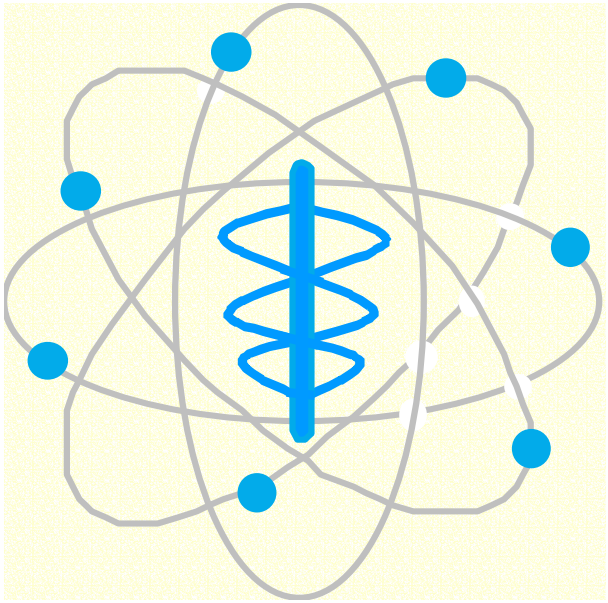
Ein Hochleistungsbetriebssystem mit verteiltem Speicher



- Schlanker Systementwurf
- Zentrale Transaktionsschleife
- Gemeinsamer verteilter Speicher
- Optimistische Synchronisierung
- Rücksetzbare Pufferobjekte
- Heapspeicher-Verwaltung
- Orthogonale Persistenz
- Performance & Resultate
- Weiterführende Themen

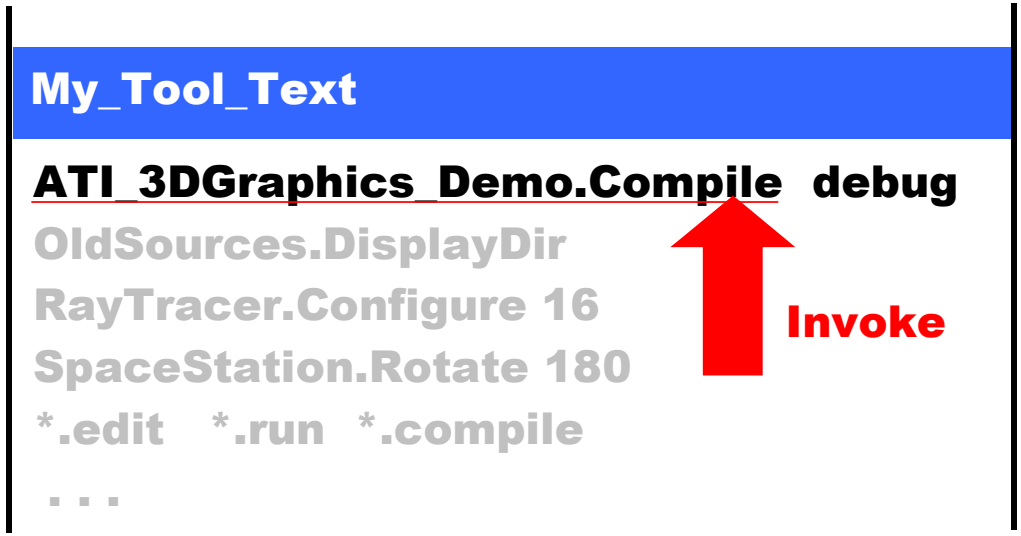
P. Schulthess, M. Schöttner, St. Frenz, R. Göckelmann, M. Fakler
Universität Ulm, Informatik, Verteilte Systeme

Schlanker Systementwurf

- Oberon Tradition:
  - Schlanke Programmierung & Übersetzung,
  - Kooperatives Multitasking,
  - Aktivierbare Tooltexte
  - Event-Schleife,
  - Textelemente,
  - Fenstertracks.

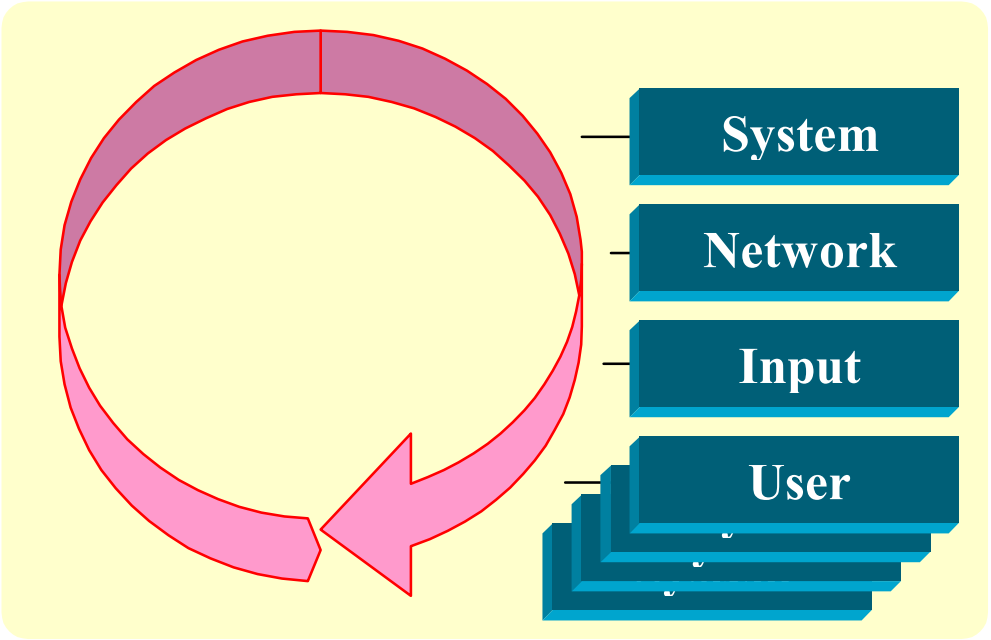
- Zusätzlich:
  - **Aktivierbare Objekte =>**
  - Verteilung,
  - Sharing …

- Mikro-Plurix:
  - 40 Kilobyte,
  - VGA-Text.



