Wilhelm G. Spruth, Erhard Rahm

Sysplex-Cluster-Technologien für Hochleistungs-Datenbanken

Wir stellen die Cluster-Architektur IBM Parallel Sysplex und ihren Einsatz zur Datenbank- und Transaktionsverarbeitung vor. Die Sysplex-Architektur ermöglicht die Nutzung von bis zu 32 Mehrprozessor-Großrechnern auf einem gemeinsamen Datenbestand, ohne Modifikation bestehender Anwendungen. Eine wesentliche Komponente ist die so genannte Coupling Facility (CF), in der allen Rechnern zugängliche globale Datenstrukturen und globale Pufferbereiche verwaltet werden. Wir diskutieren, wie mit einer solchen »nahen«Rechnerkopplung leistungskritische Cluster-Aufgaben zur Synchronisation und Kohärenzkontrolle gelöst werden können. Leistungsuntersuchungen zeigen eine hohe Skalierbarkeit der Sysplex-Performance in praktischen Einsatzfällen.

1 Einführung

Transaktions- und Datenbankanwendungen in modernen Großunternehmen erfordern Rechenleistungen, die von einem einzelnen Rechner nicht mehr erbracht werden können. Kommerzielle Großrechner werden deshalb seit einiger Zeit im Rahmen von Clustern eingesetzt. Internetanwendungen und Web-Services verstärken diesen Trend, weil der Abstand zwischen Durchschnitts- und Spitzenbelastung ständig größer wird.

Cluster leiden allgemein unter Skalierungsschwierigkeiten, insbesondere aufgrund von Synchronisationsproblemen beim Zugriff auf gemeinsam genutzte Daten. Der IBM OS/390- bzw. z/OS-Sysplex verfügt über eine Reihe neuartiger technologischer Ansätze, die eine problemlose Skalierung bis zu über einhundert CPUs ermöglichen. In dem vorliegenden Beitrag werden einige dieser Technologien erläutert.

Großrechner, besonders auch solche der S/390-Architektur, sind schon immer führend beim Einsatz neuer Technologien gewesen. Eigenschaften wie z.B. SCSI, Caches, Direct Memory Access (DMA), Pipelining, Branch Prediction, virtuelle Speicher, Speicherschutz, Multitasking, Multithreading, Netzwerkfähigkeit und Multiprozessorfähigkeit (SMP) waren lange Großrechnern vorbehalten und diffundierten nur langsam in die PC-Welt. Welche technologischen Eigenschaften wird ein PC, ein PDA oder ein Mobiltelefon von übermorgen haben? Mit hoher Wahrscheinlichkeit werden es wiederum Eigenschaften sein, die wir schon heute in Großrechnern vorfinden, wie Cluster-Technologien und die Eigenschaften des Sysplex.

Im nächsten Abschnitt führen wir zunächst wesentliche Architekturmerkmale von Clustern ein und diskutieren Alternativen bezüglich der Interprozessor-Kommunikation und Externspeicheranbindung. Abschnitt 3 gibt einen Überblick zu IBM Parallel Sysplex sowie den darin enthaltenen spezifischen Cluster-Komponenten wie der so genannten Coupling Facility (CF). Abschnitt 4 diskutiert den Einsatz der CF zur globalen Sperrvergabe sowie zur Kohärenzkontrolle, zwei leistungskritischen Funktionen für Cluster. Abschließend werden einige Messergebnisse vorgestellt, welche eine hohe Skalierbarkeit des Parallel Sysplex ausweisen.

2 Cluster-Architekturen

2.1 Taxonomie

Die meisten kommerziellen Großrechner-Installationen basieren auf Mehrprozessorsystemen / Parallelrechnern bzw. Clustern. Kennzeichnend ist der Einsatz mehrerer Prozessoren in unmittelbarer räumlicher Nachbarschaft (typischerweise ein Maschinenraum) mit sehr schnellen Interprozessor-Verbindungen. Die Einteilung der wichtigsten Cluster-Arten erfolgt aufgrund von zwei Kriterien: die Art der Prozessor/Rechner-Kopplung sowie der Externspeicherzuordnung.

Bezüglich der Prozessor/Rechner-Kopplung unterscheiden wir die drei in Abbildung 2.1 gezeigten Varianten, nämlich enge, lose und nahe Kopplung. Bei *eng gekoppelten* Parallelrechnern – auch als Symmetrischer Multiprozessor (SMP) bezeichnet – greifen mehrere CPUs auf einen gemeinsamen Hauptspeicher zu, der eine einzige Kopie (Instanz) des Betriebssystems enthält. Bei einem *lose gekoppelten* Parallelrechner verfügt jede CPU über ihren eigenen Hauptspeicher und eine eigene Kopie des Betriebssystems. Eine Mischform ist die *nahe Kopplung* (»closely coupled system«). Dies ist im Wesentlichen ein lose gekoppelter Parallelrechner, bei dem allen CPUs ein zusätzlicher, gemeinsam genutzter Speicher (Globaler Speicher) zur Verfügung steht, der in der Regel von einem eigenen Rechner verwaltet wird. Auf den globalen Speicher und die dort verwalteten Daten bzw. globalen Datenstrukturen wird ein sehr schneller Zugriff unterstützt (z.B. über eigene Maschinenbefehle), wodurch sich gegenüber der losen Kopplung aufwendige Kommunikationsvorgänge über allgemeine nachrichtenbasierte Protokolle einsparen lassen.

Die Verbindung der Rechner untereinander erfolgt durch ein Hochgeschwindigkeitsnetzwerk. Dieses wird in der Regel als Crossbar Switch implementiert. Die Datenübertragungsrate derartiger Switches liegt typischerweise im Bereich zwischen zehn und 100 GByte/s. Die Parallelschaltung mehrerer Switches ist möglich.

Basierend auf den vorgestellten Arten der Prozessorkopplung unterscheidet

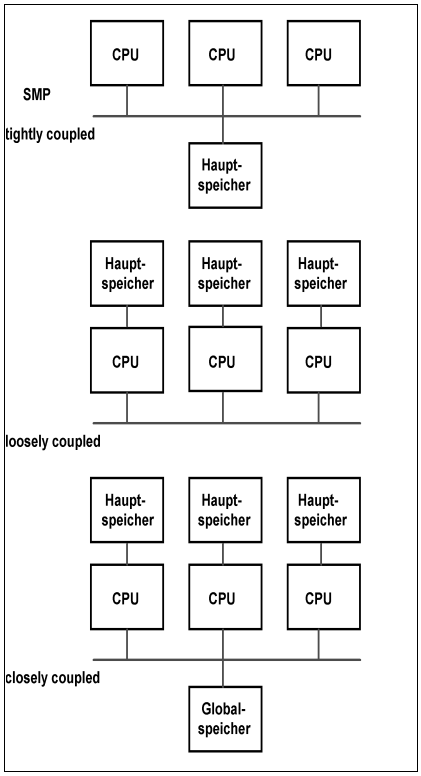

Abb. 2.1: Arten der Prozessorkopplung

man unter Berücksichtigung der Art des Plattenspeicherzugriffs im Wesentlichen drei Arten von Cluster-Architekturen: Shared Everything, Shared Nothing, Shared Disk [RAHM94]. Der einfachste Ansatz ist das *Shared-Everything*-Modell. Es wird durch einen symmetrischen Multiprocessor (SMP) implementiert, basiert also auf der engen Kopplung. Da alle Plattenspeicherdaten im einzigen Hauptspeicher des SMP zwischengespeichert werden, erfolgt ein gemeinsamer Platten-/Datenzugriff für alle Prozessoren. Synchronisationsaufgaben können über den gemeinsamen Hauptspeicher und das von allen Prozessoren gemeinsam genutzte Betriebssystem und Datenbanksystem einfach gelöst werden. Das Hauptproblem ist, dass SMPs selbst nur bis zu einer begrenzten maximalen Anzahl von CPUs skalieren. Diese Grenze liegt für normale Datenbankanwendungen beim derzeitigen Stand der Technik bei 16 CPUs.

Von Clustern im engeren Sinne spricht man daher nur bei lose oder nah gekoppelten Ansätzen nach dem Shared-Nothing- oder Shared-Disk-Modell. Üblicherweise nutzen diese Cluster als Knoten (»nodes«) SMPs, so dass sie genau genommen hybride Architekturen repräsentieren. In Abbildung 2.2 ist ein solches Cluster mit 4 Knoten und loser Kopplung dargestellt.

