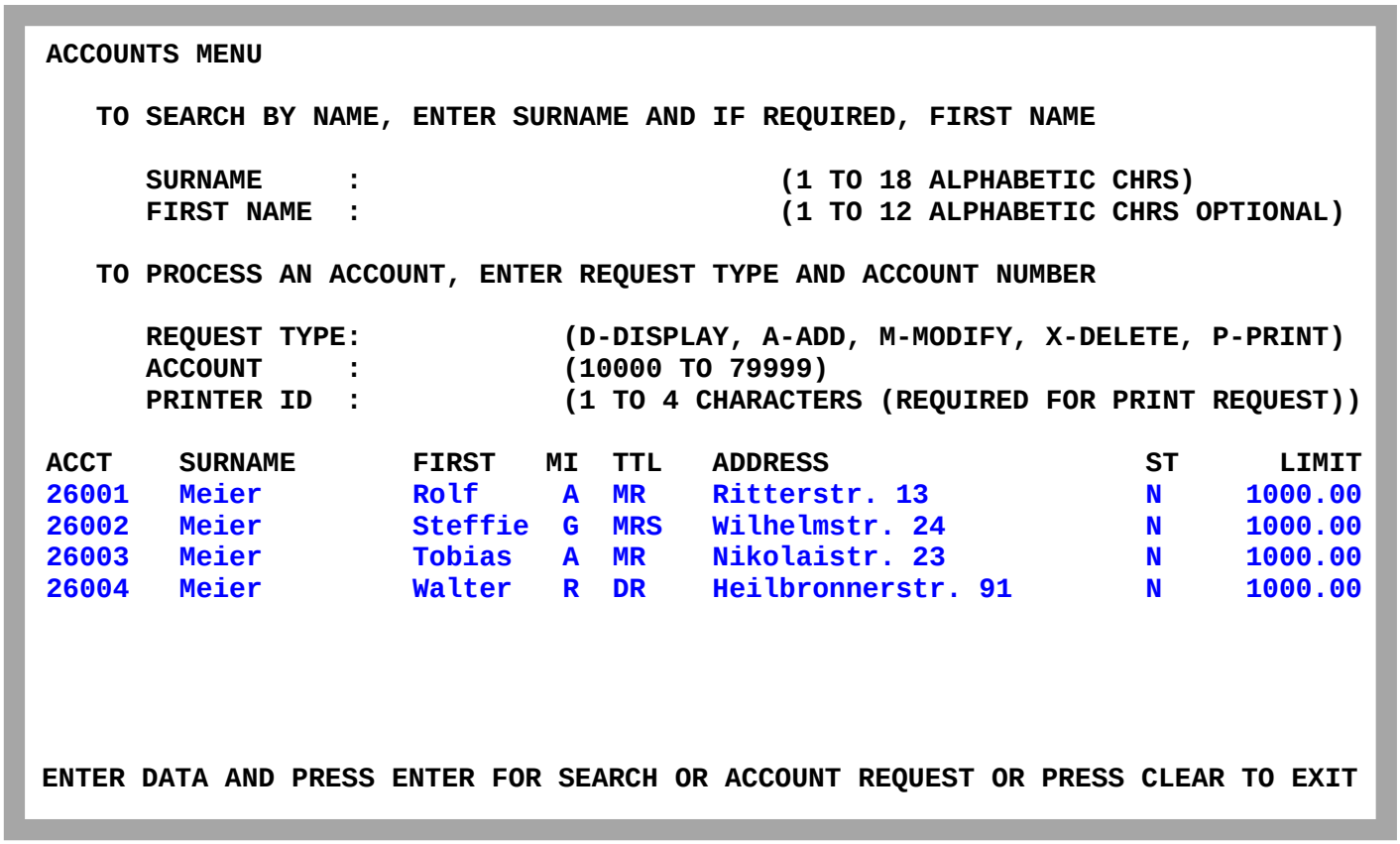


Abb. 8.2.15
CICS Ausgabe

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

Damit wurde die Transaktion erfolgreich abgeschlossen.

8.2.4 Moderne Oberflächen

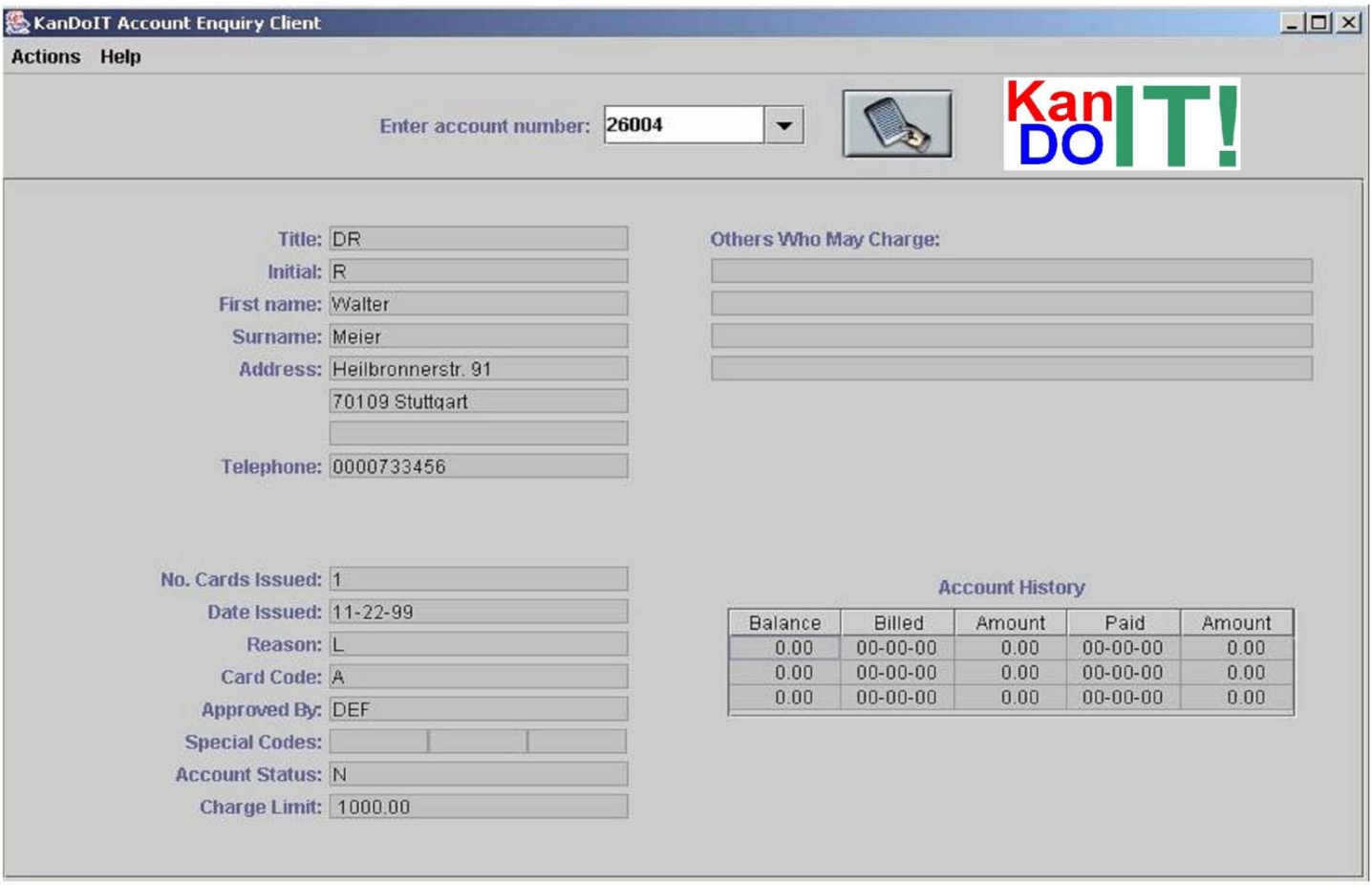


Abb. 8.2.16
Grafische Bildschirmdarstellung

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.

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.

Eine grafische Ausgabe der Abb. 8.2.13 ist in Abb. 8.2.16 gezeigt. In den meisten Unternehmen kann der CICS Benutzer zwischen mehrere Darstellungsalternativen auf seinem Bildschirm wählen. Interessanterweise ist die klassische BMS Wiedergabe bei relativ vielen Benutzern nach wie vor recht populär.

8.3 CICS Nucleus

8.3.1 Ablauf einer CICS Transaktion

Unter CICS werden viele Transaktionen gleichzeitig und parallel verarbeitet. Hierbei wird die „Isolation“ der vier ACID Bedingungen strikt eingehalten.

Unter dem z/OS Betriebssystem laufen alle CICS Anwendungen und Dienste, sowie die CICS Nucleus Komponenten im Problemstatus, ungeschützt voneinander, innerhalb eines einzigen virtuellen Adressenraums. Anwendungen und Ressource Manager laufen als Thread-ähnliche Konstrukte innerhalb dieses Adressenraums.

Thread-ähnlich: Scheduling erfolgt durch den CICS Nucleus, und nicht – wie bei echten Threads – durch die Scheduler/Dispatcher Komponente des Betriebssystem Kernels.