Zentrale Transaktionsschleife

- Kooperatives Multitasking (Oberon-style):
  - **Zentrale Event-Schleife =>**
  - System Transaktion (GC ..),
  - Legacy Netzprotokolle,
  - Tastatur & Mauslistener,
  - Installierbare Transaktionen.



- Kernel Funktionen:
  - Initialisierung,
  - Interrupt dispatching,
  - Heapspeicherverwaltung,
  - Stationsobjekt als Kontext,
  - Begin- & End-of-Transaction.

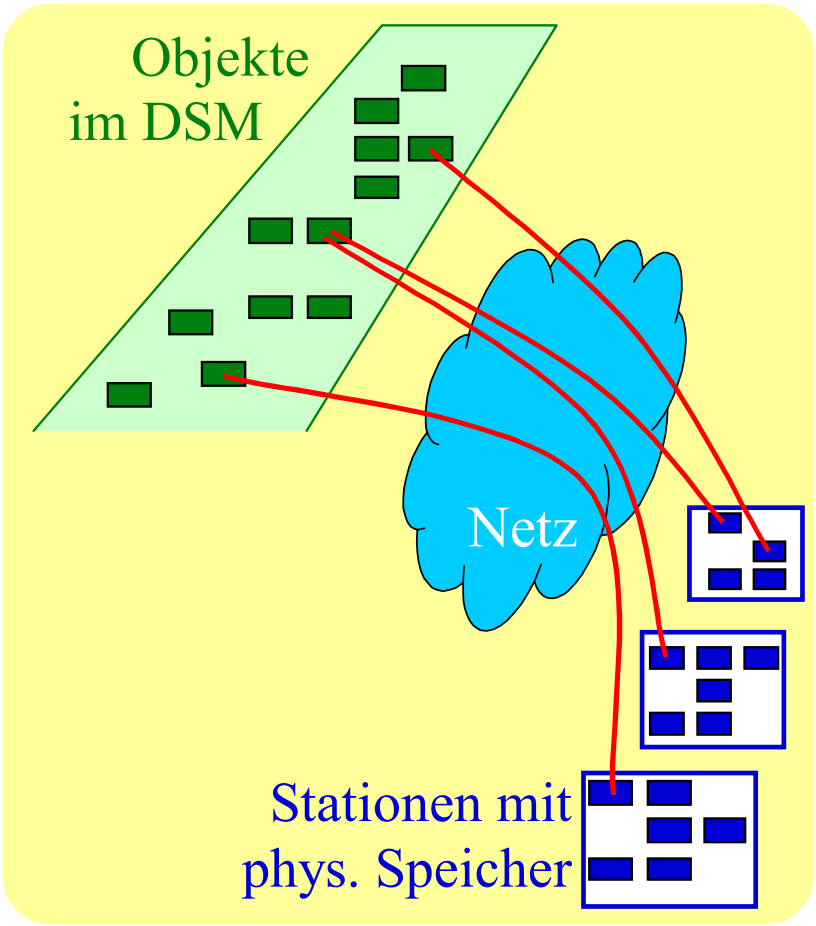
- Besonderheiten:
  - Kurze Transaktionen erwünscht,
  - Implizites Begin- & End-Of-Transaction pro Befehl,
  - Automatischer Wiederanlauf nach Zugriffskonflikten im DSM ...

Gemeinsamer Verteilter Speicher

- Implizite Kommunikation:
  - Über gemeinsamen virtuellen Speicher,
  - Kein RMI/RCP/Corba,
  - TCP/FTP nur extern,
  - Keine Pipes,
  - Kein MPI ...

- Objekte im DSM:
  - Zugriff über Zeiger,
  - Zugriffsschutz über MMU,
  - Namensdienst, aber kein Dateisystem.

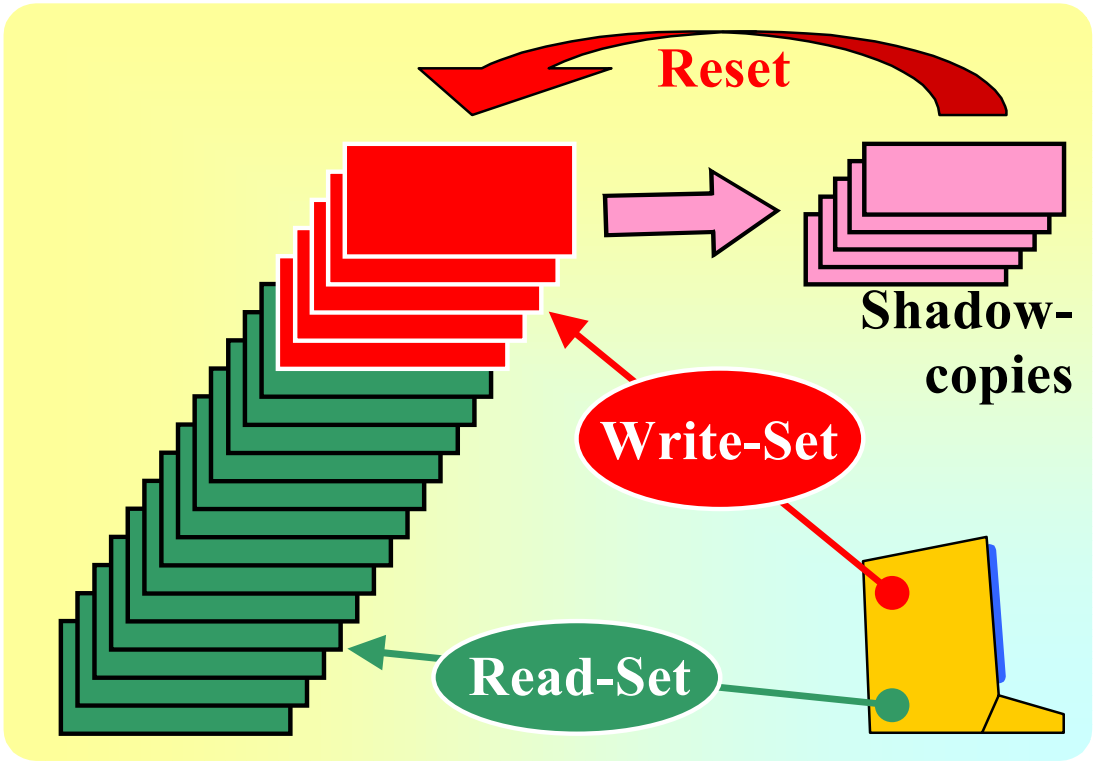
- Totales Konsistenzmodell:
  - ACID Prinzip für Transaktionen,
  - Nicht nur für einzelne Objekte,
  - Sequentielle Konsistenz,