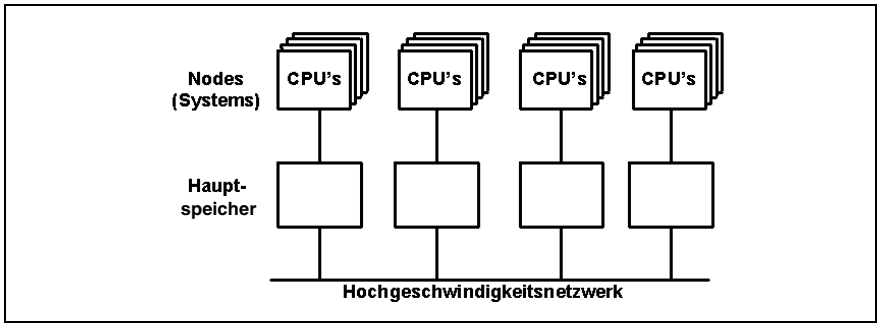

Abb. 2.2: Cluster mit 4 Knoten

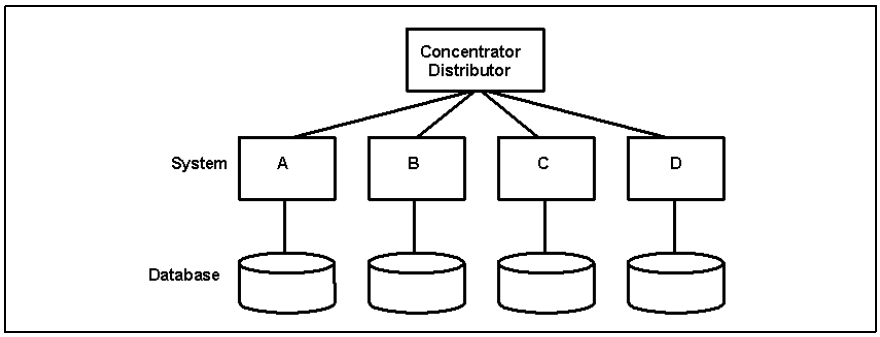

Abb. 2.3: Shared-Nothing-Modell (partitionierte Daten)

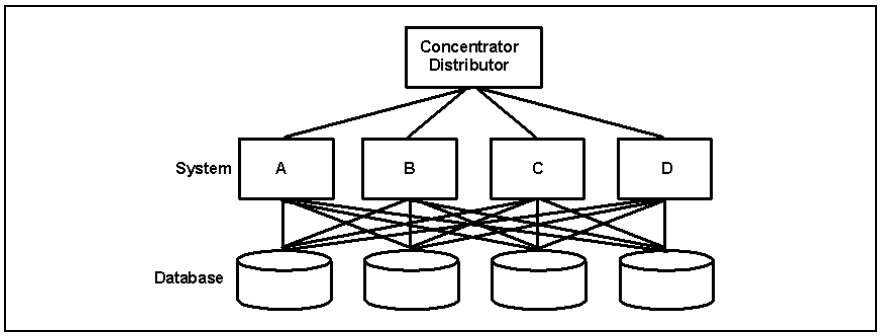

Abb. 2.4: Shared-Disk-Modell (gemeinsame Daten)

Beim *Shared-Nothing*-Modell besitzt jeder Knoten einen Teil der Daten(bank), der auf Plattenspeichern untergebracht ist, auf die nur der betreffende Knoten zugreifen kann, siehe Abbildung 2.3. Die Arbeitslast wird den einzelnen Rechnern häufig statisch zugeordnet. Durch die Daten-Partitionierung ist jedes System in der Lage, auf seine Daten direkt zuzugreifen, ohne Synchronisation mit anderen Knoten. Ein derartiges Modell kann eine gute Skalierbarkeit liefern, solange eine saubere Aufteilung der gesamten Datenbank auf die einzelnen Knoten und deren Plattenspeicher möglich ist. Echtzeit-Workload-Schwankungen können die Prozessor-Ressourcen über- oder unterfordern. Letzteres führt zu einer Leistungsverschlechterung des Clusters.

Das *Shared-Disk*-Modell sieht vor, dass jedes System auf die vollständige Datenmenge, die über die Plattenspeicher des Clusters verteilt ist, zugreifen kann, siehe Abbildung 2.4. Der Vorteil dieses Modells liegt in der Möglichkeit, die Arbeitslast dynamisch über alle Knoten des Clusters zu verteilen. Als Hauptnachteil ergibt sich eine potenziell schlechte Skalierbarkeit aufgrund des zur globalen Synchronisierung des Datenzugriffs erforderlichen Aufwands. Hinzu kommt die Notwendigkeit einer Kohärenzkontrolle, da dieselben Daten in jedem Knoten im Hauptspeicher gepuffert und modifiziert werden können, wodurch Kopien der betreffenden Daten in anderen Knoten ungültig werden.

Bei naher Kopplung lassen sich die Aufgaben der Synchronisation und der Kohärenzkontrolle über den globalen Speicher sehr effizient lösen und somit die Skalierbarkeit entscheidend verbessern. Dies ist der wesentliche Schlüssel zum Erfolg der Sysplex-Architektur. Zur Abgrenzung von lose gekoppelten Shared-Disk-Konfigurationen spricht man bei der nahen Kopplung auch gelegentlich von einer *Shared-Data-Architektur*, da der gemeinsame Datenzugriff nicht nur über die Platten, sondern auch über den globalen Speicher unterstützt werden kann (z.B. durch die dortige Pufferung von Datenbankseiten).

2.2 Cluster-Implementierungen

Eine ganze Reihe von Herstellern liefern Cluster für die Transaktionsverarbeitung und als Datenbankrechner. Bevor wir ab dem nächsten Abschnitt näher auf die Sysplex-Architektur eingehen, sollen noch kurz zwei andere wichtige Hardwareplattformen angesprochen werden, nämlich Sun Fire 15K Cluster und HP Superdome Cluster.

Der Sun Fire 15K Cluster (Abb. 2.5) besteht aus bis zu 72 CPUs auf 18 als »System Boards« bezeichneten SMP-Knoten mit je 4 UltraSPARC III 900-MHz-Prozessoren (komplexere Anordnungen sind möglich, z.B. für wissenschaftliche Zwecke). Auf jedem System Board befinden sich Hauptspeichermodule sowie Anschlüsse für den zentralen Switch. Weiterhin enthält das System Board E/A-Controller, welche eine Verbindung zu getrennten E/A-Boards herstellen. Auf den E/A-Boards ist ein PCI-Bus implementiert, der SCSI-Adapterkarten für den Anschluss von Plattenspeichern aufnehmen kann. Für Glasfaserverbindungen kann dies der serielle Fibre-Channel-SCSI-Anschluss sein.

Vom Prinzip her verwirklicht diese Konfiguration zunächst ein Shared-Nothing-Modell. Die Datenbank wird partitioniert und auf Gruppen von Plattenspeichern aufgeteilt, die den einzelnen Knoten fest zugeordnet sind. Wenn ein Knoten auf einen Plattenspeicher zugreifen will, der an einen anderen Knoten angeschlossen ist, muss dies über eine (aufwendige) Kommunikation zwischen den betroffenen Knoten abgewickelt werden. Um dies zu vermeiden, kann man alle Plattenspeicher über einen weiteren Switch mit allen Knoten verbinden, wie vom HP Superdome Cluster unterstützt.

Der HP Superdome Cluster (Abb. 2.6) besteht aus 16 als »Cell Boards« bezeichneten SMP-Knoten mit je 4 CPUs, also insgesamt 64 CPUs. Ähnlich wie beim Sun Fire 15K Cluster befinden sich Hauptspeicher, E/A-Controller und Anschlüsse für den zentralen Switch auf jedem Cell Board. Der zusätzliche Fibre Channel Switch ermöglicht es jedem Knoten, auf jeden Plattenspeicher zuzugreifen (Shared-Disk-Funktionalität).

Der in Abbildung 2.6 gezeigte Fibre Channel Switch wird entweder durch die angeschlossenen Knoten gesteuert oder er wird als Teil eines Storage Servers und eines Storage Area Networks (SAN) implementiert. Der Storage Server enthält einen eigenen Rechner und realisiert z.B. einen dynamischen Plattenspeicher-Cache, Plattensteuerung sowie RAID- und Netzwerk-Optimierungen. Beispiele für derartige Storage-Server-Produkte sind Symmetrix von EMC, Lightning von Hitachi und Shark von IBM.

