UNIVERSITÄT LEIPZIG

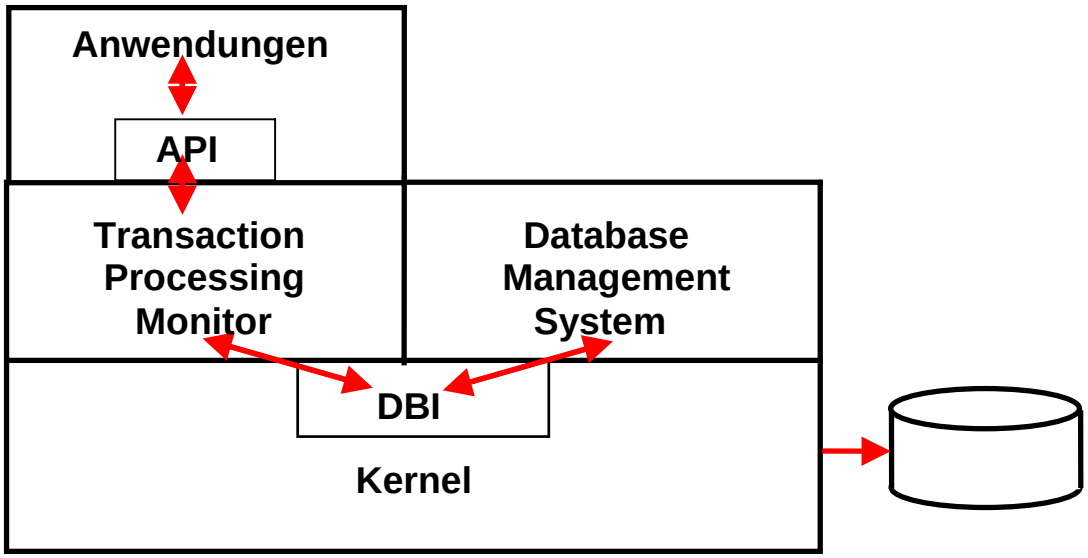
Enterprise Computing
Einführung in das Betriebssystem z/OS

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

WS2012/2013

CICS Transaktionsserver Teil 1

CICS Übersicht

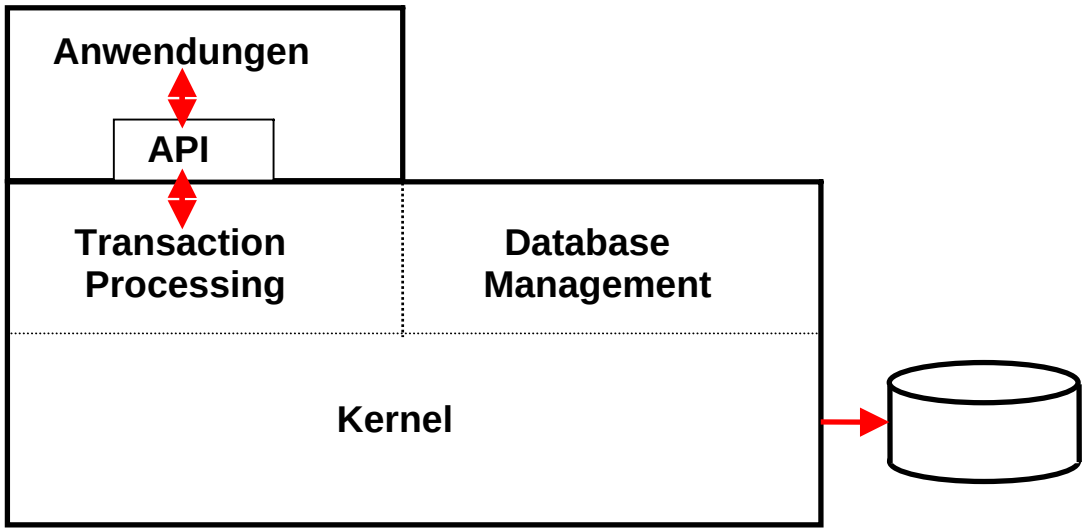


Implementierung eines Transaktionsmonitors

Unter z/OS laufen der Transaction Processing Monitor (TP Monitor) 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.



Betriebssystem-Schnittstelle

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.