➔ **Optimistische Synchronisierung von Transaktionen.**

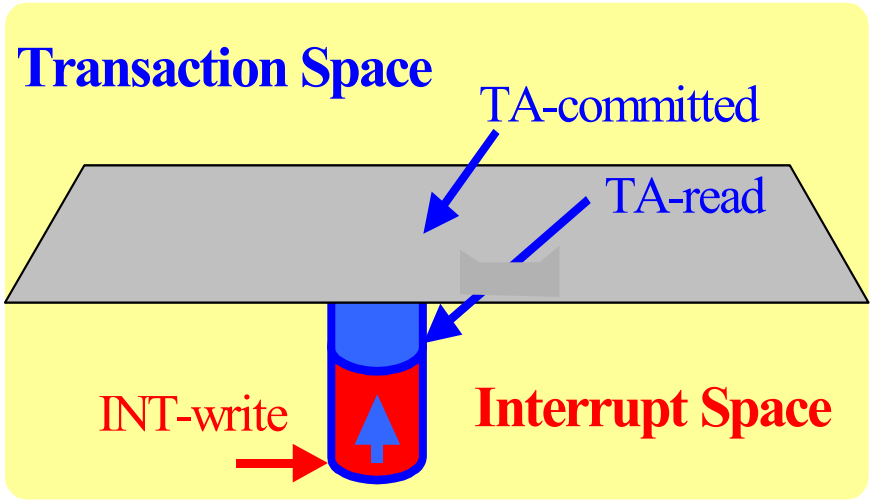
Optimistische Synchronisierung

- Zwischen Transaktionen in verschiedenen Stationen:
  - Im Lauf einer Transaktion wird ein Read- & ein Write Set aufgebaut,
  - Schattenkopie erstellen, bevor eine Seite beschrieben wird,
  - Commit-Request am Ende einer Transaktion,
  - **Im Kollisionsfall Reset ➔**
- Kollisionsauflösung:
  - Serialisierung über Token,
  - Derzeit mit „First wins“,
  - Fairness nicht garantiert.
- Kurze Transaktionen:
  - Minimale Kollisionsrate,
  - Oberon-artige Befehle,
  - Tastatur und Maus,
  - Klasse compilieren,
  - Ein video frame …



Rücksetzbare Pufferobjekte (Smart Buffers)

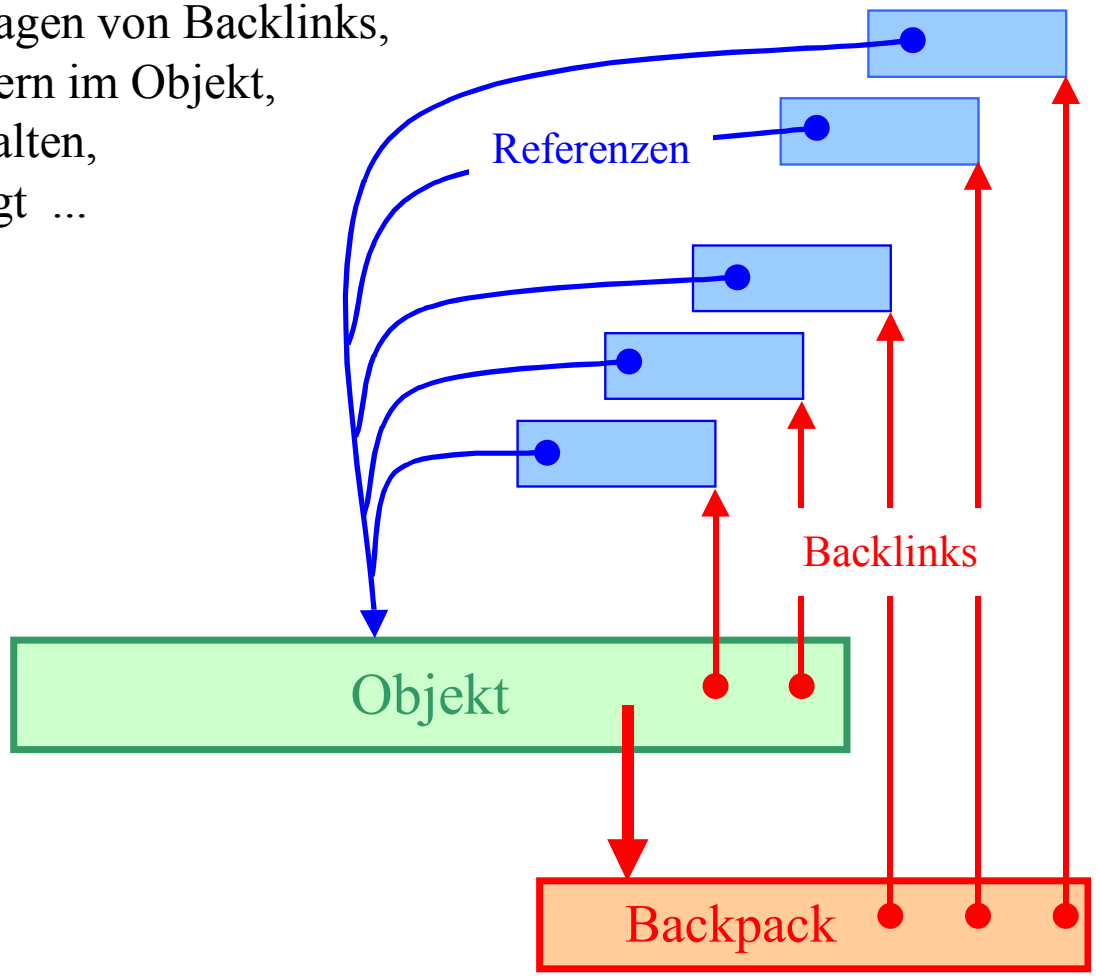
- Für wiederholbare **Eingaben**:
  - Der Schreibzeiger für die Interrupt-Routine ist definitiv.
  - Commitzeiger ist wirksam nach erfolgreichem Commit.
  - Lesezeiger für die Transaktion ist rücksetzbar.