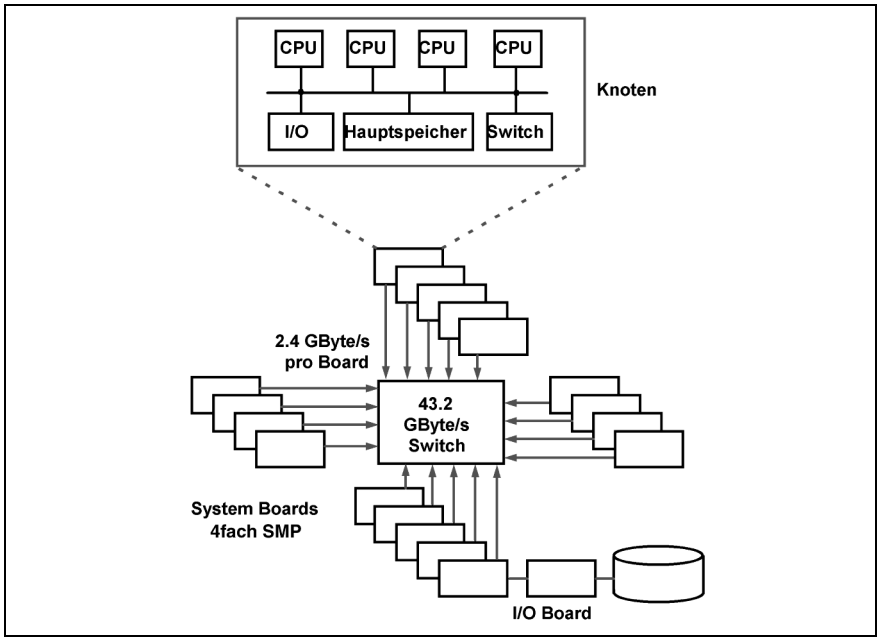

Abb. 2.5: Sun Fire 15K Cluster

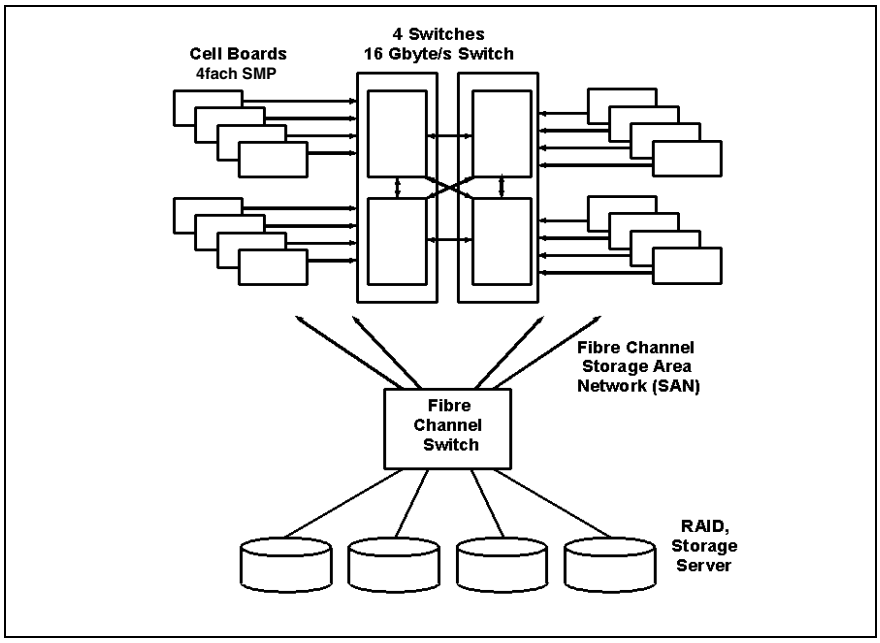

Abb. 2.6: HP Superdome Cluster

3 Parallel Sysplex

Der Sysplex (*Sys*tem Com*plex*) wurde von IBM 1990 eingeführt; die hier diskutierte Cluster-Variante Parallel Sysplex kam 1994 als Bestandteil der S/390- bzw. zSeries-Architektur auf den Markt. IBM bezeichnet die Hardware als S/390 oder zSeries und das Betriebssystem als OS/390 oder z/OS. zSeries und z/OS weisen gegenüber S/390 und OS/390 eine zusätzliche 64-Bit-Unterstützung auf; davon abgesehen sind die Unterschiede nicht so groß. Die derzeitige zSeries-Implementierung wird als z900 bezeichnet, eine kleinere Variante z800 wurde kürzlich angekündigt.

Abbildung 3.1 zeigt den Aufbau der Parallel-Sysplex-Architektur. Während die in Abschnitt 2.2 vorgestellten Cluster von Sun und HP eine lose Kopplung verwenden, handelt es sich beim Parallel Sysplex um eine nah gekoppelte Cluster-Architektur nach dem Shared-Disk- bzw. Shared-Data-Modell. Der Sysplex besteht aus den Prozessorknoten (»Syste-

me« genannt), gemeinsamen Plattenspeichern (Shared Storage Devices), Netzwerk-Controllern und den Kern-Cluster-Technologie-Komponenten. Letztere umfassen den (die) als »FICON Director« bezeichneten Switch(es), den Sysplex Timer und die »Coupling Facility« (CF). Die Coupling Facility enthält den für die nahe Kopplung charakteristischen globalen Speicher zur Realisierung globaler Kontrollaufgaben und ist von allen Knoten schnell zugreifbar. Es werden bis zu 32 Knoten unterstützt. Jeder Knoten stellt einen SMP dar und enthält maximal 16 CPUs, insgesamt also maximal 512 CPUs. Hinzu kommen noch 3 Ein-/Ausgabeprozessoren (I/O-Off-Load-Prozessoren, IOPs) pro Knoten. Derzeitige Installationen haben bis zu 100 CPUs. Die Knoten müssen nicht homogen sein, d.h., ein Knoten kann aus einem z900-Rechner und ein anderer aus einem älteren S/390-Rechner bestehen.

Zu den Sysplex-Komponenten werden spezifische Maschinenbefehle sowie Betriebssystemdienste zur Verfügung gestellt, mit denen eine effiziente Durchführung der Cluster-Aufgaben für alle Knoten erreicht wird, insbesondere zu Kommunikation, E/A sowie für globale Steuerungsaufgaben wie Synchronisation, Kohärenzkontrolle und Lastbalancierung. Diese Dienste werden in allgemeiner Form realisiert und von unterschiedlichen Software-Subsystemen, vor allem Datenbanksystemen, für den Cluster-Einsatz verwendet. IBM hat dazu die wichtigsten Subsysteme an das Arbeiten innerhalb eines Sysplex angepasst. Eine Anpassung der Anwendungen an den Sysplex ist in der Regel nicht erforderlich und i.Allg. auch nicht möglich. Sysplex-Unterstützung ist für folgende OS/390-Subsysteme verfügbar:

- Datenbanksystem IMS
- Datenbanksystem DB2
- Dateisystem VSAM
- Transaktionsmonitor CICS (CICSplex)
- Transaktionsmonitor IMS TM
- Web-Applikationsserver WebSphere
- Communication Server
- SAP System R/3

Wir stellen im Folgenden die wesentlichen Cluster-Komponenten FICON-Switch, Sysplex Timer und die Coupling Facility näher vor.

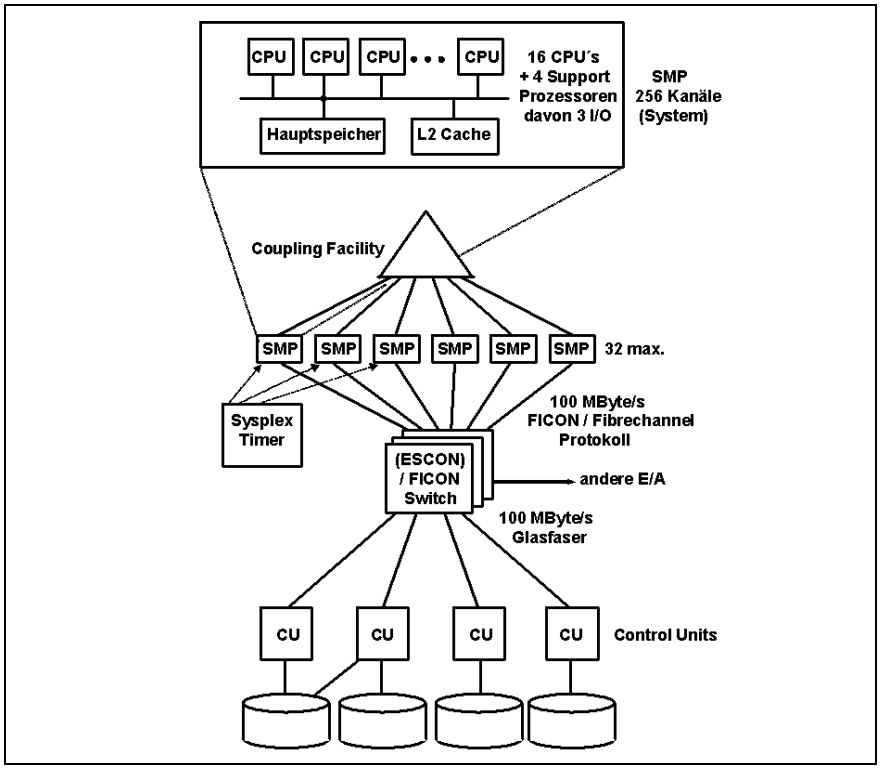

Abb. 3.1: Parallel Sysplex

3.1 FICON-Architektur für Kommunikation und E/A

Jeder Knoten kann mit bis zu 256 »Kanälen« mit der Außenwelt verbunden werden, mit bis zu 100 Mbyte/s pro Verbindung. Die Verbindung der Systeme sowohl untereinander als auch zu den Plattengeräten erfolgt über ein oder mehrere FICON (FIbre CONnection)-Switches. Da jedes System 256 Ein-/Ausgabekanäle anbinden kann, hat diese Verbindungsstruktur die Aufgabe, bei 32 Systemen maximal $256 * 32 = 2^{13}$ Kanäle zu verwalten. Damit ist jeder Knoten in der Lage, auf alle Plattenspeicher und alle anderen Knoten des Sysplex direkt zuzugreifen (Shared-Disk-Modell). Die Steuereinheiten der Plattenspeicher (Control Units) implementieren eine Storage-Server- und SAN-Funktionalität.

Die von IBM entwickelte FICON-E/A-Architektur basiert auf dem Kanal-Subsystem (channel subsystem) der ES/9000-E/A-Architektur. Diese integriert E/A-Prozessoren (IOPs), Kanäle sowie die »Staging Hardware«. Die IOPs sorgen für die Kommunikation zwischen den Knoten und den Kanälen. Diese führen zur Datenübertragung ein Kanalprogramm aus, wobei über die Steuereinheiten (Control Units) auf die Plattenspeicher zugegriffen wird. Die Staging-Hardware stellt die Kommunikationspfade zwischen den IOPs, den Kanälen und dem Rest des Systems zur Verfügung.

In der FICON-Architektur bildet der FICON-Switch (FICON Director) die Kerneinheit. Er implementiert eine Switched-Point-to-Point-Topologie für S/390-E/A-Kanäle und Steuereinheiten. Ein FICON-Switch kann bis zu 60 Kanäle und Steuereinheiten dynamisch und nicht blockierend (Crossbar Switch) über ihre Ports miteinander verschalten. Normalerweise werden eine Reihe von FICON-Switches parallel genutzt. Entfernungen von bis zu 3 km für optische Übertragungen sind möglich. Die zulässigen Entfernungen erhöhen sich beim Einsatz einer so genannten Extended-Distance-Laser-Link-Einrichtung auf 20, 40 oder 60 km. Dies ist für Unternehmen wichtig, die aus Katastrophenschutzgründen zwei getrennte Rechenzentren betreiben.

Die Kommunikation zwischen den Knoten und den Plattenspeichern erfolgt über das FICON-Protokoll, der seriellen Version des S/390-Kanalprotokolls. Zur Kommunikation der Knoten untereinander wird das so genannte »Channel-to-Channel« (CTC)-Protokoll eingesetzt, normalerweise in einer Full-Duplex-An-

ordnung. Hierbei betrachtet jeder sendende Knoten den Empfänger als eine Ein-/Ausgabeeinheit. Der Empfänger simuliert für diesen Zweck einen eigenen Typ einer S/390-Steuereinheit (Abb. 3.2). Das OS/390-Betriebssystem stellt einen Basisdienst, die »Cross System Coupling Facility« (XCF) zur Verfügung, um über eine CTC-Verbindung eine Kommunikation mit einer anderen OS/390-Instanz innerhalb eines Sysplex zu bewerkstelligen. Dieser Dienst wird wiederum von Subsystemen wie den Datenbanksystemen DB2 und IMS für den Nachrichtenaustausch eingesetzt.