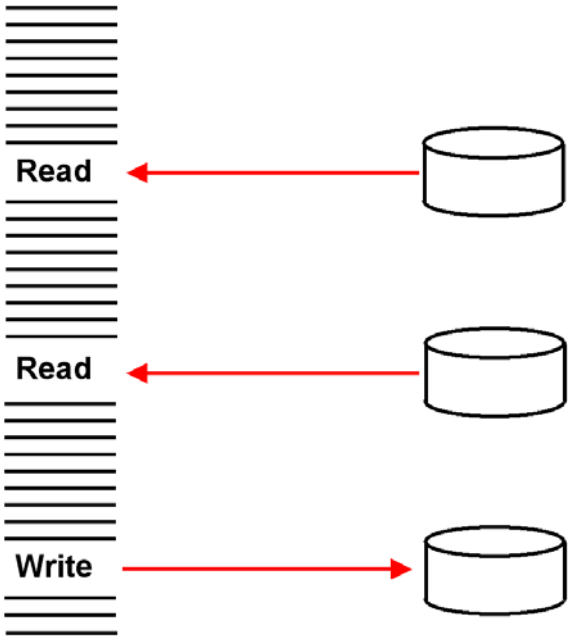


Abb. 8.3.1
Ablauf einer CICS Transaktion

Das Programm für die Ausführung einer Transaktion besteht aus Folgen von Maschinenbefehlen, in die in unregelmäßigen Abständen I/O Operationen eingebaut sind (READ, WRITE, OPEN, CLOSE, CONTROL ....).

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, also 7 Größenordnungen (10^7) länger dauert.

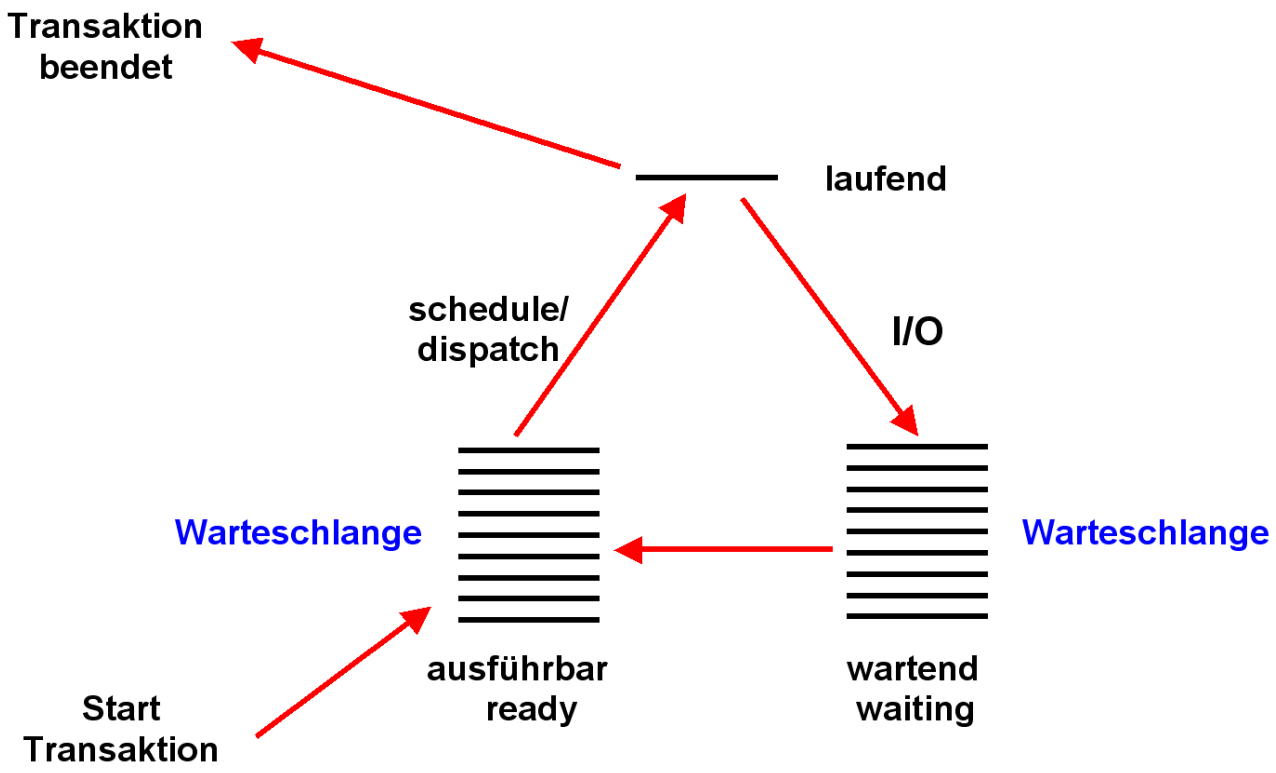


Abb. 8.3.2
Scheduling von Transaktionen

Jede Transaktion rotiert wiederholt durch die Zustände laufend, wartend und ausführbar. Der CICS eigene Scheduler selektiert aus der Queue ausführbare Transaktionen jeweils einen Kandidaten wenn immer die CPU frei wird (wir nehmen hier an, dass die CICS Region nur eine CPU benutzt).

Alle Anwendungsprogramme, die von Transaktionen aufgerufen werden, befinden sich in einer Programmbibliothek. Beim erstmaligen Aufrufen einer Transaktion wird überprüft, ob sich das entsprechende Programm bereits im Hauptspeicher befindet. Wenn nein, wird das Programm zunächst in den Hauptspeicher geladen.

Es kann aber sein, dass eine andere Transaktion das gleiche Anwendungsprogramm bereits benutzt oder benutzt hat. In diesem Fall ist kein Laden aus der Programmbibliothek erforderlich. Wenn mehrere Transaktionen das gleiche Programm verwenden, ist dieses nur einmal im Hauptspeicher vorhanden.

Die Anwendungsprogramme sind deshalb „quasi reentrant“ geschrieben: Eine Transaktion, deren Ausführung unterbrochen wird, hinterlässt de Anwendungscode so, dass eine andere Transaktion den gleichen Code ohne Probleme ausführen kann.

8.3.2 CICS Nucleus

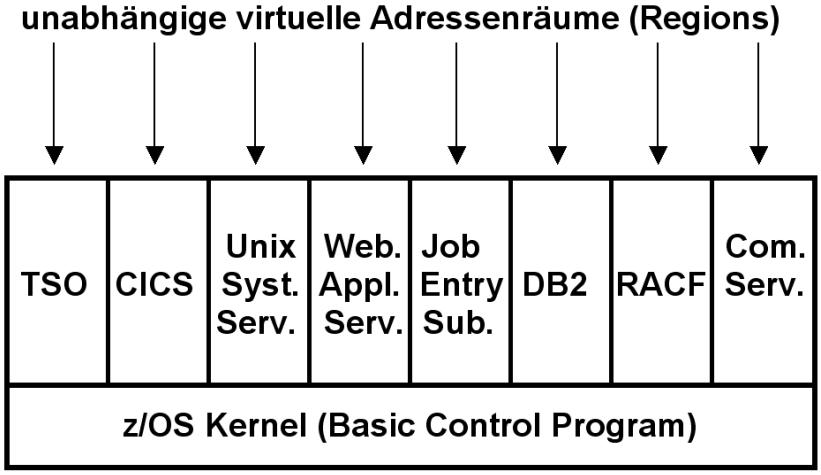


Abb. 8.3.3
z/OS Subsysteme

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

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

- **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**

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 als ein z/OS Prozess in einem einzigen virtuellen Adressenraum (Region). Dieser Adressenraum enthält einmal die CICS Laufzeitumgebung (als CICS Nucleus bezeichnet). Weiterhin laufen alle CICS Transaktionen in der gleichen Region unter Kontrolle des CICS Nucleus. CICS Anwendungsprogramme benutzen EXEC CICS Befehle für alle Schnittstellen (interfaces), die bezüglich Gewährleistung der ACID Eigenschaften relevant sind. Der CICS Nucleus wiederum greift auf den z/OS Überwacher zu.

CICS ist die Schnittstelle zwischen Anwendungsprogrammen, Datenbanken und Netzwerk-(Communication) Komponenten.

Aus Sicht des z/OS Betriebssystems existiert nur eine einzige Anwendung und nur ein Prozess – der CICS Transaktionsmonitor mit allen derzeitig ausgeführten Transaktionen. Die Ausführung einer CICS Transaktion wird als „Task“ bezeichnet. In der CICS Region sind gleichzeitig viele (möglicherweise tausende) von CICS Tasks aktiv. Eine CPU führt in jedem Augenblick nur eine dieser Tasks aus (laufend, running); die übrigen Tasks sind ausführbar (ready) oder wartend (waiting).

CICS verwendet den System z Architektur „Storage Protection Key“ Mechanismus, um den CICS Nucleus von den Anwendungen zu isolieren, die im gleichen Adressenraum laufen. Der CICS Nucleus verwendet normalerweise Storage Protection Key = 8, was einen Zugriff auf den ganzen CICS virtuellen Adressenraum ermöglicht. CICS Anwendungen laufen normalerweise mit Storage Protection Key = 9, was den Zugriff auf den von den Anwendungen benutzten Adressenraum begrenzt.

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

8.3.3 CICS Domänen (Domains)

Der CICS Nucleus besteht aus einer Gruppe von Domänen (domains). Jede Domäne enthält eine Gruppe von Objekten, die einen gemeinsamen Satz von Funktionen ausführen.

Domänen enthalten als „Manager“ bezeichnete Programm Module, Tabellen und Steuerblöcke. Einige wenige Domänen sind für den größten Teil des Verarbeitungsablaufes einer Transaction zuständig:

- **Terminal Manager Domäne**
- **Transaction Manager (XM, auch als Task Manager bezeichnet) Domäne**
- **Program Manager (PG) Domäne**
- **Storage Manager (SM) Domäne**
- **File Manager Domäne**

Die „Manager“ wurden in der Vergangenheit als “Controls” bezeichnet: Terminal Control, Task Control, Program Control, Storage Control und File Control. Die Begriffe Control und Manager haben unter CICS die gleiche Bedeutung.