- Smart Buffers liegen zwischen Interrupt- & Transaktionsraum:
  - Die Interrupt-Routine legt Eingabe-Daten in SB
  - Eingaben in eine erneut gestartete Transaktion können erneut gelesen werden
- Die Pufferinhalte nehmen nicht am Rücksetzungsalgorithmus teil (Klassen mit Attribut „NonTransactional“).
- **Ausgabepuffer** können die Ausgaben einer Transaktion verwerfen.

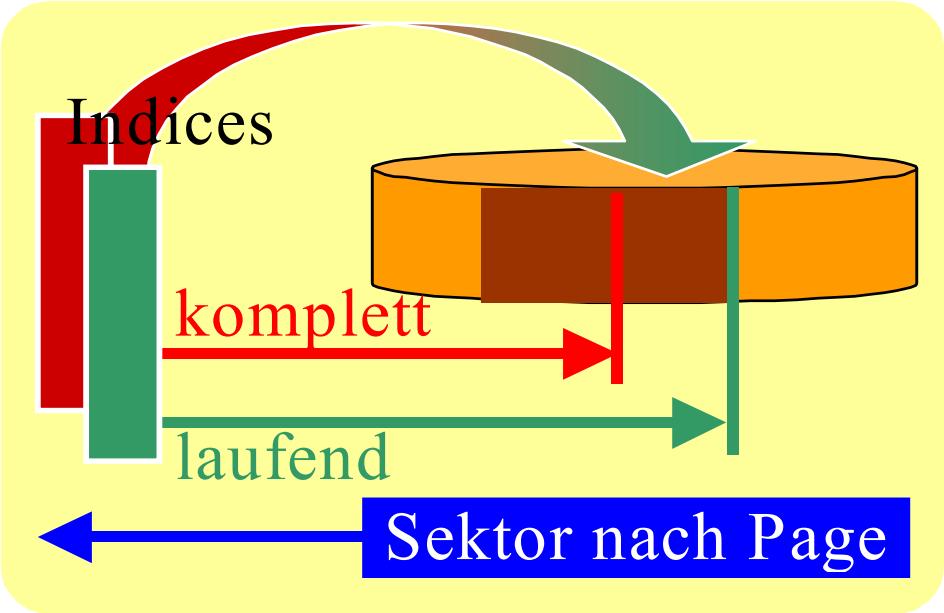
Rückwärtsverkettung mit Backpacks

- Verbesserung gegenüber einer linearen Rückwärtsverkettung:
  - Keine Verdoppelung der Containergrösse für Zeiger,
  - schnelleres Ein- / Austragen von Backlinks,
  - 95% aller Backlinks intern im Objekt,
  - besseres Lokalitätsverhalten,
  - als Hashtabelle ausgelegt ...
- Inkrementelle GC.
- Zyklische GC.
- Relozierung.



Orthogonale Persistenz

- Dauerhaftigkeit (Persistenz):
  - für **alle** noch referenzierbaren Daten (orthogonal),
  - konzeptuell wird der DSM-Cluster nie abgeschaltet,
  - Stationen verlassen den Cluster und treten wieder bei,
  - Import & Export von serialisierten Objekten ist optional,
  - es besteht ein Namensdienst für Objekte, aber kein Dateisystem.
- Page-Server sichert Generationen konsistenter Heap-Images:
  - Minimale Konsistenzverzögerung,
  - Beantworten von Page-requests,
  - Abschalten des DSM-Heaps,
  - Wiederanlauf im Fehlerfall,
  - Mithören am Netz,
  - 45 Mbytes / sec ...
- Periodisches Heap-Image:
  - Mithören am Netz,
  - Seiten anfordern,
  - Komplettieren.