3.2 Sysplex Timer

Die Anforderungen an einen Cluster bezüglich Genauigkeit und Konsistenz des Taktes in den einzelnen Teilsystemen (Knoten) werden durch einen Sysplex Timer erfüllt. Dieser stellt eine externe Zeitreferenz (ETR, External Time Reference) dar. Der Timer generiert die synchronisierte Tageszeit (Time Of Day, TOD) für alle Knoten. Von der ETR werden 3 Signale (oscillator signal, on-time signal, data signal) für die Clock-Synchronisation erzeugt und an die Knoten des Sysplex gesendet. Der Sysplex Timer kann umgekehrt auch Informationen aus einer externen Quelle erhalten (z.B. Time Code Receiver). Im Falle eines Sysplex-Timer-Ausfalls arbeiten die Knoten weiter und verwenden ETR aus einem lokalen Modul. Beim Wiederanschluss des Timers besteht je nach ETR-Parameter die Möglichkeit der Resynchronisation.

Datenbanksysteme wie DB2 und IMS nutzen den Sysplex Timer u.a. für die Kennzeichnung ihrer Log-Einträge, mit denen Änderungen an Datenbankobjekten für Recovery-Zwecke protokolliert werden. Die Log-Sätze werden zunächst in jedem System in eine separate Log-Datei geschrieben; über die synchronisierten Zeitstempel lässt sich nun ein Mischen dieser Dateien zu einer globalen Log-Datei vornehmen, in der alle Änderungen in der Cluster-weit chronologischen Reihenfolge angeführt sind. Damit können im Fehlerfall die an verschiedenen Rechnern erfolgten Änderungen eines Datenbankobjektes oder einer ganzen DB-Platte vollständig rekonstruiert werden [RAHM94].

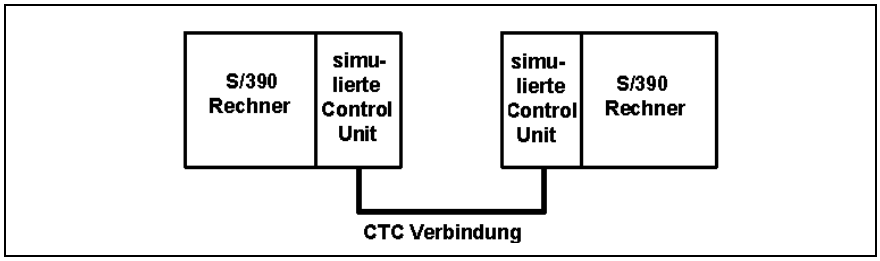

Abb. 3.2: Channel-to-Channel-Verbindung

3.3 Coupling Facility (CF)

Die wichtigste Komponente des Sysplex ist die Coupling Facility (CF). Sie setzt sich physisch aus S/390-Hardware mit eigenem Prozessor (SMP) mit einem relativ großen Hauptspeicher (mehrere Gigabytes) sowie speziellem Mikrocode (Control Code) zusammen. Die Kopplung mit anderen S/390-Prozessoren erfolgt über serielle Glasfaserverbindungen mit einer Übertragungsrate von 100 MByte/s (High Speed Coupling Links). Auf der CF läuft kein Standard-S/390-Betriebssystem, und die vielfältigen Steuerfunktionen der CF sind für den Systemprogrammierer auch nicht direkt verfügbar. Aus Leistungs- und Verfügbarkeitsgründen können innerhalb eines Sysplex zwei oder mehrere Coupling Facilities eingesetzt werden.

Jeder Knoten sowie die CF selbst enthalten eine »Coupling Support Facility«. Diese bewerkstelligt die Kommunikation und den Datenaustausch zwischen Knoten und CF. Die Coupling Support Facility besteht aus je einer Glasfaserverbindung (100 MByte/s), einem dedizierten Link-Prozessor für die Verarbeitung des (proprietären) Übertragungsprotokolls und einer Erweiterung des S/390-Maschinenbefehlssatzes. Die zusätzlichen Maschinenbefehle realisieren die nahe Kopplung zwischen Knoten und CF und ermöglichen allen Knoten den Zugriff auf Inhalte in CF-Datenstrukturen. Das Kommunikationsprotokoll wurde für eine besonders geringe Latenzzeit optimiert, welches *synchrone Zugriffe* eines jeden Knotens auf die CF ermöglicht. Synchron bedeutet, dass ein auf einem Knoten laufender Prozess während eines Zugriffs auf die Coupling Facility blockiert; es findet kein Prozesswechsel und kein Übergang »User-Status – Kernel-Status« statt. Die Zugriffsgeschwindigkeit liegt im Mikrosekundenbereich und ist somit weitaus effizienter als die Kommunikation über allgemeine Protokolle bei loser Rechnerkopplung.

Die nahe Kopplung über die CF wird für die leistungskritischen Kontrollaufgaben in Shared-Disk-Clustern genutzt, um durch die effiziente Realisierung eine hohe Skalierbarkeit zu erreichen. Dies betrifft die globale Synchronisation über Sperrverfahren (Locking), die Kohärenzkontrolle der Pufferinhalte mit schnellem Austausch geänderter Daten zwischen den Knoten sowie die flexible Lastverteilung. Entsprechend werden spezifische Funktionen (Maschinenbefehle) für Locking, Caching und Queuing sowie zugeschnittene Datenstrukturen im Hauptspeicher der CF bereitgestellt. Die CF-Hauptspeicher-Ressourcen können dynamisch partitioniert und einer der CF-Strukturen zugewiesen werden. Innerhalb derselben CF sind mehrere CF-Strukturen desselben oder unterschiedlichen Typs möglich. Die auf diesen Strukturen bereitgestellten Funktionen bzw. Cluster-Protokolle repräsentieren drei Verhaltensmodelle:

- Lock-Modell: Es unterstützt feingranulares, globales Locking für hohe Performance und eine Signalisierung von Zugriffskonflikten.
- Cache-Modell: Es liefert eine globale Kohärenz-Steuerung für die verteilten Pufferinhalte der einzelnen Knoten sowie einen globalen Puffer in der CF (Shared Data Cache).
- Queue-Modell: Es implementiert einen umfangreichen Satz an Queuing-Konstrukten für die Verteilung von Arbeitslasten, zur Realisierung einer schnellen Nachrichtenübertragung (Message Passing) sowie für die Verwaltung globaler Status-Informationen. Ein Beispiel für Letzteres ist die Verwaltung globaler Sicherheitsrechte.

Abbildung 3.3 zeigt die Anbindung der Knoten an die CF und die dort verwalteten globalen Datenstrukturen. Der Zugriff

erfolgt dabei nicht nur von den Systemen auf die CF, sondern die CF kann auch direkt (ohne Involvierung des Betriebssystems) auf bestimmte Hauptspeicherinhalte der Systeme mit Bit-Vektoren zugreifen und ändern. Solche Bit-Vektoren existieren für jede logische Verbindung zu einer CF-Datenstruktur und erlauben z.B. die schnelle Signalisierung von Zugriffskonflikten (s.u.).

Im nächsten Abschnitt betrachten wir die Nutzung der CF-Lock- und Cache-Strukturen zur Realisierung der Clusterweiten Synchronisation (Locking) und Kohärenzkontrolle.

4 Locking und Kohärenzkontrolle mit der CF

Die Datenbankverarbeitung in Shared-Disk-Konfigurationen erfordert die globale Synchronisation der Transaktionen beim Zugriff auf die allen Rechnern zugänglichen Datenbankobjekte. Hierzu wird ein globales Sperrprotokoll unter Beteiligung der DBS-Subsysteme aller Rechner benötigt. Das zweite wesentliche Shared-Disk-Problem ist die Kohärenzkontrolle, da jedes DBS-Subsystem Kopien derselben DB-Seiten in seinem Hauptspeicher puffern und ändern kann. Hierzu ist es erforderlich, veraltete Kopien zu erkennen und Transaktionen jedes Knotens stets die aktuellsten Daten zur Verfügung zu stellen. Diese Probleme werden durch Änderungstransaktionen verursacht; bei reinen Leselasten (z.B. Data-Warehouse-Analysen) gibt es weder ein Synchronisations- noch ein Kohärenzproblem.

Für die Lösung dieser Aufgaben wurde für lose und nah gekoppelte Shared-Disk-DBS eine Reihe von leistungsfähigen Protokollen entwickelt [RAHM94]. Insbesondere kann die Kohärenzkontrolle weitgehend mit der globalen Synchronisation zur Reduzierung zusätzlicher Kommunikationsvorgänge kombiniert gelöst werden. Zur Einsparung von Nachrichten zur globalen Sperrbehandlung wurden ebenfalls Techniken entwickelt, um möglichst viele Sperren rechnerlokal entscheiden zu können. Dennoch zeigt sich für viele Anwendungslasten, dass bei loser Kopplung eine hohe Kommunikationsbelastung durch die Nachrichten verursacht wird und dieser Aufwand typischerweise mit der Knotenanzahl ansteigt, so dass die Skalierbarkeit entsprechend begrenzt wird. Im Sysplex wird durch die Coupling Facility sowohl die globale Sperrbehandlung als auch die Kohärenzkontrolle durch die zentralen CF-Strukturen wesentlich unterstützt, wodurch die Voraussetzung für hohe Skalierbarkeit und Performance des Shared-Disk-Clusters geschaffen wird.