Anwendungsprogramme (Transaktionen) greifen auf Manager (CICS Management Module) zu um Terminal I/O, File I/O, Program loading und Zugriffe auf andere System Ressourcen für sie durchzuführen.

Anwendungsprogramme (Transaktionen) teilen sich die CICS Region mit anderen CICS Komponenten. Neben den Anwendungsprogrammen und den Management Modulen existieren weitere Arten von Objekten in der CICS Region:

- **Tabellen (Tables) – Die Management Module benutzen CICS System Tables um Informationen über Terminals, Dateien und Anwendungsprogramme zu erhalten und zu verwalten, die Teil des CICS Systems sind. Die Tabellen werden in den Hauptspeicher geladen, wenn CICS gestartet wird. Sie bleiben aktiv, bis CICS heruntergefahren wird. (Beides sind Vorgänge, die nur sehr selten stattfinden; CICS bleibt normalerweise 24 Stunden, 7 Tage/Woche, im Betrieb).**
- **Steuerblöcke (Control blocks) – Die Management Module erstellen Steuerblöcke, mit denen der Status aller Transaktionen festgehalten wird, die in jedem Augenblick aktiv sind. Die Steuerblöcke werden freigegeben, wenn sie nicht mehr benötigt werden, z. B. wenn eine Transaktion das System verlässt.**
- **System Data Sets – Die Management Module und Anwendungsprogramme benutzen CICS System Data Sets für Transaction Logging, System Recovery, und für das Speichern von Ergebnissen, die von anderen Anwendungsprogrammen oder Transaktionen verwendet werden.**

8.3.4 CICS Nucleus Komponenten

CICS läuft als lang laufender Stapelverarbeitungsjob in einem einzigen virtuellen Adressenraum (Region in z/OS 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) laufen in dem gleichen virtuellen Adressenraum wie alle Anwendungen. Jede Nucleus Komponente hat eine zugeordnete Tabelle: TCT, PCT, PPT, FCT. Über den Scratchpad 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.

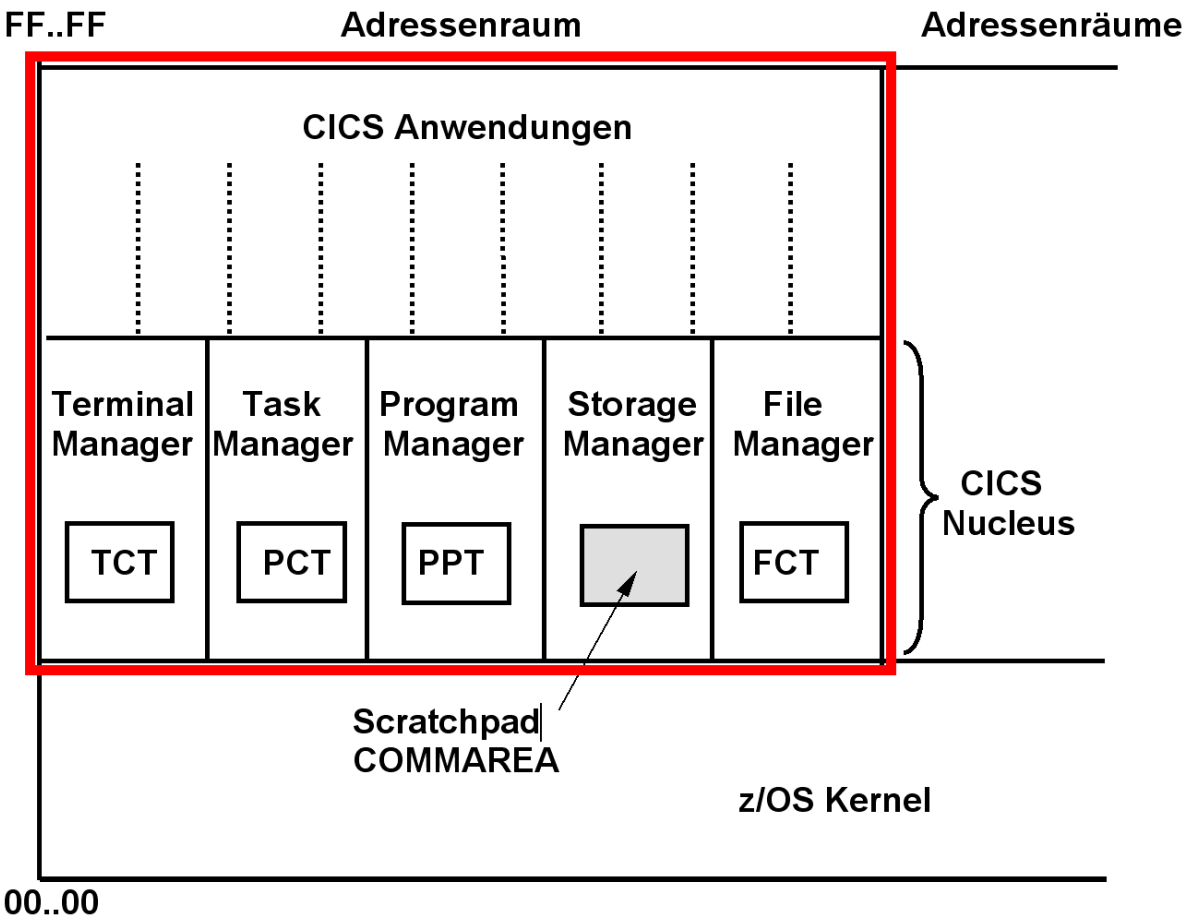


Abb. 8.3.4
CICS Bestandteile

Die Überwachung und Steuerung der Transaktionsverarbeitung erfolgt vor allem durch drei Komponenten:

- **Der Task Manager ist für den Empfang von *Transaction Requests* zuständig, sowie für die Erstellung und Steuerung von *Tasks*, welche die Transaction Requests verarbeiten.**

- **Der Program Manager 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.**

- **Der Storage Manager ist zuständig für die Zuordnung von (virtuellem) Speicherplatz, der für die Transaktionsverarbeitung benötigt wird.**

Terminal Manager. Task Manager. Program Manager, Storage Manager und File Manager wurden früher als Terminal Control, Task Control Program Control, Storage Control und File Control bezeichnet.

8.3.5 Task Manager

Eine Task ist die Ausführung einer Transaktion durch CICS. Während der Ausführung hat die Task die Verfügungsgewalt über eine CPU. Wenn die Task von CICS die Ausführung eines Dienstes verlangt, (z. B. Einlesen von Daten) wird sie in den Wartezustand versetzt.

Während diese Task wartet, wird die nächste ausführbare (ready to run) Task vom CICS Nucleus gestartet. Auf diese Art bewirkt CICS Multitasking innerhalb seiner Region ohne Mitwirkung des Scheduler/Dispatchers des Betriebssystem Kernels. Dies ist der Unterschied zu der Verarbeitung von normalen Threads innerhalb eines Address Spaces.

Wenn eine Task erstmalig gestartet wird (dispatched for execution), ruft der Task Manager als erstes den Program Manager (PG) auf. PG ruft das erste Anwendungsprogramm (von möglicherweise mehreren Anwendungsprogrammen) auf, das für die Ausführung der Transaktion benötigt wird.

8.3.6 Program Manager

Um das gewünschte Anwendungsprogramm aufzufinden, durchsucht der Programm Manager (PG) den Processing Program Table (PPT) nach dem Namen des Moduls, welches als nächstes aufgerufen werden soll.

CICS lädt lediglich eine einzige Kopie des Anwendungsprogramms in den Hauptspeicher, auch wenn mehrere Transaktionen den gleichen Anwendungsprogrammcode gleichzeitig benutzen wollen. Damit das funktioniert, müssen Anwendungsprogramme quasi-reentrant geschrieben werden.

- **Wenn sich das gewünschte Anwendungsprogramm bereits im virtuellen Speicher befindet, ruft PG es auf, und ermöglicht die Ausführung wie gewünscht.**

- **Wenn sich das gewünschte Anwendungsprogramm nicht im virtuellen Speicher befindet, lokalisiert PG es in der CICS Program Library, lädt es in den virtuellen Speicher und ruft es auf.**

Wenn das derzeitig laufende Programm terminiert, übernimmt PG wiederum die Verantwortung und entscheidet was als Nächstes zu erfolgen hat.

An der Ausführung einer einzelnen CICS Task können mehrere Anwendungsprogramme beteiligt sein. Die Verarbeitung kann seriell erfolgen (ein Programm nach dem anderen). Alternativ kann ein Programm ein anderes als ein Unterprogramm aufrufen. PG steuert die Übergabe von einem Programm auf ein anderes.