Transaction Processing Facility
TPF

TPF *(Transaction Processing Facility)* ist ein eigenständiges Mainframe Betriebssystem, das nur eine einzige Anwendung erlaubt, nämlich die Transaktionsverarbeitung. Bei TPF laufen Anwendungen, Transaktion-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*-Flugplatzreservierungssystem hervorgegangen, das 1959 gemeinsam von American Airlines und IBM entwickelt wurde. Später ist es in ACP und dann in TPF umbenannt worden.

Marriot Hotels

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

CICS
Customer Information Control System

CICS ist der am weitesten verbreitete, IBM proprietäre Transaktionsmonitor.

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 (2010) ist die Version CICS TS 4.1 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.informatik.uni-leipzig.de/cs/Literature/Textbooks/CicsIntro.pdf

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.

Investition in Anwendungen

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.

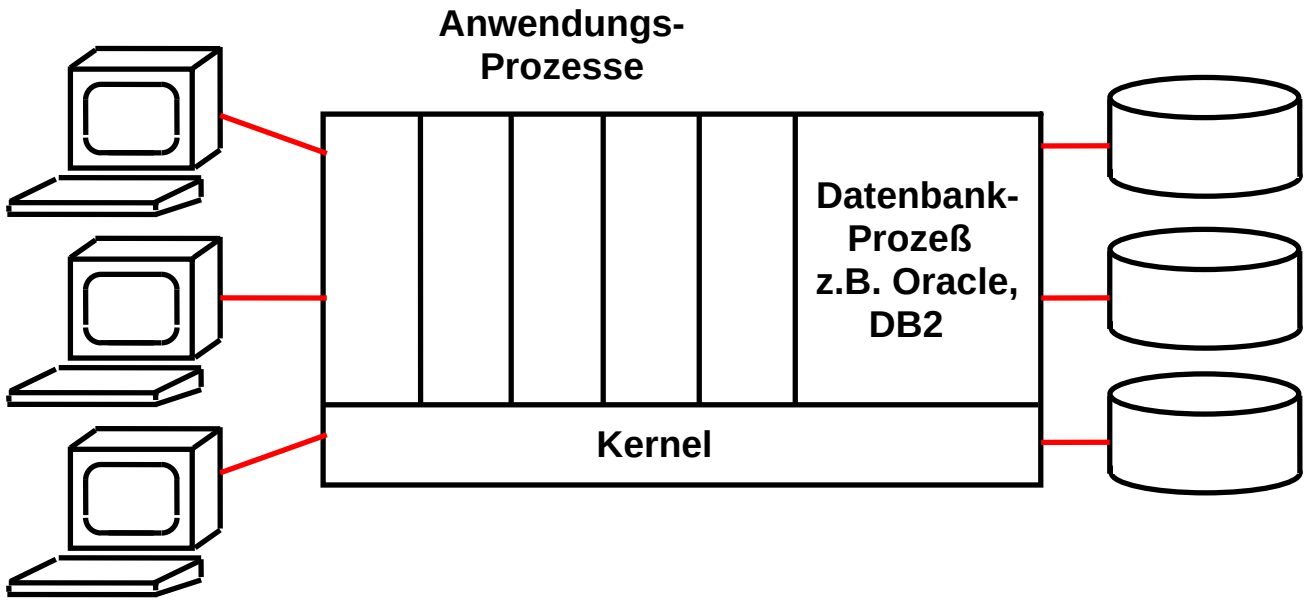
In der Weltwirtschaft sind weder das Geld und erst recht nicht die erforderliche Anzahl von Programmierern vorhanden um existierende Anwendungen ohne Not umzuschreiben. Vorhandene Ressourcen werden dringend für anstehende Aufgaben benötigt.

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

There are approximately 950 000 CICS programmers worldwide.