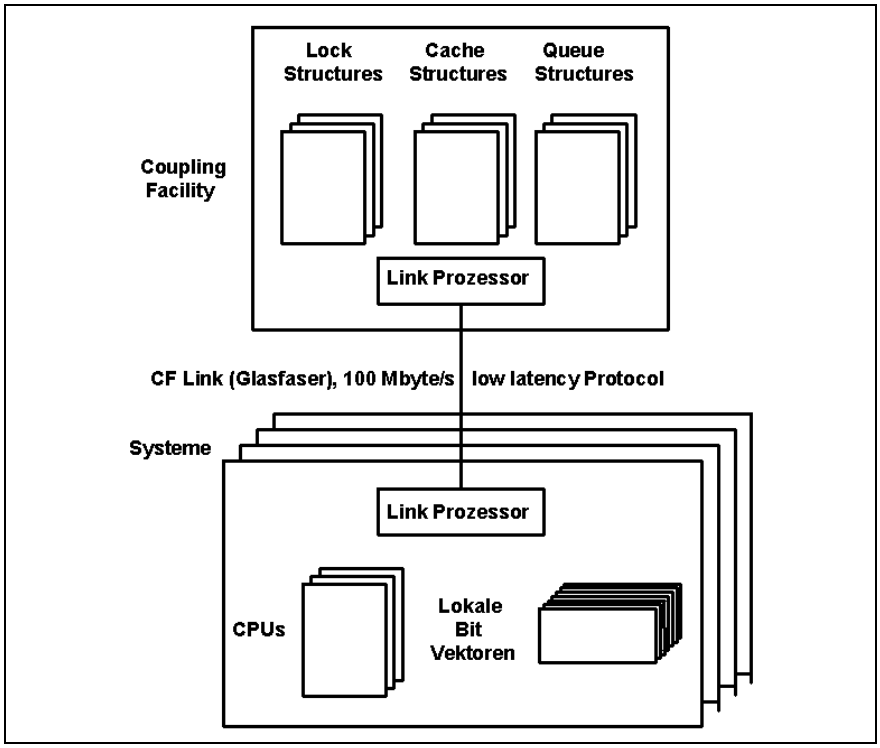

Abb. 3.3: Anbindung der Systeme an die Coupling Facility

4.1 Locking mit der CF

Die globale Sperrbehandlung im Sysplex erfolgt in Zusammenarbeit der CF, des OS/390-Betriebssystems sowie der auf jedem System laufenden DBS-Instanzen (bzw. anderen Subsystemen, die eine globale Synchronisation benötigen). Wie in Abbildung 4.1 für das Datenbanksystem DB2 gezeigt, läuft auf jedem Rechner ein lokaler Lock-Manager (LLM), der für die Sperrbehandlung aller Transaktionen und Prozesse des jeweiligen Rechners zuständig ist. In der CF-Lock-Struktur wird eine zentrale Sperrtabelle verwaltet, auf die alle LLMs zugreifen, um eine schnelle systemweite Synchronisation konkurrierender Datenzugriffe zu erreichen. Diese Zugriffe erfolgen synchron über eine als »Cross System Extension Services" (XES) bezeichnete OS/390-Komponente, die u.a. spezielle Maschinenbefehle zum Setzen und Freigeben von globalen Sperren in der CF-Sperrtabelle verwendet. Der wesentliche Vorteil dieser Konfiguration liegt darin, dass alle konfliktfreien Sperranforderungen, die typischerweise über 99% aller Sperren betreffen, über die zentrale Sperrtabelle mit minimaler Verzögerung behandelt werden können. Nur solche Sperranforderungen, für die zwischen verschiedenen Rechnern ein Konflikt auftritt, erfordern die Synchronisierung zwischen den betroffenen Rechnern. Hierzu erfolgt ein Nachrichtenaustausch über das erwähnte CTC-Protokoll sowie die OS/390-Komponente XCF.

Die CF verwaltet in der zentralen Sperrtabelle eine reduzierte Sperrinformation mit einer festen maximalen Anzahl n von Sperreinträgen. Jeder Eintrag der CF-Sperrtabelle bezieht sich nicht notwendigerweise auf ein einzelnes Objekt, sondern auf eine Hashklasse, d.h., jedes zu sperrende Objekt ist über eine Hashfunktion auf einer der n Hashklassen abzubilden. Zum Beispiel haben Sperren in IMS-Datenbanken bis zu 19 Bytes (152 Bits) lange Bezeichner. Da eine Sperrtabelle mit 2^{152} Einträgen für den Hauptspeicher viel zu groß (und sehr dünn belegt) wäre, erfolgt eine Abbildung in eine Hashklassen-Bezeichnung von 18 Bit. Der Adressraum für die Lock-

Einträge kann somit über eine CF-Sperrtabelle von 2^{18} = 262.144 Einträge abgedeckt werden. Dabei wird unterstellt, dass jeder Knoten den gleichen Hashing-Algorithmus benutzt. Da die n Einträge Hashklassen repräsentieren, können »unechte Konflikte« auftreten, wenn zwei Sperrbezeichner in dieselbe Hashklasse abgebildet werden. Die Häufigkeit solcher Kollisionen ist eine Funktion der Sperrtabellengröße. Diese kann vom Systemprogrammierer dynamisch verändert werden, um die Menge der falschen Konflikte zu minimieren. Eine Faustregel besagt, dass nicht mehr als 1% der Sperranforderungen zu Konflikten führen sollte.

Abbildung 4.2 verdeutlicht für eine Beispielkonfiguration mit drei Systemen den Aufbau der globalen Sperrtabelle in der CF sowie der lokalen Sperrtabellen der LLMs. Die zentrale Sperrinformation wird nicht auf Ebene einzelner Transaktionen, sondern nur für ganze Systeme geführt. Das Sperrprotokoll unterscheidet dabei wie üblich Lesesperren, welche pro Objekt gleichzeitig an mehrere Systeme gewährt werden können, und exklusive Schreibsperren. Die CF-Sperrtabelle (oberer Teil in Abb. 4.2) verwendet hierzu pro Sperreintag (Hashklasse) zwei Felder: Für den Fall einer exklusiv gesetzten Sperre (EXC) enthält das erste Feld die Knotenadresse des Besitzers. Ein Bit-Vektor mit 32 Bit Länge nimmt das zweite Feld ein. Jedes Bit dieses Vektors zeigt an, welcher der (maximal möglichen) 32 Rechner eine Lesesperre erworben hat (Shared, SHR=1) bzw. ob keine solche Sperre vorliegt (SHR=0). Das Beispiel weist somit aus, dass System 1 exklusiver Besitzer von allen Objekten der Hashklasse i ist und dass Systeme 2 und 3 Leseberechtigungen für alle Objekte der Hashklasse k besitzen.

Wie im unteren Teil von Abbildung 4.2 gezeigt, bestehen die lokalen Sperrtabellen der LLMs aus drei Teilen: lokale Sperrzustände (Local Lock State), objektspezifische Sperranforderungen lokaler Transaktionen (Local Queue) und globale Warteschlangen mit objektspezifischen Sperranforderungen mehrerer Rechner (Global Queue). Im lokalen Sperrstatus führt der LLM seine Kenntnis vom Zustand der CF-Sperreinträge bezüglich der Hashklassen: »0« bedeutet keine Kenntnis (Eintrag wird möglicherweise von keinem anderen Knoten benutzt), »S« (Shared-Kenntnis, LLM hat Leserecht), »E« (exklusive Nutzung durch den eigenen Knoten) oder »Gx« (ein anderer Knoten x besitzt exklusive Rechte und kontrolliert die Sperrvergabe im Sysplex). Im Beispiel weisen die lokalen Einträge entsprechend der CF-Sperreinträge System 1 als exklusiven Besitzer der Hashklasse i und Systeme 2 und 3 als leseberechtigt für Hashklasse k aus. Der zweite Bereich der lokalen Sperrtabellen enthält Informationen zu den spezifischen Objekten und lokal laufenden Transaktionen bzw. Prozessen, welche Sperren besitzen oder darauf warten. So bedeutet der Eintrag »(A, i, P1, E)« in