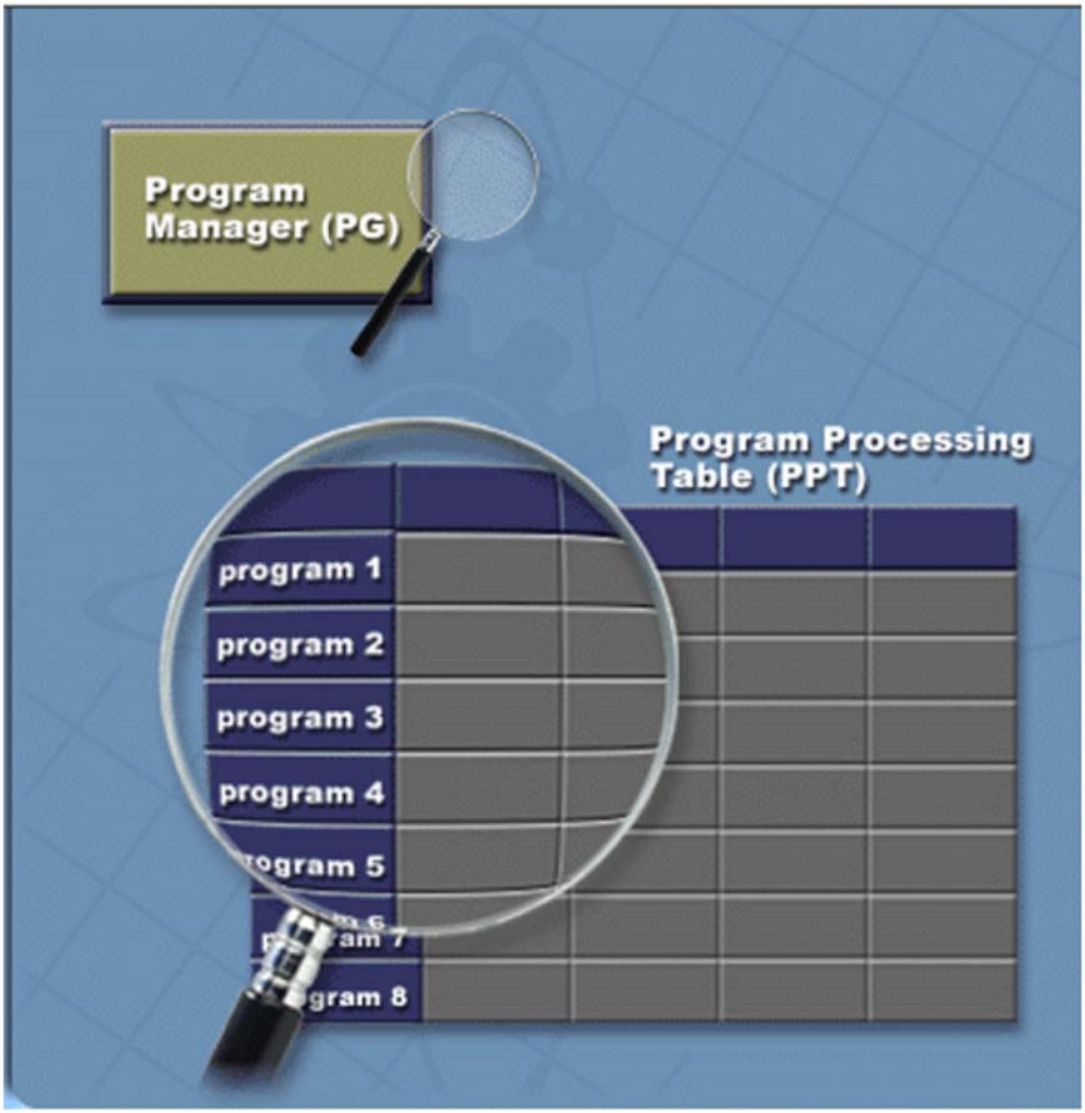


Abb. 8.3.5
Program Processing Table

Der Program Manager ist die zuständige Domäne für:

- **Auffinden des Anwendungsprogramms**
- **Laden des Anwendungsprogramms**
- **Starten des Anwendungsprogramms**

Er benutzt hierfür seinen Program Processing Table (PPT).

8.3.7 Storage Manager

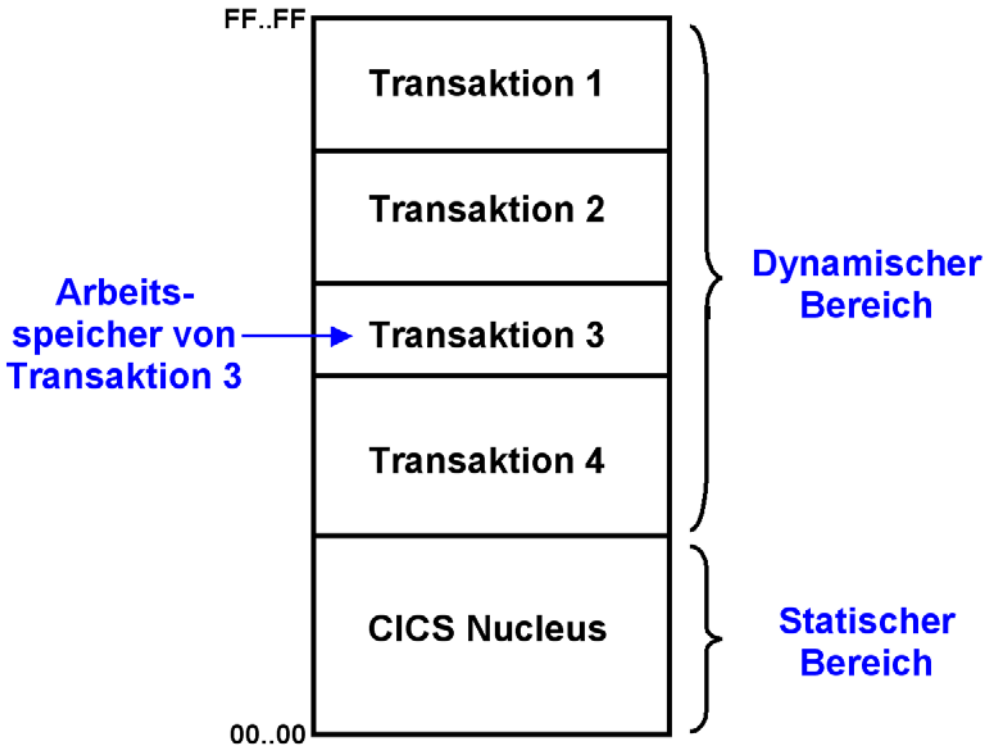


Abb. 8.3.6
Aufteilung des CICS virtuellen Speicherraums

Der Storage Manager (SM) ist die Domäne, die für die Zuordnung (allocation) von virtuellem Speicher verantwortlich ist, der für die Ausführung einer Transaktion benötigt wird.

Unter z/OS verwaltet CICS den virtuellen Speicherplatz seiner Region. Der Storage Manager (SM) verwaltet den dynamischen Bereich des virtuellen Speichers. Dies ist der Teilbereich des virtuellen Speichers, der übrig bleibt nachdem CICS (der CICS Nucleus) geladen wurde.

Dynamischer Speicher wird für Programme, Ein/Ausgabe Bereiche und Arbeitsbereiche genutzt.

Auf Anforderung der anderen CICS Domain Managers bewirkt der Storage Manager die Zuordnung, Freigabe und Verwaltung von verfügbarem virtuellem Speicherplatz.

Vor der Einführung der 64 Bit Adressierung betrug die maximale Größe der CICS Region 2^{31} = 2 GByte. Für tausende von Transaktion muss virtueller Speicherplatz zugeordnet, und nach Abschluss wieder freigegeben werden.

Auch nach der Einführung der 64 Bit Adressierung ist virtueller Speicherplatz eine kritische Ressource. Platz in dem Dynamischen Bereich der CICS Region (Dynamic Storage Area, DSA), muss vom Storage Manager verwaltet und den einzelnen Transaktionen zugeordnet werden. Da der reentrant Program Code von mehreren Transaktionen benutzt werden kann, befindet er sich außerhalb des Arbeitsspeichers einer jeden Transaktion.

8.3.8 COMMAREA

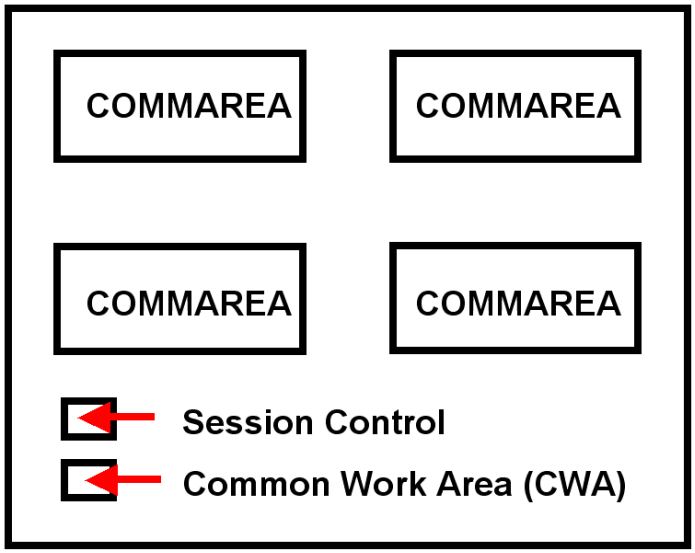


Abb. 8.3.7
Scratchpad

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.

COMMAREA (Communication Area) ist eine Art Ein/Ausgabepuffer, über den eine Transaktion Daten mit dem Klienten (Terminal) austauschen kann.

COMMAREA ist von so zentraler Bedeutung für die CICS Transaktionsverarbeitung, dass es sich lohnt, den Namen zu merken: COMMAREA

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

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

Eine CICS Anwendung kann, muss aber nicht, COMMAREA als I/O Puffer benutzen. Es ist möglich, die Ein/Ausgabe unter Umgehung von COMMAREA zu programmieren. Dies verbessert die Performance, und wurde in der Vergangenheit häufig gemacht. Viele (nicht alle) früher geschriebene CICS Anwendungen machen hiervon Gebrauch.

Heute gilt dies als sehr schlechter Programmierstiel. Praktisch alle in den letzten 10 Jahren geschriebenen CICS Anwendungen nutzen COMMAREA.