Mary E. Shacklett: The Importance of building Mainframe Talent. Mainframe Executive, Jan./Feb. 2012, p.33.
http://www.unicomglobal.com/files/9513/2675/4104/Mainframe_Executive_Jan_Feb12_v2.pdf

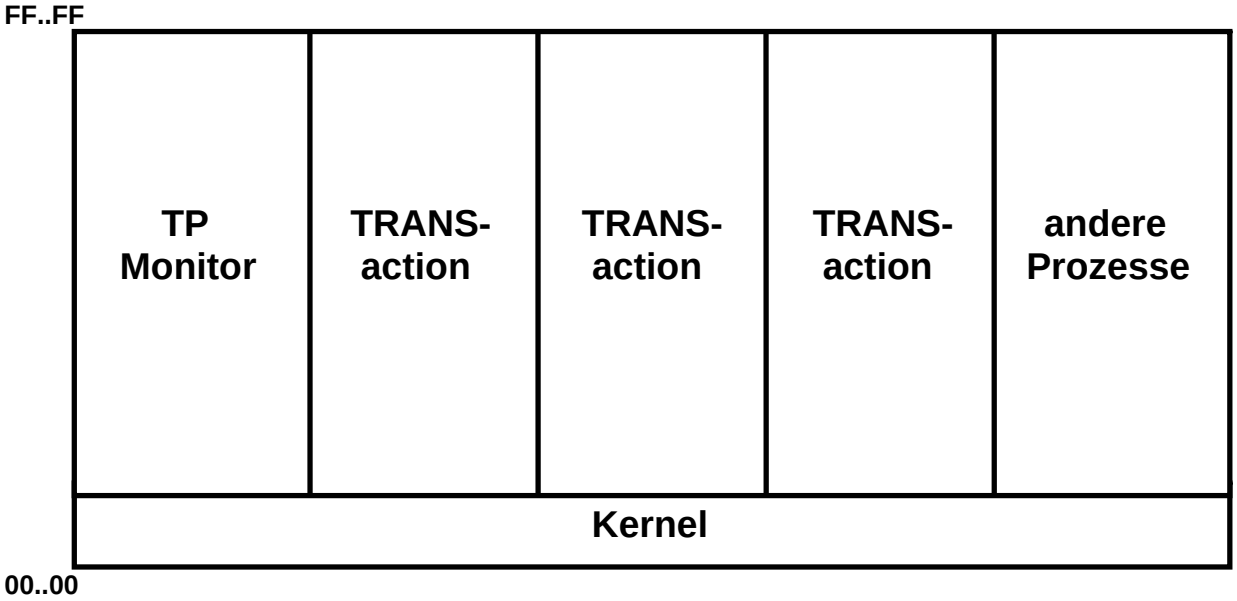
http://www-3.ibm.com/developer/solutionsevent/pdfs/spector_lunchtime_keynote.pdf



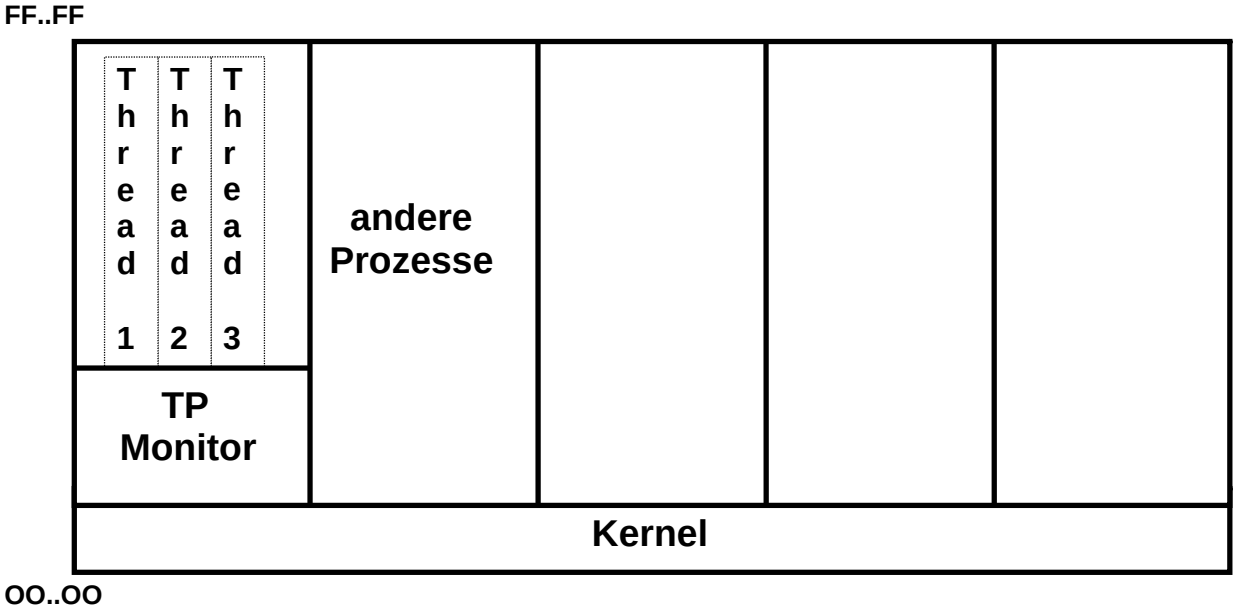
Multiprogrammierte Verarbeitung von Transaktionen

Ein Hochleistungs-Transaktions-system 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.