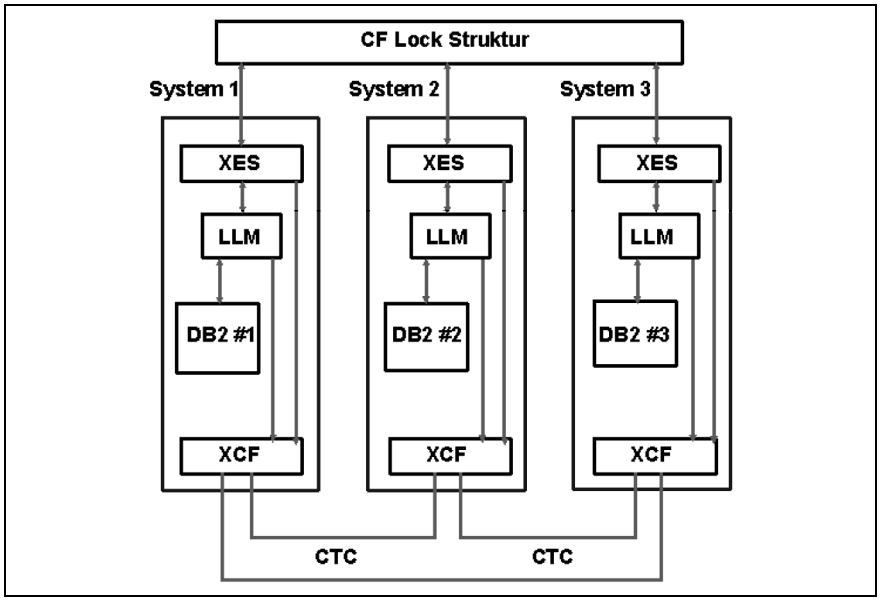

Abb. 4.1: Kommunikationsstruktur zur globalen Sperrbearbeitung im Sysplex

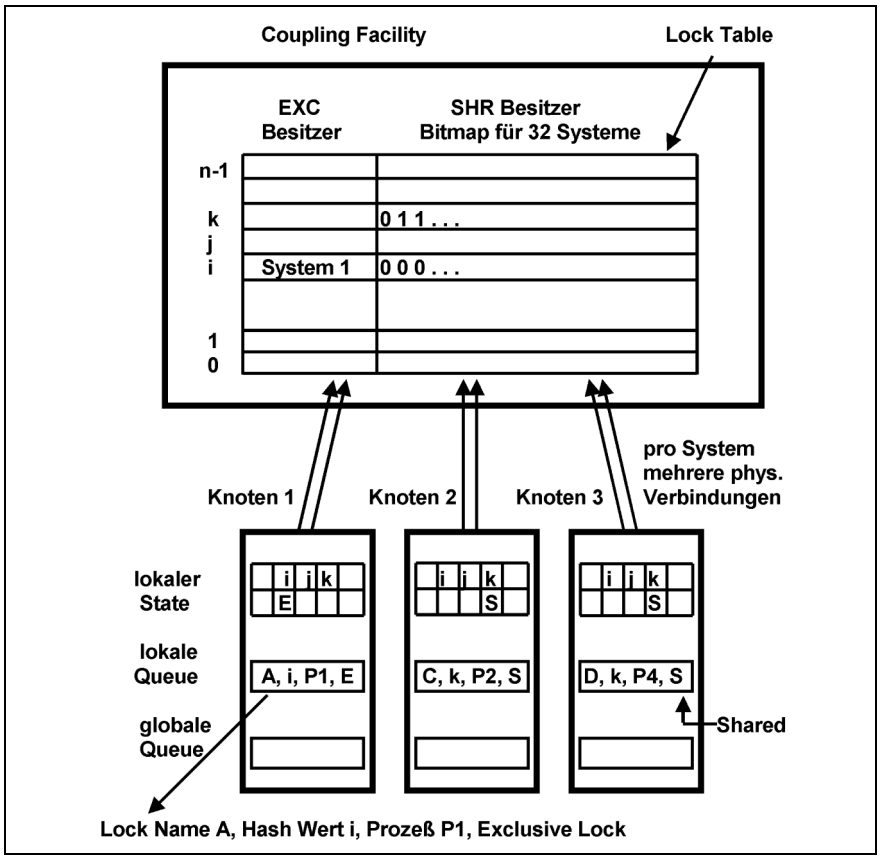

Abb. 4.2: Zentrale CF-Sperrtabelle und lokale Sperrtabellen der LLM

System 1, dass der lokale Prozess P1 die Sperre zu Objekt A, welche zur Hashklasse i gehört, im exklusivem Modus besitzt. Der dritte Bereich (Global Queue) ist nur relevant für Objekte, an denen ein systemübergreifender Sperrkonflikt in der CF erkannt wurde und der LLM zur Konfliktauflösung mit anderen LLMs zusammenarbeiten muss. Hierzu werden in dem Sperrbereich die Sperranforderungen der anderen Rechner abgelegt, um diese global synchronisiert abzuarbeiten.

Mit diesen Datenstrukturen läuft die Sperrbehandlung für Objekt O folgendermaßen ab:

- Hat der LLM für die O zugeordnete Hashklasse j kein Besitzrecht (Eintrag »0« im lokalen Sperrstatus), wird ein Sperraufruf für j mit dem gewünschten Zugriffsmodus (E oder S) an die CF gerichtet. Liegt kein Konflikt vor, was in der Mehrzahl der Fälle zu erwarten ist, wird die Sperre an den LLM gewährt und der globale Sperreintrag für j entsprechend angepasst. Diese globale Sperrgewährung erfolgt aufgrund der nahen Kopplung sehr schnell (Mikrosekunden). Der LLM passt den lokalen Sperrstatus an und gewährt der Transaktion die angeforderte Sperre.
- Sperranforderungen, zu deren Hashklasse der LLM ein exklusives Besitzrecht hat, können lokal – ohne Zugriff auf die zentrale CF-Sperrtabelle – behandelt werden. Dies ermöglicht eine noch schnellere Sperrbearbeitung und reduziert die Auslastung der CF. Im Beispiel von Abbildung 4.2 kann so System 1 alle Lese- und Schreibsperranforderungen zu Objekten der Hashklasse i (also nicht nur für Objekt A) lokal behandeln.
- Analog können lesende Sperranforderungen, zu deren Hashklasse der LLM ein S-Besitzrecht hat, ohne Verzögerung lokal bewilligt werden. Im Beispiel von Abbildung 4.2 können die Systeme 2 und 3 alle Leseanforderungen bezüglich Hashklasse k ohne erneuten CF-Zugriff behandeln.
- Bei globalen Sperrkonflikten stellt die Coupling Facility sicher, dass nur diejenigen Knoten von dem Auflösungsprozess betroffen sind, die vorher ein Interesse angemeldet haben (in der zentralen Sperrtabelle vermerkt). Dabei erhält einer der beteiligten LLMs die Verantwortung zur Auflösung des globalen Konflikts (i.Allg. der Knoten, welcher das exklusive Besitzrecht zur Hashklasse hat bzw. möchte). Über ein nachrichtenbasiertes Protokoll erfolgt ein Austausch der spezifischen Sperranforderungen sowie deren schrittweise Abarbeitung durch die beteiligten Rechner.

Eine detaillierte Beschreibung der Sperrbehandlung ist in [ISJ97] zu finden. Wichtig ist, dass typischerweise mehr als 99% der Sperranforderungen rechnerlokal oder sehr schnell über die CF behandelt werden können und nur in den wenigen Fällen eines gleichzeitigen rechnerübergreifenden Sperrkonflikts ein aufwendigeres nachrichtenbasiertes Protokoll abläuft.

Das skizzierte Basisprotokoll steht allen Subsystemen der Knoten zur Verfügung, lässt sich jedoch auf spezifische Erfordernisse anpassen, z.B. bezüglich der Objektnamen und Sperrgranularitäten. So wird im relationalen Datenbanksystem DB2 ein *hierarchisches Sperrverfahren* (Explicit Hierarchical Locking, EHL) eingesetzt, bei dem einerseits feingranulare Sperren auf Satz- und Seitenebene zur Reduzierung von Sperrkonflikten unterstützt werden, andererseits grobe Granulate wie Tabellen oder Segmente, um wenig Sperranforderungen stellen zu müssen. Dabei erfolgt die Sperrbehandlung mit der Coupling Facility mit möglichst grober Granularität. Der lokale Lock-Manager (LLM) verwaltet alle Sperren auf der feinsten Granularitätsstufe, wobei die gröberen Granulate mit Anwartschaftssperren (intention locks) belegt werden. Damit können einerseits viele Sperren lokal behandelt werden, wenn der betreffende LLM das exklusive oder Leserecht für ein großes Granulat besitzt. Kommt es zu einem Konflikt, kann durch dynamische Verfeinerung der Sperrgranularität dennoch erreicht werden, dass nur im Falle tatsächlicher Konflikte auf der feinsten Ebene Sperranforderungen längere Zeit blockiert werden.

Im Beispiel von Abbildung 4.3 hat Transaktion 1 auf Knoten 1 auf einem ganzen Segment eine explizite Sperre, die in der CF-Sperrtabelle vermerkt ist (nicht gezeigt). Damit sind in Knoten 1 alle Seiten und Sätze dieses Segmentes implizit gesperrt und Zugriffe darauf innerhalb von Knoten 1 können ohne Kommunikation mit der CF (also lokal) behandelt werden. Die entsprechenden Zugriffe werden dennoch vom LLM vermerkt, z.B. dass Transaktion 1 die Sätze 3 und 4 in den Seiten 1 bzw. 2 referenzierte. Im unteren Teil von Abbildung 4.3 ist die Situation gezeigt, nachdem eine neue Transaktion 2 auf Knoten 2 eine Sperre für Datensatz 5 angefordert hat. Die Sperranforderung geht an die CF, welche Knoten 2 daraufhin vom Konflikt mit Knoten 1 informiert. Da kein tatsächlicher Konflikt zwischen den beiden Transaktionen besteht, kann der Konflikt durch eine Herabstufung und Verfeinerung der Granularität behoben werden. Dies handeln die beiden Knoten über einen Nachrichtenaustausch (CTC-Kommunikation) untereinander aus. Transaktion 1 erhält eine explizite Sperre mittlerer Granularität für Seite 1 und eine explizite Sperre feiner Granularität für Datensatz 4. Transaktion 2 erhält eine explizite Sperre für Datensatz 5. Das Segment und die Seite 2 werden mit einer Anwartschaftssperre belegt, welche anzeigt, dass mehrere unterschiedliche Knoten explizite Sperren des Segments bzw. der Seite besitzen.

4.2 Kohärenzkontrolle

Die Cache-Funktion der Coupling Facility wird zur Kohärenz-Steuerung der Datenbankpuffer in den Hauptspeichern der Sysplex-Knoten verwendet. Wir erläutern die Vorgehensweise am Beispiel von DB2; die CF-Funktionalität zur Kohärenzkontrolle wird jedoch auch von anderen Subsystemen genutzt.