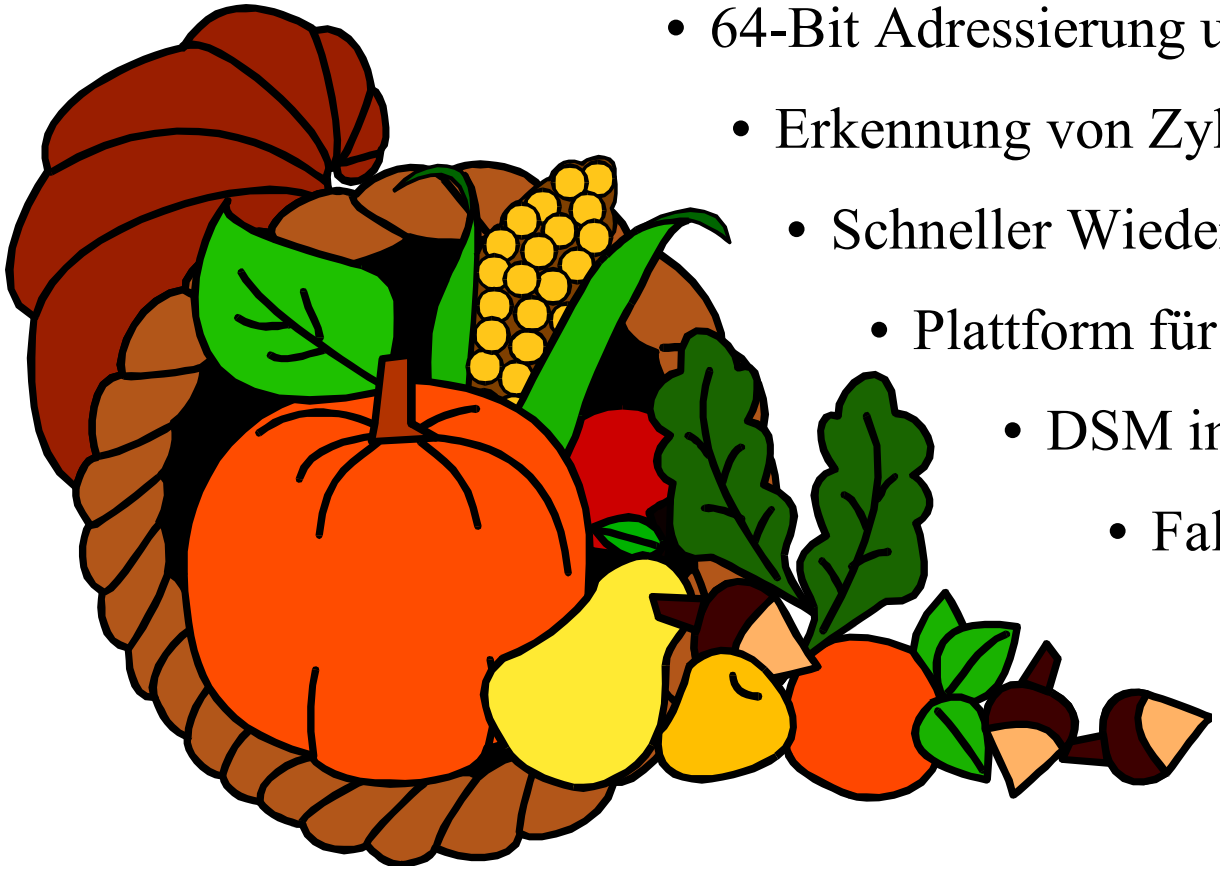
Performance

- Pro:
  - 20 000 leere Transaktionen / sec,
  - 10 000 Zeilen / sec Übersetzung,
  - 45 Mbytes / sec pro Festplatte,
  - 200 000 Schattenkopien / sec,
  - High-Speed 3D-Grafik …

- Contra:
  - Mangelhafte Fairness,
  - Java Allozierungsstrategie,
  - Rücksetzbarkeit von Grafikausgaben,
  - False-Sharing Situationen schwer erkennbar ...

- Optimierungspotentiale:
  - Explizite Variablenallozierung,
  - Optimierender Compiler,
  - Fairnessprotokoll ...

Weiterführender Themen



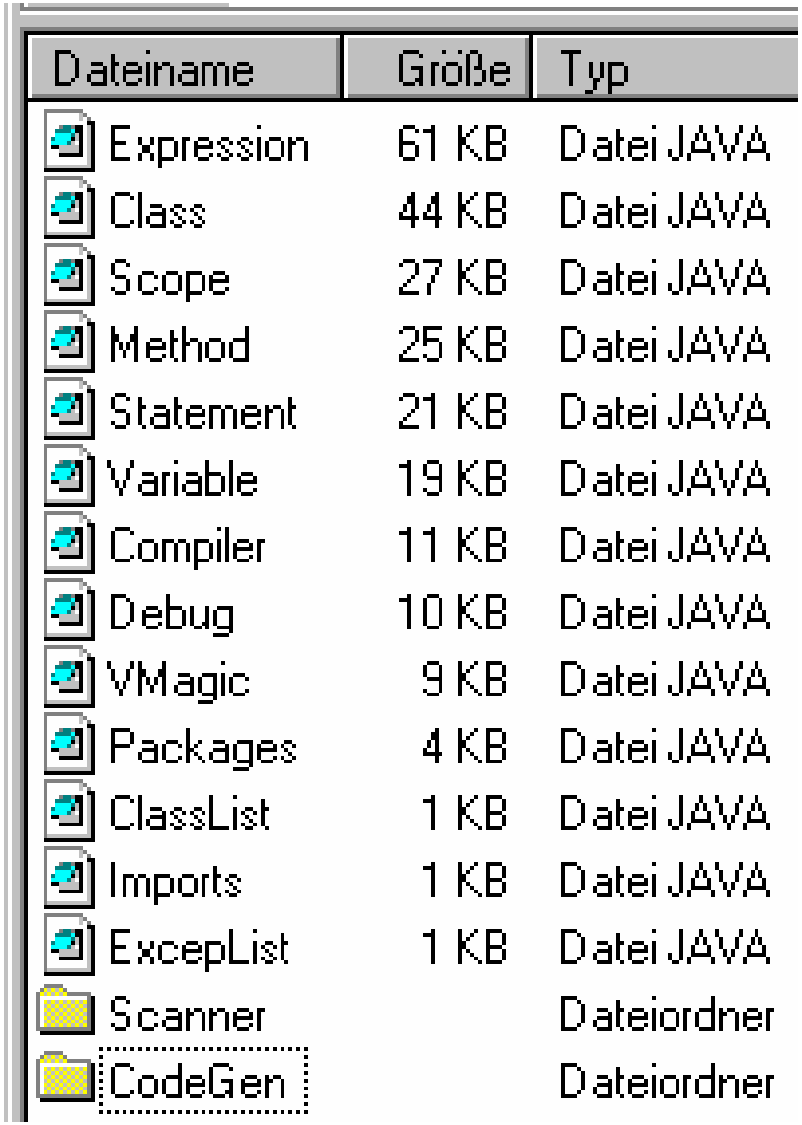
- Alternative Kollisionsauflösungs und Fairness.
- 64-Bit Adressierung und Codeerzeugung.
- Erkennung von Zyklen im DSM Heap.
- Schneller Wiederanlauf vom Server.
- Plattform für 3D-Spiele im Netz.
- DSM im Weitbereichsnetz.
- False-Sharing Control.
- Versionierung.
- Sicherheit !
- **www.plurix.de**