Prozess-Ansatz



Thread-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.

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 Adressenraumläuft. 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 Transaktions-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.

Implementierung des CICS Transaktionsmonitors

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**

CICS Transaktionsmonitor

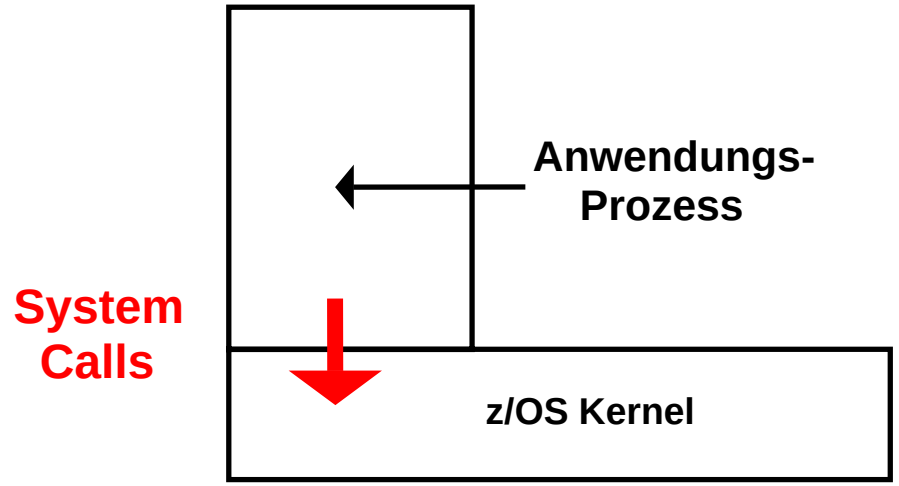
Bei allen transaktionskritischen Aktivitäten rufen CICS Anwendungsprogramme den Betriebssystem-Kernel nie direkt auf. Stattdessen führt ein CICS Anwendungsprogramm spezielle *EXEC CICS* Kommandos aus, um Betriebssystem-Funktionen zu bewirken wie:

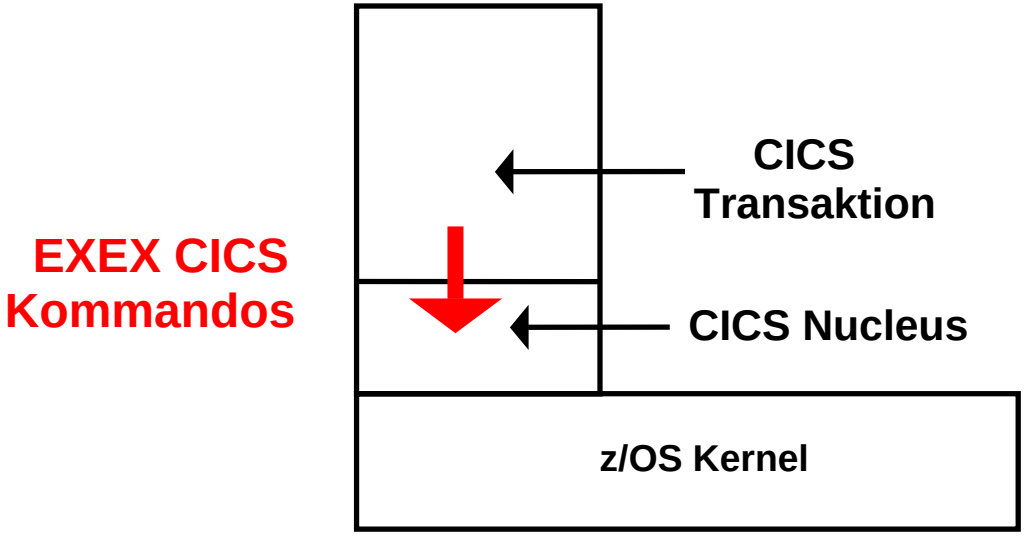
- **Dateizugriffe,**
- **Klienten-Kommunikation**
- **Prozesswechsel oder**
- **Funktionen, die System-Ressourcen erfordern.**

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.



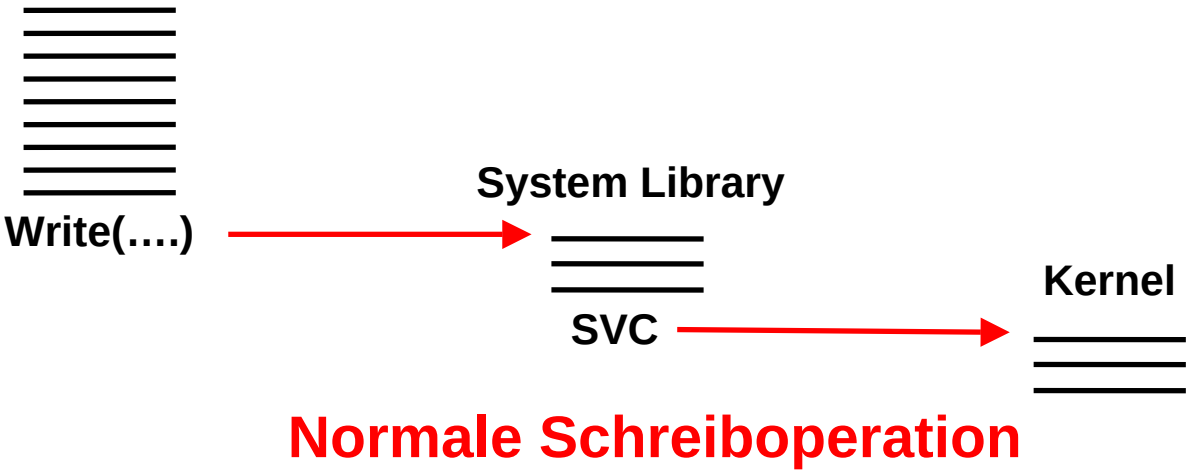


Ein normaler Anwendungsprozess nimmt Dienstleistungen des Kernels über System Calls wie z.B. OPEN oder READ in Anspruch.

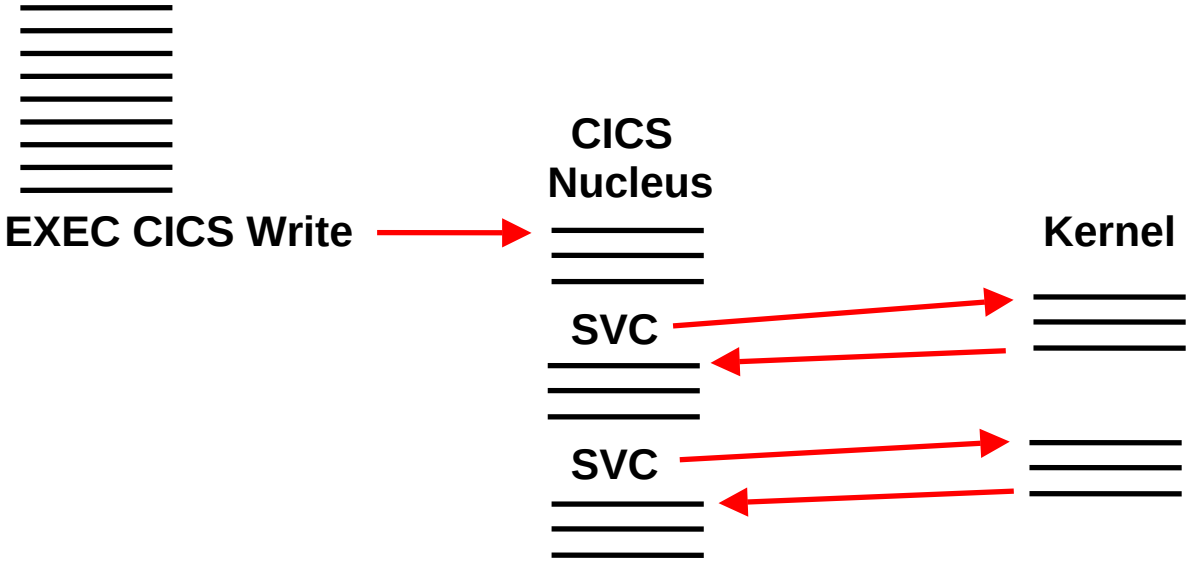
Eine CICS Transaktion verwendet statt dessen EXEC CICS Kommandos, die Dienstleistungen einer speziellen CICS Komponente (CICS Nucleus) in Anspruch nehmen.

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

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



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 diese 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.

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

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
RETREAVE

Systemsteuerung

ADDRESS
ASSIGN

VSAMf

READ
WRITE
UNLOCK
DELETE
STARTBR
READNEXT
READPREV
RESETBR
ENDBR

Hauptspeichersteuerung

GETMAIN
FREEMAIN

Andere

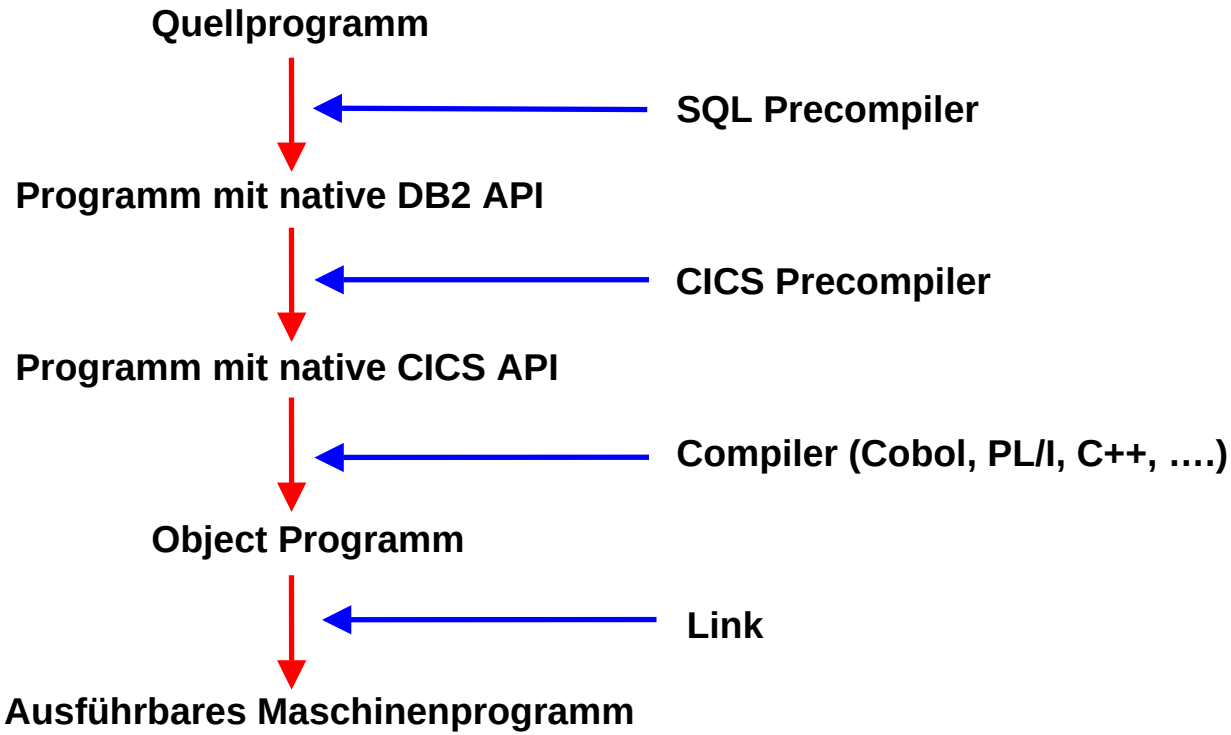
END DEQ
SUSPEND
WRITEJOURNALNUM
HANDLE CONDITION
IGNORE CONDITION

```
#include <memory.h>
#include <stdlib.h>
#include <//'PRAKT20.LIB(SET5020)'>

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;

}
```

Beispiel für Embedded SQL

Der hier wiedergegebene Code in C++ ist Bestandteil einer Übung, welche Sie in einer der nächsten praktischen Übungen entwickeln werden. Er 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.



Erstellen einer CICS - DB2 Anwendung

Das oben 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.