Der Execute Interface Block (EIB) ist ein CICS Bereich, auf den Anwenderprogramme mit EXEC CICS Kommandos zugreifen und CICS Informationen abfragen können. Der EIB besteht aus einer Reihe von Feldern. z.B. steht die Länge der COMMAREA in dem Parameter EIBCALEN des EIB.

Für CICS Anwendungsprogramme gibt es zwei Arten, Daten temporär zu speichern: Temporary Storage und Transient Data.

Temporary Storage (TS) ist eine gewöhnliche Datei. TS ist als VSAM-Entry Sequenced Data Set (ESDS) organisiert und wird von CICS unter dem Namen DFHTEMP angesprochen

DFHTEMP wird als Queue benutzt. Entweder ist dies eine einzige TS Queue, auf die viele Programme zugreifen. Häufiger sind Situationen, bei denen für jeden Benutzer eine benutzerspezifische Queue angelegt wird. Anlegen und Löschen einer TS-Queue ist Aufgabe des Anwendungsprogrammierers. Das Cobol Statement

EXEC CICS WRITEQ TS QUEUE('ACCTLOG') FROM(ACCTDTLO) LENGTH(DTL-LNG) END EXEC

ist ein Beispiel für die Nutzung der TS Queue. TS Queue Sätze sind bis zu 32767 Bytes lang.

Transient Data (TD) ist ebenfalls VSAM-ESDS organisiert und wird von CICS unter dem Namen DFHINTRA angesprochen

Die Zwischenspeicherung in der TS-Queue bietet sich an, wenn Daten innerhalb von CICS übergeben werden sollen. Transient Data (TD) sind vorgesehen, um Daten zwischen CICS und seiner Umgebung auszutauschen.

8.3.9 File Manager

Das File Manager CICS Module ist für den Zugriff auf Daten zuständig, welche in Data Sets gespeichert sind, z.B.:

- **VSAM files**
- **Temporary Storage Queues**
- **Transient Data Queue**

Bei einem Datenbank Zugriff benutzt CICS Locking Einrichtungen der Datenbank. Derartige Einrichtungen sind bei VSAM nicht vorhanden. Sie werden deshalb von dem CICS File Manager zur Verfügung gestellt.

Neben Datenbanken kann CICS mit Daten arbeiten, die in VSAM Data Sets gespeichert sind. Hiervon wird häufig Gebrauch gemacht. VSAM (im Gegensatz zu DB2) beinhaltet jedoch keine Funktion die es ermöglicht, Files von unabhängigen Anwendungen in unterschiedlichen CICS Regions gemeinsam zu nutzen (share), mit voller Update Fähigkeit. Der File Manager (mittels seiner DFSMS RLS Komponente) macht dies möglich.

Die beiden wichtigsten CICS Module, die bei einem VSAM Datenzugriff beteiligt sind, sind der File Manager und der File Control Table (FCT).

Der File Manager benutzt die Daten in der File Control Table (FCT) für einen Zugriff auf den Dataset. Jeder FCT Eintrag enthält:

- **den Namen des Data Sets,**
- **die benutzte Access Method (z.B. VSAM),**
- **die mögliche Zugriffsart (z.B. update, read-only),**
- **andere Data Set Attribute, z.B. die maximale Anzahl von Tasks, die gleichzeitig auf den Data Set zugreifen können.**

8.3.10 Weitere Funktionen

Eine CICS Region besteht aus mehreren Domains, von denen jede ihre eigenen Resourcen und Funktionen hat.

Domains kommunizieren miteinander über Schnittstellen, die als "Gates" bezeichnet werden.

CICS verfügt über Domains, die für die eigentliche Transaktionsverarbeitung zuständig sind (z.B. die Storage Manager, Task Manager und Program Manager Domains), Daneben existieren CICS Domains, die zuständig sind für

- **das Laden von Anwendungsprogrammen (Loader Domain)**
- **Dispatching CICS Messages (Message Domain)**
- **Interfacing mit externen Security Systemen (Security Manager Domain)**

Zwei sehr wichtige Domains sind die Domain Manager Domain sowie die Application Domain. Die Domain Manager Domain unterhält einen Katalog mit Kenndaten über alle anderen Domains.

Die Application Domain enthält einige Schlüssel-Komponenten, wie:

- **Application und System services,**
- **Extended recovery facility (XRF) für Fehlersituationen,**
- **Intercommunication Segmente wie Multiregion operation (MRO) und Inter-System Communication (ISC) Systemsteuerung (wird in Abschnitt 9.3 besprochen).**

8.4 CICS Ablaufsteuerung

8.4.1 Bildschirm Wiedergabe

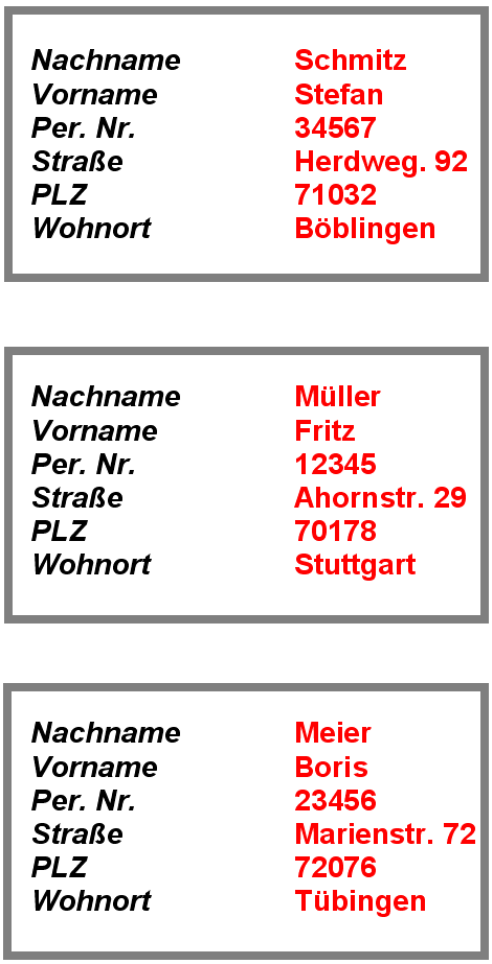


Abb. 8.4.1
Statische und dynamische Bildschirm Information

Vorgefertigte Daten, die bei einer Bildschirmausgabe des gleichen Typs immer wieder identisch sind, werden als „MAP“ bezeichnet.

In Abb. 8.4.1 sind drei Bildschirmwiedergaben dargestellt, 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 Map hierfür vorgesehen sind.

Eine bestimmter Typ einer Transaktion verwendet in der Regel mehrere Maps (z.B. zwei bei der NACT Transaktion, wesentlich mehr bei den meisten Transaktionen).

Die Summe aller Maps eines Transaktions-Typs werden als Mapset bezeichnet.

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.