Vielen Dank für Ihre Aufmerksamkeit

Native Java Compiler

- Java Source nach native i486:
  - 150 kByte JVM Bytecode,
  - 10000 Zeilen / sec.
- Direkte Übersetzung in den DSM:
  - Binden & Laden zur Compilationszeit,
  - Frühe Initialisierung von Klassen,
  - Serialisierung entfällt.
- Direkte Programmierung der HW:
  - Compilerklasse “Magic” & “pMapper”,
  - “return” sequenz für Interruptmethoden.
  - inline assembly code …
- Java Übersetzung ist umständlich:
  - Zyklisches Importieren & Initialisieren,
  - Polymorphien und Interfaces,

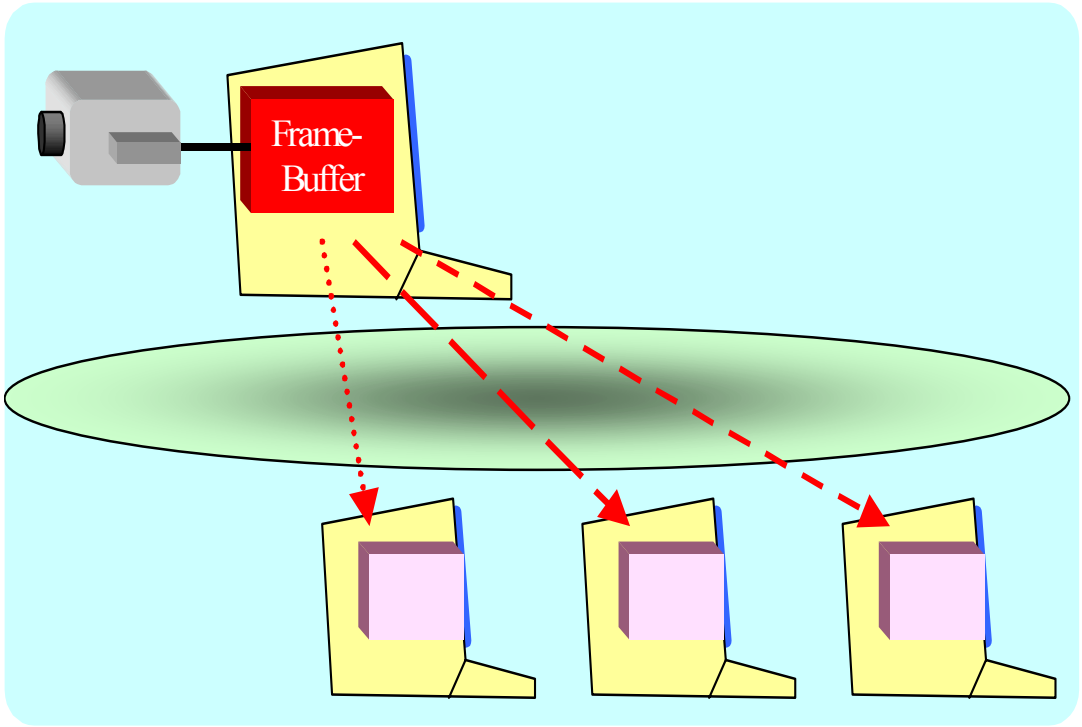
➔ **Multipass compiler**



Dateiname	Größe	Typ
Expression	61 KB	Datei JAVA
Class	44 KB	Datei JAVA
Scope	27 KB	Datei JAVA
Method	25 KB	Datei JAVA
Statement	21 KB	Datei JAVA
Variable	19 KB	Datei JAVA
Compiler	11 KB	Datei JAVA
Debug	10 KB	Datei JAVA
VMagic	9 KB	Datei JAVA
Packages	4 KB	Datei JAVA
ClassList	1 KB	Datei JAVA
Imports	1 KB	Datei JAVA
ExcepList	1 KB	Datei JAVA
Scanner		Dateiordner
CodeGen		Dateiordner

Frame-On-Demand“ Technik

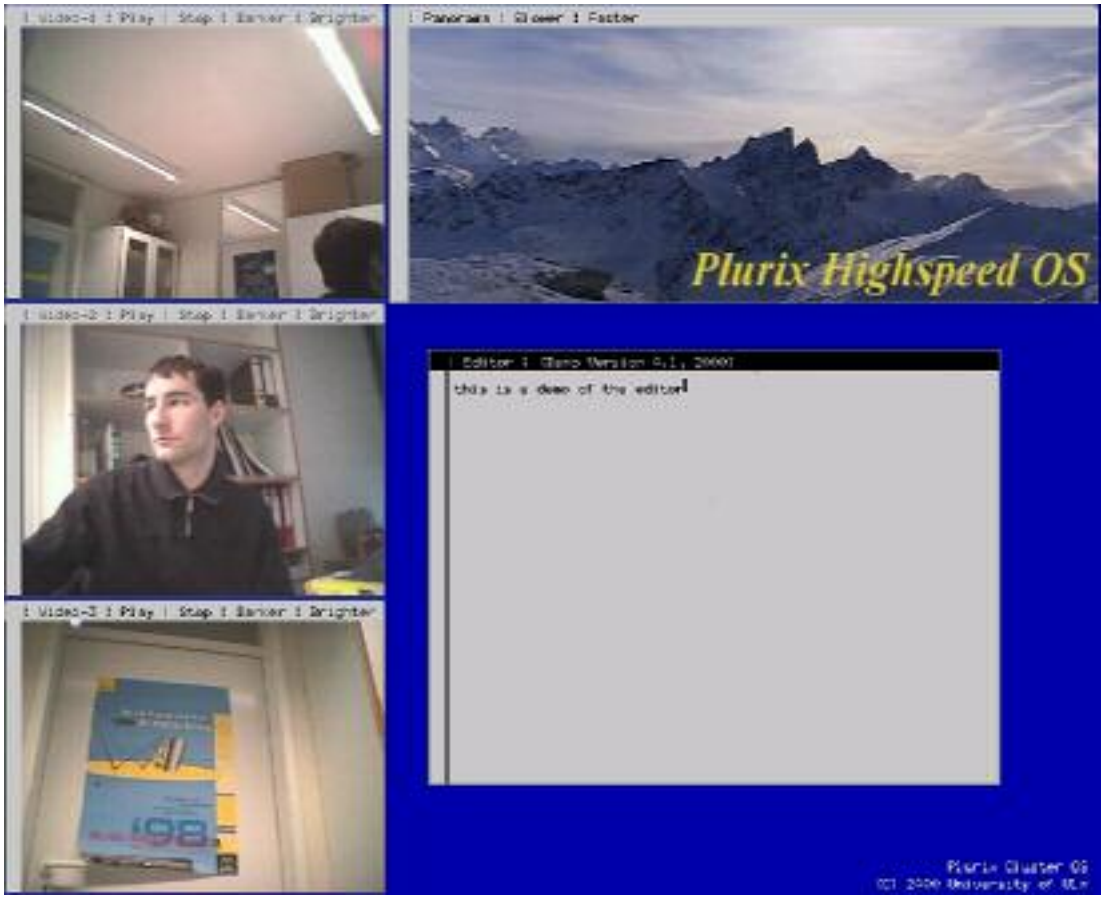
- Je nach Belastung der Stationen und des Netzes werden mehr oder weniger Frames abgeholt.
- Neuen Frame erst holen, nach Invalidierung des alten durch den Sender.
- Verfeinerungen:
  - Komprimierung,
  - „Interest Bit“,
  - Rundspruch.