Eine allgemeine, wenngleich ineffiziente Lösungsmöglichkeit zur Kohärenzkontrolle bei loser Kopplung (also ohne CF) ist ein so genannter Broadcast-Invalidierungsansatz, der in Abbildung 4.4 illustriert ist. Dabei wird die Änderung einer Seite in einem Knoten (im Beispiel A) am Ende der ändernden Transaktion über eine Broadcast-Nachricht allen Knoten mitgeteilt. Diese können daraufhin ältere Versionen der betreffenden Seite in ihrem Puffer invalidieren und somit den Zugriff darauf umgehen. Der ändernde Rechner A schreibt vor Freigabe der exklusiven Sperre die geänderte Seite auf Extern-

speicher (»cast-out«), so dass danach alle Knoten auf die aktuelle Version der Seite zugreifen können. Nachteile der Vorgehensweise sind hoher Nachrichtenaufwand durch die Broadcast-Invalidierung, der mit der Knotenanzahl zunimmt (geringe Skalierbarkeit), sowie hoher I/O-Aufwand für das Cast-out auf die langsamen Plattenspeicher. Die Nachteile werden abgemildert in einer Variante des Ansatzes, bei der die Sperre für eine geänderte Seite über Transaktionsgrenzen hinweg gehalten wird und die Freigabe der Sperre und das Cast-out erst erfolgen, wenn ein Konflikt durch einen anderen Knoten vorliegt.

Eine effiziente Lösung zur Kohärenzkontrolle wird jedoch erst durch Einsatz der CF erreicht. Hierbei werden die CF-Cache-Strukturen verwendet, die aus zwei Teilen bestehen: Directory-Einträge und optional globale Pufferbereiche zur Aufnahme von Datenseiten bzw. -blöcken. Ein Directory-Eintrag existiert für jeden Datenblock, der sich entweder in dem lokalen Puffer irgendeines Knotens oder in einem globalen Puffer der CF befindet. Jeder Directory-Eintrag enthält einen Hinweis darauf, welche Knoten momentan Kopien des entsprechenden Datenblocks besitzen. In dem geschützten Bereich des Hauptspeichers eines jeden Knotens ist daneben ein Bit-Vektor definiert, wobei ein Bit zu jedem lokalen Pufferrahmen existiert.

Beim Lesen einer noch nicht lokal gepufferten Datenseite durch eine DB2-Instanz an Knoten B wird der Aufruf zunächst an die CF gerichtet, um den Block ggf. sehr schnell vom globalen Puffer einzulesen. Falls die Seite nicht im globalen Puffer vorliegt, wird sie vom Plattenspeicher gelesen. In beiden Fällen wird im CF-Directory die Datenseite registriert und die Position vermerkt, an der sie in dem lokalen Puffer in B abgelegt wird. Außerdem wird im Bit-Vektor von B das entsprechende Bit gesetzt, um eine gültige Datenseite anzuzeigen. Am Ende einer Update-Transaktion werden geänderte Seiten in den globalen Puffer in der CF durchgeschrieben, was wesentlich schneller als das Schreiben auf Platte erfolgt. Die CF übernimmt die Seiten und ermittelt über die Directory-Einträge, welche Knoten Kopien von den betroffenen Seiten haben, und schickt nur an diese Invalidierungssignale. Das Invalidierungssignal bewirkt ein Umschalten das

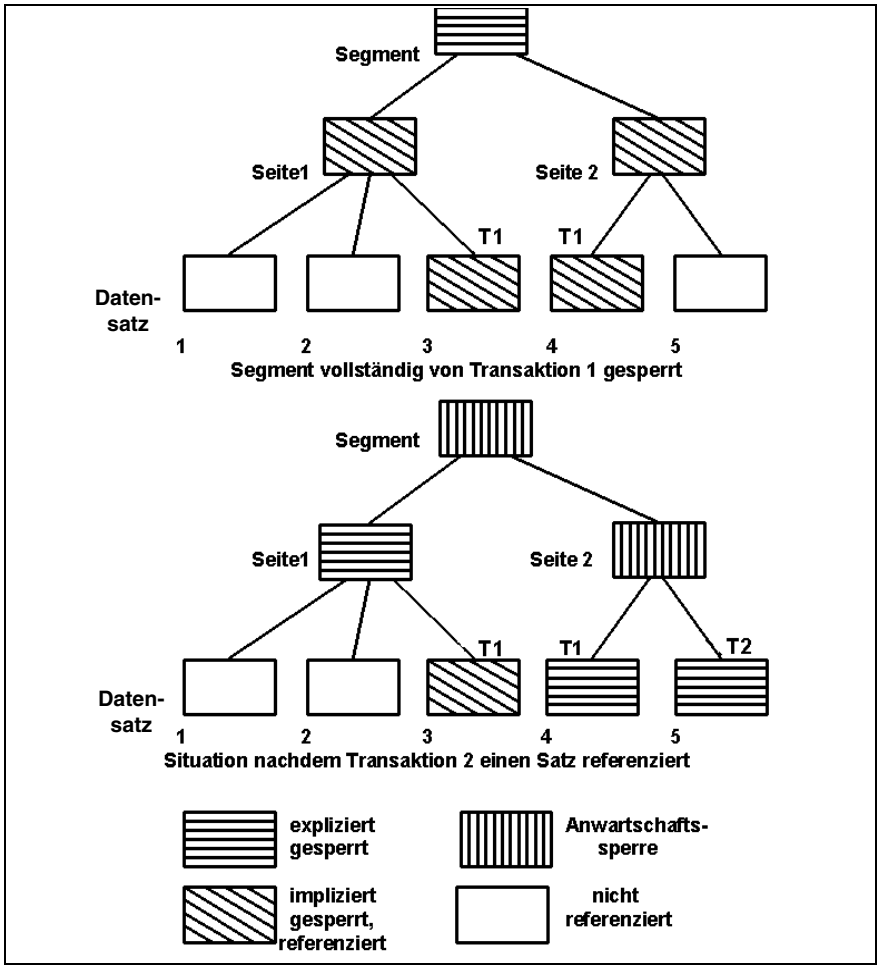

Abb. 4.3: Explizites hierarchisches Locking

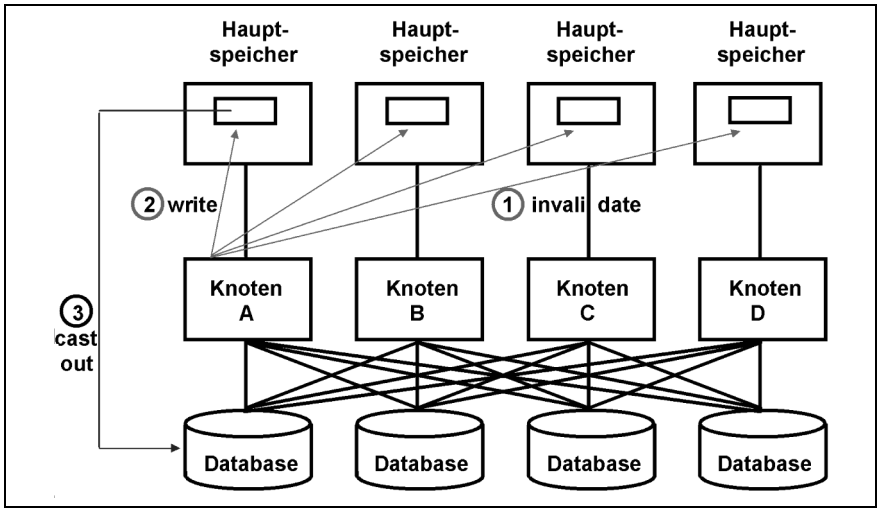

Abb. 4.4: Broadcast-Invalidierung zur Kohärenzkontrolle

der Datenseite entsprechende lokale Bit des Bit-Vektors zur Anzeige eines ungültigen Zustands. Dies geschieht ohne Unterbrechung der in dem Knoten laufenden Prozesse. Immer, wenn das Datenbanksystem eine lokal gepufferte Datenseite benutzen möchte, zeigt ein einfacher Bit-Test die Gültigkeit an. In dem Fall, dass dieser Test einen ungültigen Zustand feststellt, erfolgt ein Refresh der lokalen Kopie dieser Datenseite. Dies kann meist sehr schnell aus dem globalen Puffer der CF erfolgen, also wesentlich schneller als das Einlesen der Seite vom Plattenspei-

cher. Die CF ermöglicht somit eine sehr effiziente Invalidierung von Seiten als auch einen schnellen Austausch geänderter Seiten zwischen Knoten.

Der Aufwand zur Kohärenzkontrolle kann weiter begrenzt werden, indem diese nur auf solchen Datenbankpartitionen (Segmenten) erfolgt, bei denen Invalidierungen nicht schon von vorneherein ausgeschlossen sind. Dies ermöglicht Performance-Vorteile für Partitionen, die exklusiv von einem Knoten genutzt bzw. die Sysplex-weit nur gelesen werden. DB2 ermöglicht ferner, auch ungeänderte Seiten, die aus einem lokalen Puffer verdrängt werden sollen, in den globalen CF-Puffer zu schreiben. Dies kann eine bessere E/A-Leistung ermöglichen, da jeder Treffer auf eine Seite des globalen Puffers einen langsamen Plattenzugriff einspart. Da die CF nicht direkt mit den Plattenspeichern verbunden ist (Abb. 3.1), muss das Ausschreiben (Cast-out) geänderter Seiten vom globalen Puffer auf Plattenspeicher über die Hauptspeicher der Knoten erfolgen. Diese Aufgabe wird durch Hintergrundprozesse auf den Knoten erledigt, die aktiviert werden, sobald die Anzahl geänderter Seiten im globalen Puffer einen bestimmten Schwellwert übersteigt. Die Seitenersetzung für ungeänderte Seiten im globalen CF-Puffer erfolgt gemäß LRU (Least Recently Used).

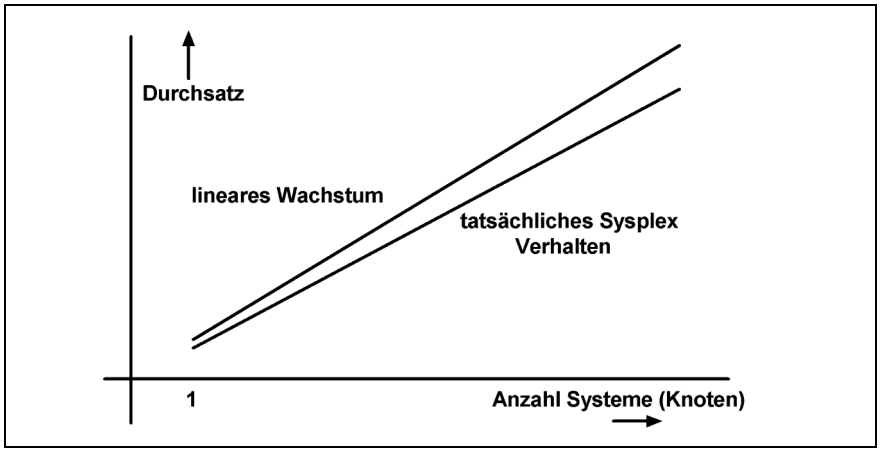

Abb. 5.1: Sysplex-Overhead: typischer Verlauf

Installation	Anzahl Systeme	% Sysplex-Overhead
A	4	11
B	3	10
C	8	9
D	2	7
E	11	10
Relational Warehouse Workload	2	13,30

Abb. 5.2: Sysplex-Overhead: Beispielwerte

5 Leistungsverhalten

Zur Leistungsbewertung von Sysplex-Konfigurationen wurden seitens IBM und Anwendern umfangreiche Analysen und Messungen vorgenommen, welche eine sehr gute Skalierbarkeit zeigten. Hier sollen nur einige wenige Ergebnisse angeführt werden; nähere Angaben zu den Experimenten und weitere Ergebnisse finden sich in [ISJ97, RED98].

Beim Vergleich von Konfigurationen ohne und mit Sysplex-Unterstützung zeigt sich, dass die Softwareunterstützung für den Sysplex – selbst bei nur einem Knoten – einen gewissen Overhead verursacht. Abbildung 5.1 stellt den typischen Verlauf für das Ausmaß dieses Overheads dar, wobei die obere Kurve dem linearen Wachstum und die untere dem realen Sysplex-Verhalten entspricht.

Eine entsprechende Aufstellung des prozentualen Sysplex-Overheads als Funktion der Systemanzahl in verschiedenen Installationen zeigt Abbildung 5.2. Dargestellt sind Messungen aus 5 konkreten Transaktions-Produktionsumgebungen A, B, C, D, E sowie das Ergebnis einer Simulation einer relationalen Anwendung. Die Anzahl der Knoten (Systeme) lag bei 2 bis 11. Der zusätzliche Overhead aufgrund des Einsatzes der Sysplex-Software lag zwischen 7 und 13,3 %. Um diesen Betrag liegt der Durchsatz beim Einsatz der Sysplex-Software unter dem theoretischen Maximum eines linearen Durchsatzwachstums ohne Kontroll-Overhead.

Abbildung 5.3 zeigt die Ergebnisse einer Simulation mit dem CICS-Transaktionsmanager, CICSplex System-Manager und IMS-Datenbank für eine Mischung aus OLTP- und Analyseanwendungen. Beim Scaleup mit 8 Systemen stieg die Leistung auf das 3,89fache, bei 16 Systemen auf das 7,40fache der Leistung einer Installation mit 2 Systemen. Dies ist ein fast linearer Scaleup. Ähnliche Ergebnisse wurden aus der Praxis berichtet, darunter auch mit SAP R/3-Anwendungen.

Von Bedeutung ist, dass die Anwendungen selbst nicht angepasst wurden. In Anbetracht der Skalierungsprobleme, die normalerweise bei der Portierung von Einzelprozessoranwendungen auf einen Cluster auftreten, ist dies ein sehr gutes Ergebnis.

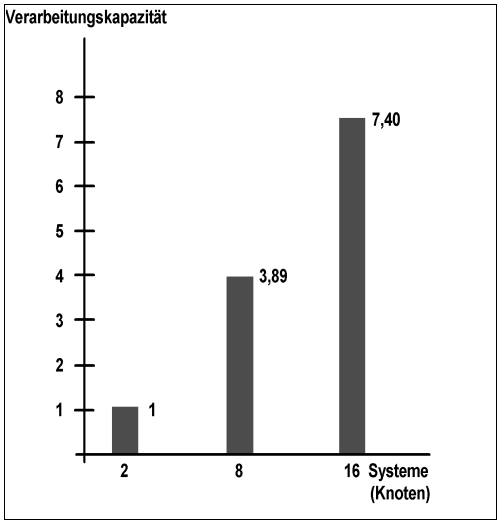

Abb. 5.3: Sysplex-Durchsatzverhalten (Beispiel)

6 Zusammenfassung

Der vorgestellte Sysplex-Ansatz realisiert eine innovative Großrechner-Cluster-Architektur, die in zahlreichen Datenbank- und Transaktionsanwendungen eine sehr hohe Leistungsfähigkeit und Skalierbarkeit unter Beweis gestellt hat. Die wesentlichen Cluster-Komponenten sind der FICON-Hochleistungs-Switch zur Kopplung der Rechner und Plattenspeicher (Shared Disk), der Sysplex Timer sowie die Coupling Facility (CF). Die CF unterstützt eine nahe Kopplung (close coupling), bei der die Rechner synchron und sehr schnell auf globale CF-Datenstrukturen und Datenseiten in globalen Pufferbereichen zugreifen. Damit werden leistungskritische Cluster-Aufgaben zur Synchronisation, Kohärenzkontrolle und Lastbalancierung wesentlich effizienter lösbar als bei loser Kopplung über allgemeine nachrichtenbasierte Kommunikationsprotokolle. Die Kontrollaufgaben werden durch eigene Maschinenbefehle und Betriebssystemfunktionen unterstützt, welche von Subsystemen wie Datenbanksystemen (DB2, IMS), Transaktions- und Applikationsservern (CICS, WebSphere) genutzt werden.

Literatur

[HKS02] *Herrmann, P.; Kebschull, U.; Spruth, W. G.:* Einführung in z/OS und OS/390. Oldenbourg-Verlag, erscheint 2002.

[HORS00] *Horswill, J.:* Designing & Programming CICS Applications. O´Reilly, 2000. ISBN 1-56592-676-5.

[ISJ97] Sonderheft des IBM Systems Journal zum Thema Sysplex, Vol. 36, No.2, 1997. http://www.research.ibm.com/journal/sj36-2.html

[JEDI02] http://jedi.informatik.uni-leipzig.de, 2002.

[JRD92] Sonderheft des IBM Journal of Research and Development zum Thema Sysplex, Vol. 36, No.4, 1992.

[RAHM94] *Rahm, E.:* Mehrrechner Datenbanksysteme. Addison-Wesley (Oldenbourg), 1994. ISBN 3-89319-702-8.

[RED98] System/390 Parallel Sysplex Performance. SG 24-4356.-03, 4th edition, Dec. 1998. IBM Red Book. http://www.redbooks.ibm.com/redbooks/SG244356.html

[RED02] IBM Red Books. http://www.redbooks.ibm.com, 2002.

[SHS01] *Sloan, S.; Hernandez, A. K.; Sloan, S. G.:* DB2 Universal Database for OS/390: An Introduction to DB2 for OS/390 Version 7. Prentice Hall, 2001. ISBN 013019848X.

[ZACK99] *Zack, W. H.:* Windows 2000 and Mainframe Integration. Macmillan Technical Publishing, 1999. ISBN 1-57870-200-3.

Bibliographische Hinweise: Zum Thema Sysplex existiert ein Sonderheft des IBM Journal of Research and Development [JRD92] sowie ein weiteres Sonderheft des IBM System Journal [ISJ 97].

Eine Einführung in z/OS und OS/390 ist in [HKS02] nachzulesen. [ZACK99] enthält einen Vergleich OS/390 mit Windows 2000. Anleitungen für Übungen zum Thema OS/390 sind auf der Leipziger Website [JEDI02] zu finden. [HORS00] enthält einen sehr guten Überblick über die Informationsverarbeitung mit CICS; [SHS01] zu dem Einsatz von DB2 unter OS/390. Umfangreiche technische Informationen zu allen IBM-Produkten sind auf der IBM Redbook-Site verfügbar [RED02]. [RAHM94] behandelt u.a. die wichtigsten Arten von Cluster-Datenbanksystemen (Parallelen DBS), insbesondere Lösungsansätze für Synchronisation und Kohärenzkontrolle in Shared-Disk-Systemen.

Prof. Dr.-Ing. Wilhelm G. Spruth
Fakultät für Informatik, Universität Tübingen
Fakultät für Mathematik und Informatik,
Universität Leipzig
Institut für Informatik
Augustusplatz 10–11
04109 Leipzig
spruth@informatik.uni-leipzig.de

Prof. Dr. Erhard Rahm
Universität Leipzig
Institut für Informatik
Augustusplatz 10-11
04109 Leipzig
rahm@informatik.uni-leipzig.de
http://dbs.uni-leipzig.de

Prof. Dr.-Ing. Wilhelm G. Spruth war langjähriger Mitarbeiter in den IBM Laboratorien in Böblingen und dort u.a. für die Entwicklung des ersten S/370 Mikroprozessors in CMOS Technologie verantwortlich. Er übernahm vertretungsweise von 1993–1998 die beiden Lehrstühle Rechnerarchitektur und Verteilte Systeme an der Universität Leipzig. Derzeitig ist er als Honorarprofessor in Leipzig und ebenso an der Universität Tübingen tätig.

Prof. Dr. Erhard Rahm ist Lehrstuhlinhaber für Datenbanken am Institut für Informatik der Universität Leipzig. Er promovierte und habilitierte an der Univ. Kaiserslautern und verbrachte Forschungsaufenthalte bei IBM und Microsoft Research. Er ist derzeit Sprecher des GI-Arbeitskreises »Web und Datenbanken« und leitet die BTW2003-Tagung in Leipzig.