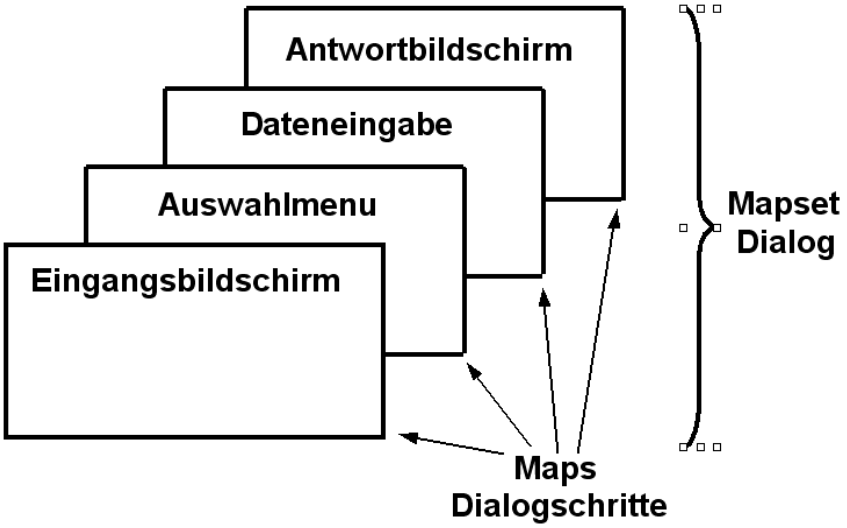


Abb. 8.4.2
Die verschiedenen Maps einer CICS Anwendung bilden den Mapset

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

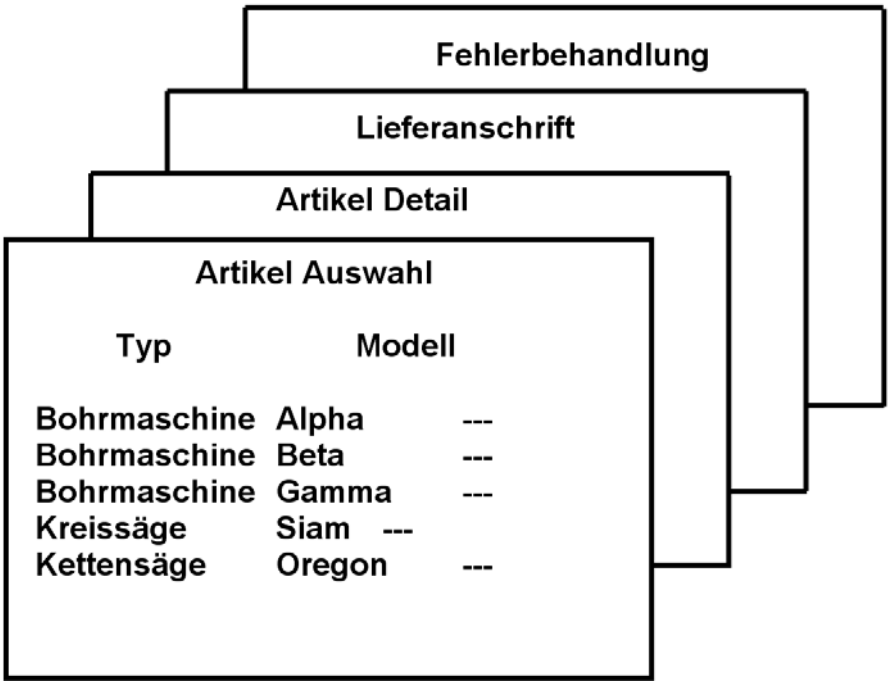
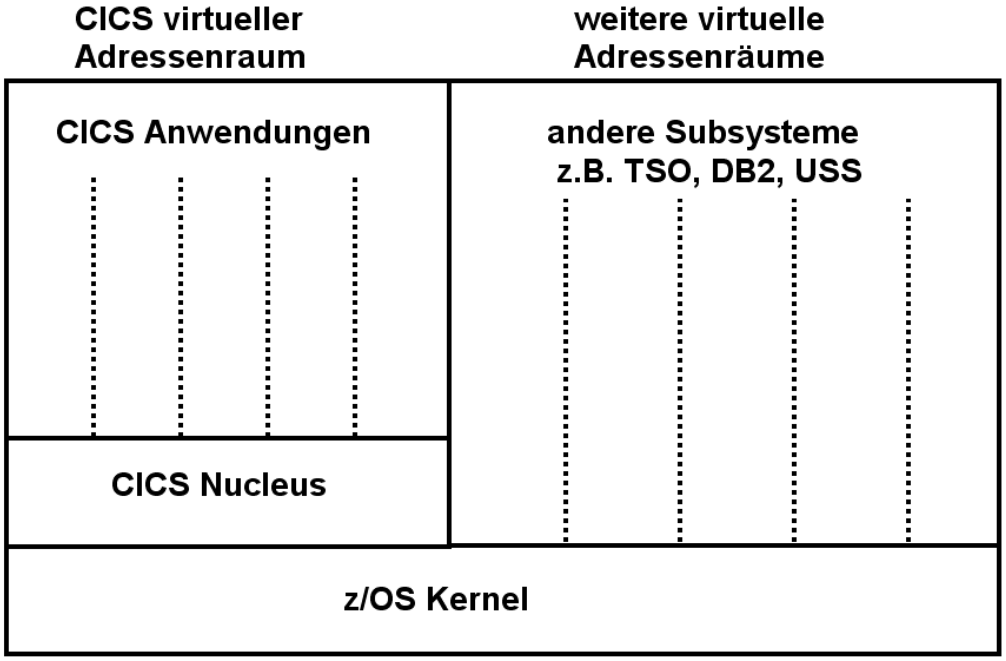


Abb. 8.4.3
Beispiel eines Mapset für einen Lieferauftrag, bestehend aus 4 Maps

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).

8.4.2 CICS Erstellung der Bildschirm Ausgabe



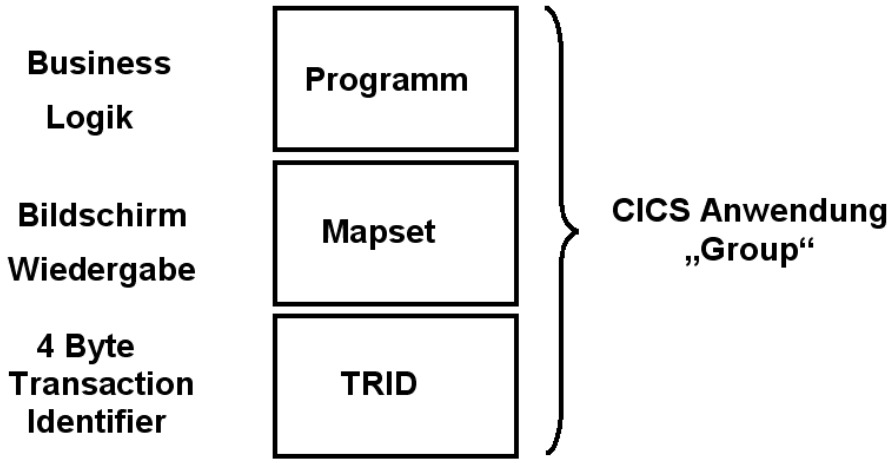


Abb. 8.4.4
Bestandteile einer Group

Die Erstellung der Ausgabe erfolgt in 3 Schritten:

1. **Die richtige Map des Mapsets laden (statische Information)**
2. **Den Mapset mit dynamischer Information ergänzen.**
3. **Beides an den Terminal (Klienten Rechner) senden.**

Die Ausführung der Transaktionsverarbeitung bewirkt die Erstellung von dynamischer Information. Die statische Information liegt in Form einer vorgefertigten Map bereits vor.

Anwendungen, die auf einem Transaktionsserver 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 (TRID).

Die Anwendungsentwicklung kann auf einem externen Rechner oder in einer anderen LPAR erfolgen. Beispiele sind die CICS Anwendungsentwicklung unter TSO und ISPF, unter dem z/VM Betriebssystem oder unter Windows und Eclipse mit den RDz (Rational Developer for System z) plugin. Im letzteren Fall erzeugt ein Crosscompiler z/OS executable code.

8.4.3 Entwicklung und Installation einer neuen CICS Anwendung

CICS ist ein logischer Server, der (neben anderen logischen Servern) auf einem physischen z/OS Server Läuft. Auf CICS laufen nur fertige Anwendungen; neue Anwendungen können nicht auf dem CICS Transaction Server entwickelt werden.

Ähnliches gilt für Java Anwendungen, die unter z/OS typisch auf einem Web Application Server laufen. Ein weiteres Beispiel sind MQSeries Anwendungen (Kapitel 10), die unter z/OS unter einem „Queue Manager“ ausgeführt werden.

Für die Entwicklung neuer CICS Anwendungen braucht man eine Entwicklungsumgebung. In unserer ersten CICS Übungsaufgabe entwickeln wir eine einfache Hello World CICS Anwendung unter TSO und installieren sie dann unter CICS, um sie dort auszuführen. Andere populäre Entwicklungsumgebungen für CICS Anwendungen sind z/VM und besonders Eclipse mit dem RDz (Rational Developer for System z) Plugin.

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-Komponente 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.

8.4.4 Ablaufsteuerung einer CICS Transaktion

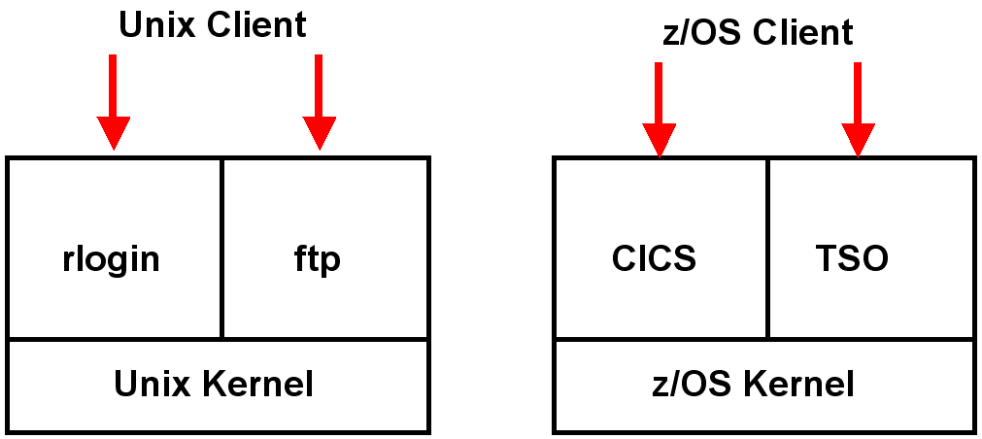


Abb. 8.4.5
Aufbau einer Sitzung

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

Äquivalent besteht z.B. bei einem Unix System die Möglichkeit, sich mit rlogin oder ftp in zwei unterschiedliche Shells einzuloggen.

CICS verfügt über mehrere Shells, von denen CEDA die wichtigste ist.

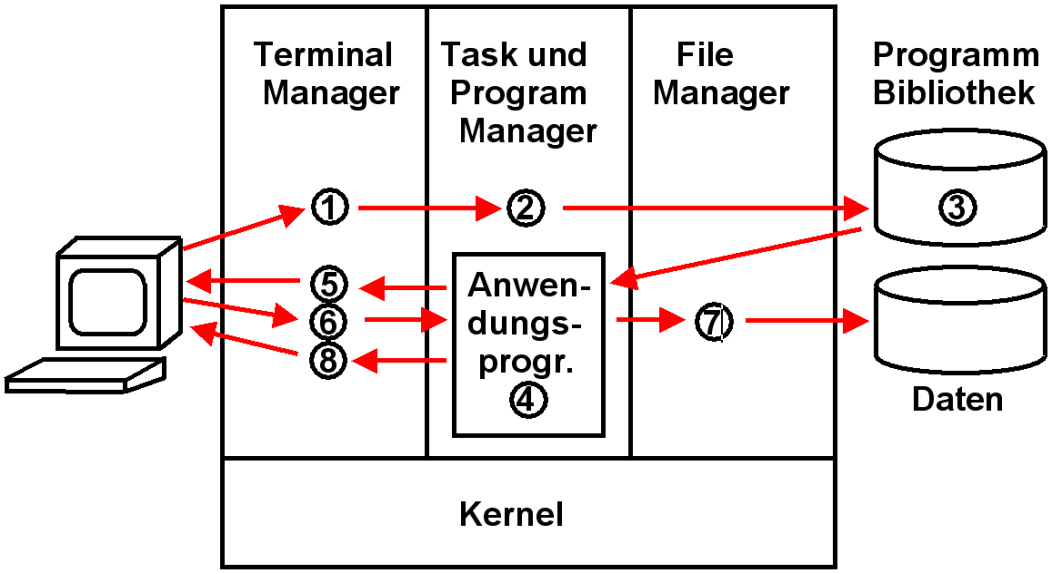


Abb. 8.4.6
Ablaufsteuerung einer CICS Transaktion

Beim Logon erstellt das CICS Subsystem eine Sitzung (Session). Die login Nachricht enthält die Adresse des Klienten. Der CICS Terminal Manager identifiziert den Klienten. Dies geschieht in vielen Fällen einmal pro Tag, und der Benutzer startet danach viele Transaktionen.

Bein Start einer neuen Transaktion verifiziert der Terminal Manager, dass bereits ein gültiges Logon und Authentifizierung vorliegt. Hierfür benutzt Terminal Manager Information, die in seinem Terminal Control Table (TCT) enthalten ist. Dieser enthält einen gültigen Eintrag für jeden eingeloggten Terminal.

- **Eine neue Transaktion ruft CICS mit Hilfe ihrer 4 Byte Transaction ID (TRID) auf. Der CICS Terminal Manager prüft, ob eine laufende Transaktion (eine Session, die fast immer aus mehreren Schritten besteht) mit dem Klienten bereits existiert. Wenn nein, werden die ersten 4 Bytes der Nachrichten als TRID interpretiert. Terminal Manager übernimmt die Eingabenachricht und speichert sie ab**

- **Die TRID am Anfang der Nachricht wird von CICS als solche identifiziert und an Task Manager weitergereicht. (siehe Abb. 8.4.6). CICS interpretiert die Nachricht als Transaktion und ruft das entsprechende Anwendungsprogramm auf.**

- **Das Anwendungsprogramm befindet sich entweder schon im Arbeitsspeicher oder wird vom Program Manager aus der Programmbibliothek geladen. Der Processing Program Table PPT der Program Manager Domain enthält Information über alle CICS interne und alle Benutzer geschriebenen Anwendungen.**

- **Ein CICS Prozess (Task) wird erzeugt. Der Prozess führt das Anwendungsprogramm aus. Information über alle laufenden Transaktionen ist im Program Control Table (PCT) Table festgehalten. Task Manager liest aus seinem PCT Table die zu der TRID gehörige Group aus, darunter Referenzen auf Mapset und Anwendungsprogramm. Das Anwendungsprogramm liest die Nachricht des Klienten.**

- **Terminal Manager baut ein Bildschirm Menü auf (z.B. mit BMS, oder mit Java Präsentationslogik) welche dem Benutzer eine Spezifikation der durchzuführenden Aktivität ermöglicht.**

- **Weitere Eingaben werden von Terminal Manager entgegengenommen und zur Verarbeitung weitergereicht. Die gelesenen Daten (Unit Record) werden von Terminal Manager aufbereitet.**

- **File Manager liest/schreibt gewünschte Daten aus einem VSAM Data Set. Für Zugriffe auf DB2 oder IMS Datenbanken existieren ähnliche SQL oder DL/I Komponenten.**

Wenn vom Klienten die nächste Nachricht eintrifft, erinnert sich CICS Terminal Manager, dass eine Sitzung bereits besteht. Weitere Eingaben werden von Terminal Manager entgegengenommen und zur Verarbeitung Task Manager weitergereicht.

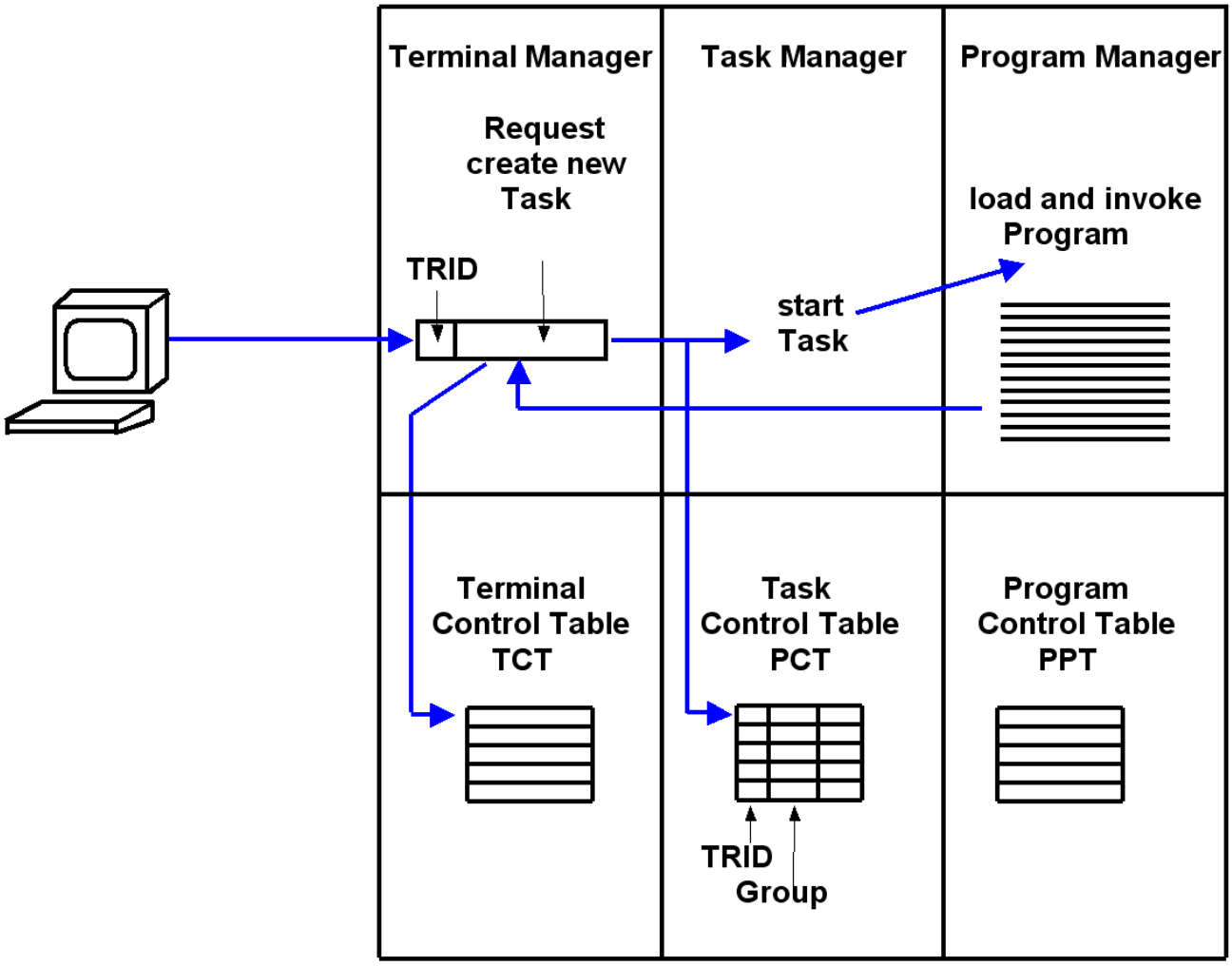


Abb. 8.4.8
Zusammenspiel von Terminal Manager, Task Manager und Program Manager

Task Manager legt für jede TRID, welche in das CICS System eintritt, einen Control Block, (Task Control Area, TCA) als Teil des Task Control Table (PCT) an. Die TCA enthält einen Pointer auf den PCT Eintrag für die TRID und den entsprechenden TCT Eintrag für das Terminal. Task Manager identifiziert an Hand der TRID, um welche von vielen möglichen Transaktionsarten es sich handelt. Für die Ausführung einer bestimmten Transaktionsart sind eine Reihe von Komponenten erforderlich, z.B. mindestens ein Anwendungsprogramm, Kenndaten für die Terminal Ein/Ausgabe wie z.B. ein Mapset (siehe Abb. 8.4.3), die TRID selbst, usw.

Bei Bedarf bittet Task Manager den Programm Manager, das in der Group definierte Anwendungsprogramm aus einer Bibliothek zu laden und startet dann die Transaktion, indem der entsprechende Task Control Block (TCB) in die Warteschlange ausführbarer Prozesse eingereiht wird.

8.4.5 VSAM bzw. Datenbank Zugriff durch ein Anwendungsprogramm

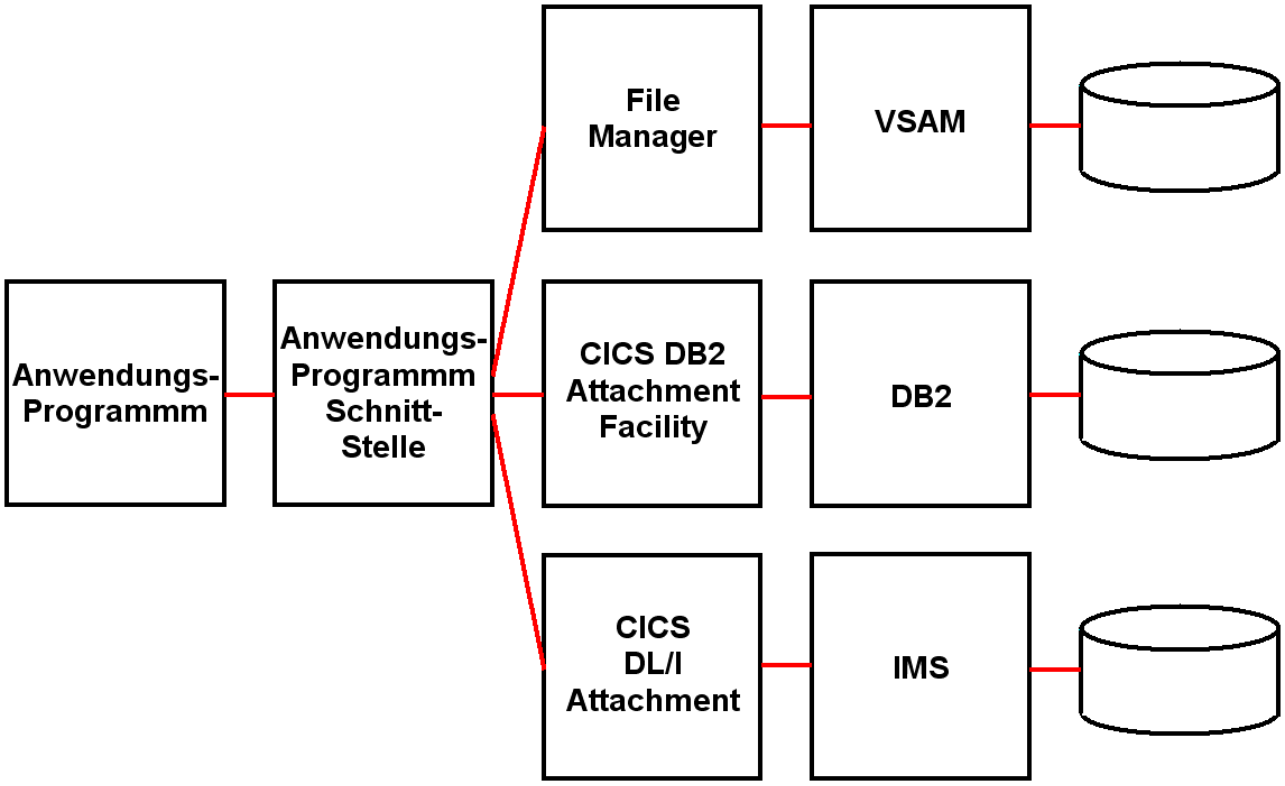


Abb. 8.4.9
Datenbank bzw. VSAM Data Set Zugriff

CICS stellt drei Funktionen für den Datenzugriff zur Verfügung, jeweils mit unterschiedlichem Funktionsumfang. Diese Funktionen müssen von dem Anwendungsprogramm über entsprechende APIs aufgerufen werden.

Ein CICS Programm greift auf DB2 über SQL Befehle zu. Mehrere CICS Module mit dem Namen CICS/DB2 Attachment Facility stellen die Verbindung von der CICS Region zu der DB2 Region her.

Als erster Schritt muss das CICS Anwendungsprogramm eine Connection (Verbindung) zu der gewünschten DB2 Datenbank herstellen. Die CONNECT-Anweisung verbindet eine Anwendung mit einem Datenbankserver. Dieser Server wird der aktuelle Server für den Prozess. Danach kann des CICS Anwendungsprogramm EXEC SQL Befehle wie SELECT, INSERT, UPDATE und DELETE enthalten. Diese EXEC SQL Befehle werden von der CICS/DB2 Attachment Facility in der CICS Region an die DB2 Region zur Verarbeitung weitergereicht. Nach der Verarbeitung gibt DB2 das Ergebnis an das CICS Anwendungsprogramm zurück.

Für IMS benutzt CICS DL/1 Befehle, für VSAM die entsprechenden EXEC CICS READ, WRITE usw. Befehle, siehe Abb. 8.1.9

8.4.6 CICS Shell

Der CICS Application Server braucht zur Bedienung eine Command Line Interface (shell).

Die einzigen Programme, die mit einem Benutzer interagieren, sind Transaktionen, die jeweils durch eine TRID gekennzeichnet sind. Die CICS Shell ist ebenfalls als eine Gruppe von CICS internen Transaktionen implementiert.

Diese CICS internen Shell Transaktionen sind, wie jede andere Transaktion, durch eine 4 Zeichen TRID gekennzeichnet, und werden über diese aufgerufen. Im Gegensatz zu normalen Transaktionen werden sie aber nicht von einem Anwendungsprogrammierer entwickelt, sondern sind schon vorhanden, wenn CICS erstmalig installiert wird.

Die wichtigsten TRIDs dieser Shell Transaktionen sind CEDA, CEDB und CEDC, sowie CESF und CESN für login und logoff. Zu CICS gehören viele weitere interne Transaktionen

Dies ist eine Liste der CICS internen Transaktionen:

CDBC	DBCTL IMS Database Control Menu
CDBI	IMS Database Control Inquiry
CDBM	Database Control Interface
CEBR	Temporary storage browse
CECI	Command-level interpreter
CECS	Command-level syntax checker
CEDA	Dynamic addition of various tables
CEDB	Update CICS CSD data set
CEDC	Interrogate CICS CSD data set
CEDF	Execution diagnostic facility (EDF)
CEMT	All master terminal functions
CEOT	Terminal status
CESF	Signoff (previously CSSF)
CESN	Signon (previously CSSN)
CEST	Inquire or set terminals, lines, control units or tasks
CETR	Trace control facility
CIND	In-doubt Testing Tool
CMAC	Display messages and codes
CMSG	Message switching
CRTE	Route transaction to control units or tasks
CSFE	Terminal test function, trace control and storage freeze
CSPG	Terminal paging
CWTO	Write to operator

Die Mehrzahl dieser Transaktionen werden Sie niemals benutzen

Der CICS System Definition (CSD) Dataset ist eine VSAM KSDS File, welche CICS Resource Definitionen speichert.

8.5 Weiterführende Information

Eine CICS Übersicht finden Sie hier

http://www.cedix.de/VorlesMirror/Band1/IntroandOverview.pdf

und hier ein 140 Seiten Lehrbuch

http://de.scribd.com/doc/48888220/CICS

Hier wird ein Logon zu CICS gezeigt

http://www.youtube.com/watch?v=OFjpRnmMKQ0

Murach's CICS for Cobol isst ein Standard Textbook. Verfügbar unter
http://tc3.hccs.edu/itse1402/Murach-CICS-for-COBOL-programmer.pdf

Das folgende Video enthält ein Video der IBM Vertriebsorganisation, welches die Vorzüge von CICS anpreist. Interessant ist es vor allem, weil es vor dem Hintergrund des IBM Entwicklungslabors in Hursley (Südengland) gedreht wurde. Das Labor ist in einem landschaftlich wunderschön gelegenen Schloss untergebracht, welches im 19. Jahrhundert von einem englischen Adeligen errichtet wurde.

http://www.youtube.com/watch?v=8KoyvfieQZ8

Zum Thema Cobol

CICS Anwendungen können in vielen Programmiersprachen entwickelt werde, z.B. C++, Java, REXX, Ada, und andere. Die am häufigsten eingesetzte Programmiersprache ist aber nach wie vor Cobol,

Eine Einführungen in Cobol ist zu finden unter

http://www.csis.ul.ie/cobol/course/Default.htm

und

http://www.csis.ul.ie/cobol/

Information über die Zukunft der Programmiersprache Cobol finden sie unter

http://www.cedix.de/VorlesMirror/Band1/TestOfTime.pdf

Die Programmiersprache Cobol hat den Vorteil der besonders effektiven Darstellung von Zahlen

http://www.youtube.com/watch?v=XvA9J2oL4qM

oder

http://cedix.de/VorlesMirror/Band1/Numeric.pdf

Cobol Software hat die Reputation, besonders wartungsfreudig zu sein, deutlich besser als alle anderen Programmiersprachen. Siehe hierzu heise online > News > 2011 > KW 49 > :

09.12.2011 17:25 « Vorige | Nächste »

Unterhalt von Java-Software teurer als von Cobol-Programmen

vorlesen / MP3-Download

Zum zweiten Mal hat die US-Firma CAST Software Unternehmenssoftware unter anderem in Hinblick auf Sicherheit, Leistung und Wartbarkeit untersucht. Dabei kamen 745 Anwendungen mit 365 Millionen Zeilen Code aus 160 Organisationen auf den Prüfstand.

J2EE-Anwendungen machen mit knapp 46 Prozent den Löwenanteil der untersuchten Programme aus, gefolgt von 16 Prozent, die verschiedene Techniken mischen. Auf den dritten Platz kommt Cobol-Software mit 11 Prozent. Vertreten sind außerdem die Kategorien .NET, ABAP, C, C++, Oracle Forms, Oracle CRM/ERP und Visual Basic.

Bei der statischen Code-Analyse mit dem hauseigenen Werkzeug "Application Intelligence Platform" ermittelte CAST eine als "technische Schuld" (technical debt) bezeichnete Größe zum Messen der Code-Qualität. Sie beschreibe "die Kosten für die Behebung jener Probleme in einer Anwendung, die das Unternehmen einem ernsthaften Risiko aussetzen."

Am höchsten liegen, so CAST, die durchschnittlichen Schulden bei J2EE-Anwendungen, während Cobol- und ABAP-Anwendungen sehr niedrige Werte haben. Dazu dürfte auch beitragen, dass Cobol-Programme im Durchschnitt wesentlich älter sind als etwa in .NET oder Java geschriebene, sodass viele Bugs bereits entdeckt und beseitigt sind.

http://www.heise.de/ix/meldung/Unterhalt-von-Java-Software-teurer-als-von-Cobol-Programmen-1392849.html