- Herkömmliche Streaming-Verfahren:
  - Übertragungsratensteuerung,
  - Skalierung beim Sender,
  - Flusskontrolle.

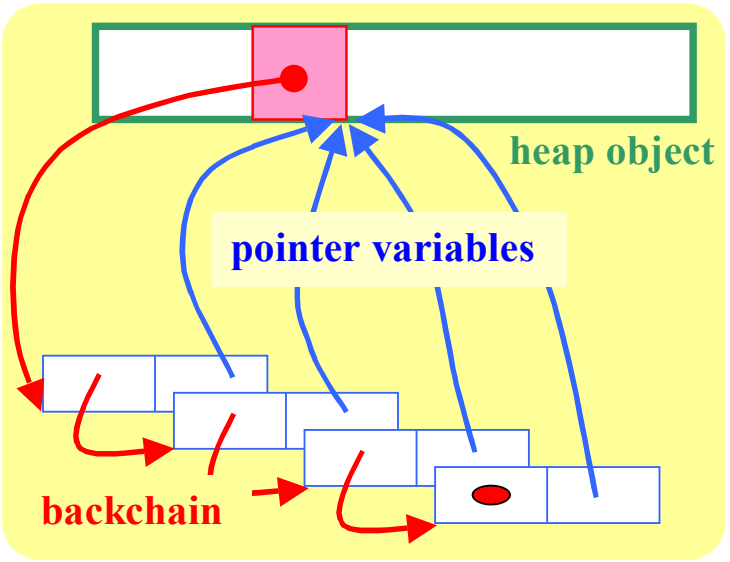
Beispielszenarium CeBit 2000:

- **CeBit2000 Demo ➔**
- Cluster mit 3 PCs:
  - 3 Videokameras
  - (Shared) Texteditor
  - 100 MBit/s Ethernet
  - 360 Grad Panoramabild
  - Program & Daten im DSM
- Progammgrösse:
  - Panorama: 1 KB + 1 MB
  - OS mit Treibern: 40 KB
  - Editor: 10 KB
  - Video 1 KB
- CeBit 2001:
  - QuickBoot mit CD Sound,
  - ATI Radeon Hochleistungsgrafik,
  - Shared TVWindow, DSM-Monitor ...



Heapspeicher-Verwaltung mit Backlink

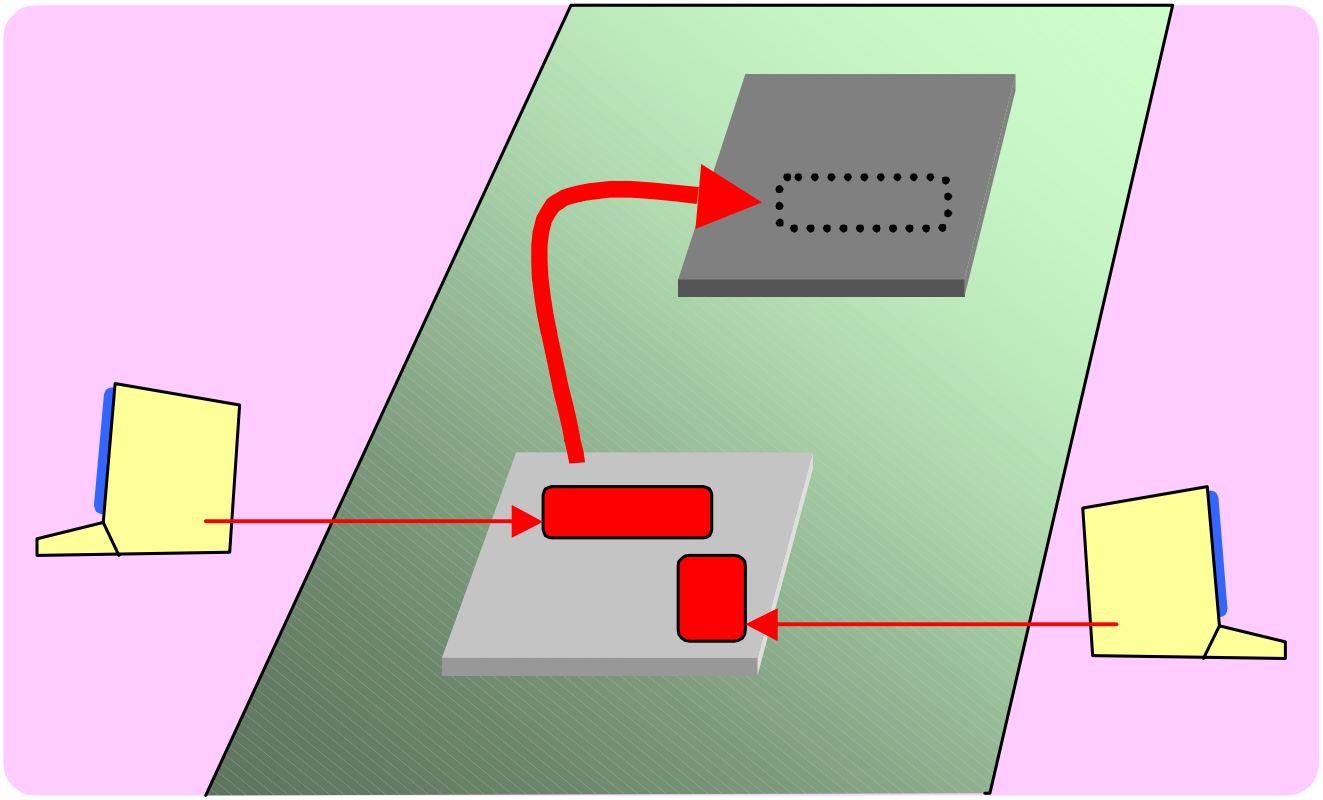
- Virtualisierung durch MMU:
  - Derzeit 4 Gigabytes großer Heap
  - 4 Megabytes für Seitentabellen
  - Verwaltung der Seitenattribute
- Relozierung mit Backchain:
  - Findet alle Referenzen auf ein Objekt,
  - Verschieben eines Objektes im Heap
  - Referenzen mithilfe d. Backchain anpassen
  - Zum Zwecke der Speicherdefragmentierung
  - Zur Speicherentflechtung bei ➔ **False-Sharing**
- Garbage Collection:
  - Objekte mit leerer Backchain sind Garbage
  - Inkrementelle Freispeichersammlung
  - Automatisch und im ganzen Cluster
  - Sammlung nur mit leerem Stack
  - Zyklenerkennung Off-Line ...



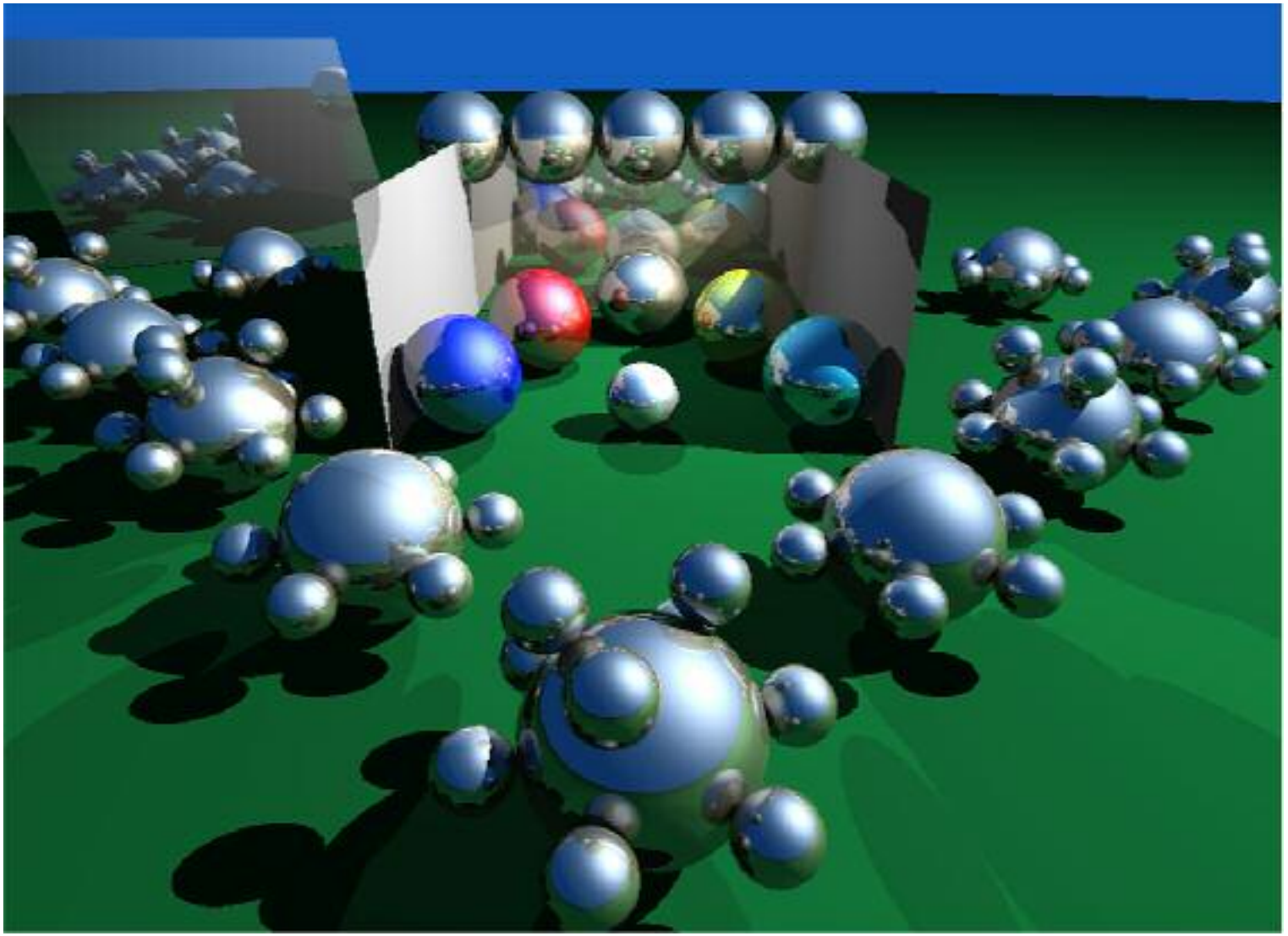
False-Sharing von DSM-Objekten

- Grundphänomen in seitenbasierten DSM-Systemen:
  - Zwei Objekte liegen auf derselben Speicherseite,
  - Zugriffe erfolgen von verschiedenen Knoten,
  - Die Seite wird unnötig oft transportiert,
  - Exakte Definition schwierig ...

Speicher-entflechtung ➔



Raytracing Szene



RayTracing im Cluster

- Skaliert gut für den aktuellen Cluster mit 12-15 Maschinen: