Quality of Service OS Nemesis

Proseminar: Konzepte von Betriebssystem-Komponenten
Thema: Quality of Service OS / Nemesis
Vortragender: Carsten Riedel (sicaried@stud.informatik.uni-erlangen.de)
Datum: 30.01.2003

Übersicht

- Quality of Service (QoS)
- Entstehung von Nemesis
- Aufbau & Struktur von Nemesis

Was ist QoS ?

- Es existiert keine allgemein gültige Definition
- QoS Parameter:
  - Durchsatz
  - Latenzzeit
  - Jitter
  - Zuverlässigkeit

Wo trifft man auf QoS ?

- Voice over IP (VoIP)
- Videokonferenzen
- Web-basierte Datenbanken
- Client Accounting im Rechenzentrum

Entstehung von Nemesis

- Pegasus Projekt (1992-1995)
  - Ziel: Entwicklung eines OS zur Unterstützung von QoS

  - Anforderungen:
    - Normale & Multimedia-Anwendungen
    - Mehrere Multimedia-Anwendungen gleichzeitig
    - Dynamische Ressourcenzuteilung
    - Vermeidung von QoS-Crosstalk

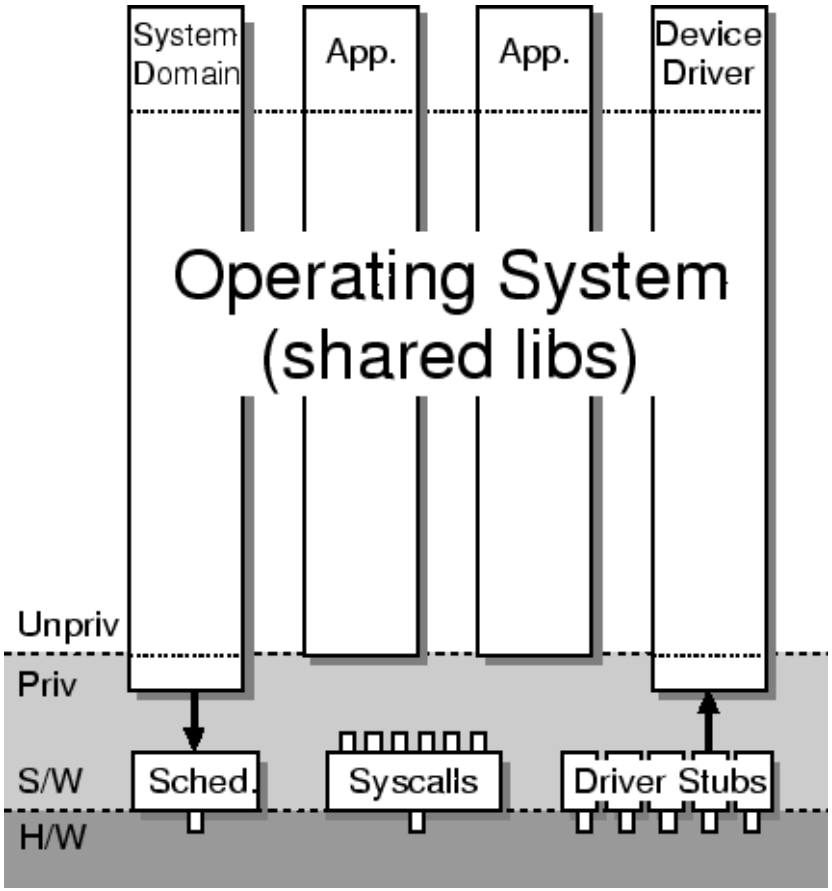
Einige Daten zu Nemesis

- Implementierung in C
- Single Adress Space OS

- Erste Version (1994-1995):
  - Digital 3000/400 AXP
  - 5000/25 (Maxine)

- Neuere Versionen:
  - x86
  - Alpha

Aufbau & Struktur von Nemesis



Nemesis Kern

- verticalized kernel

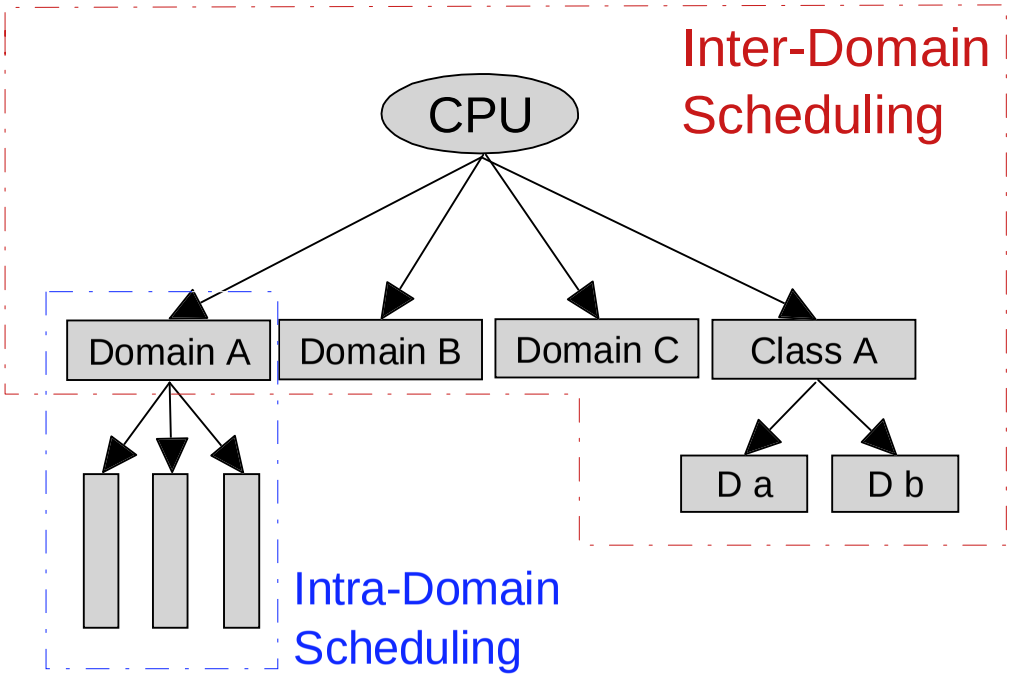
- Komponenten:
  - Scheduler (Atropos)
  - Nemesis Trusted Supervisior Code (NTSC)
  - First-Level Interrupt Stubs

Domains

- Programme in Ausführung
- Komponenten:
  - Threads
  - Interfaces für Systemfunktionen
  - Domain Control Block (DCB):
    - read-only Bereich für Domainverwaltung
    - read-write Bereich

Scheduling

- Zweistufiges Scheduling:
  - Inter-Domain
  - Intra-Domain



Inter-Domain Scheduling

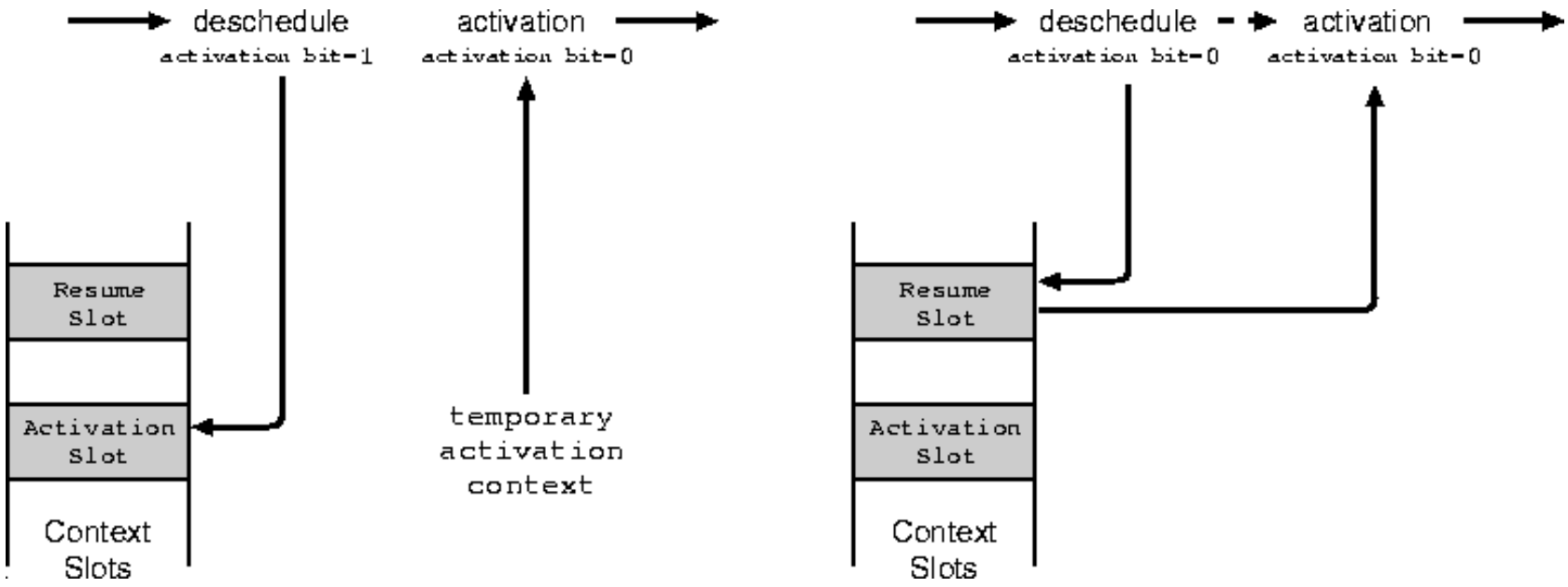
- Atropos:
  - Extern:
    - Tupel {s, p, x, l} von QoS-Parametern:
      - s = benötigte Rechenzeit
      - p = Periode
      - x = Variable vom Typ Boolean für Nutzung überschüssiger Rechenzeit
      - l = latency hint
  - Intern:
    - Earliest Deadline First (EDF) Algorithmus

Virtual Prozessor Interface

- Schnittstelle zwischen Kern & Domains:
  - Versorgung der Domains mit Systeminformationen
  - Events als einfache Inter Domain Kommunikation
- Realisierung über DCB

Prozessorzuteilung

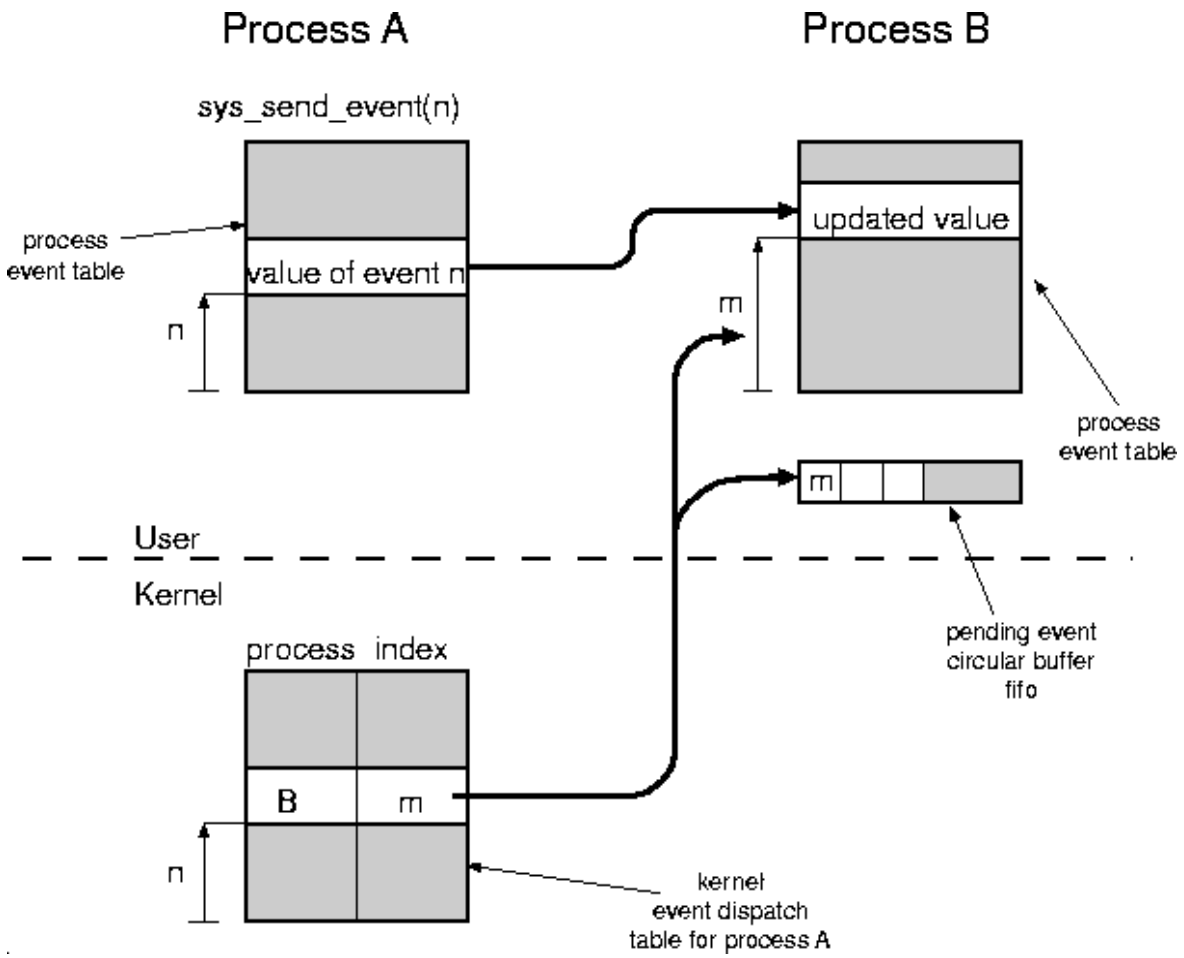
- Kein Resume, sondern Aufruf der Domains
- Activation Handler



Events

- über DCB realisiert
- Repräsentation:
  - Integerwert
  - atomar veränderbar
- Event Channels:
  - Asynchrone simplex Verbindung
  - Verbindungsaufbau über Binder

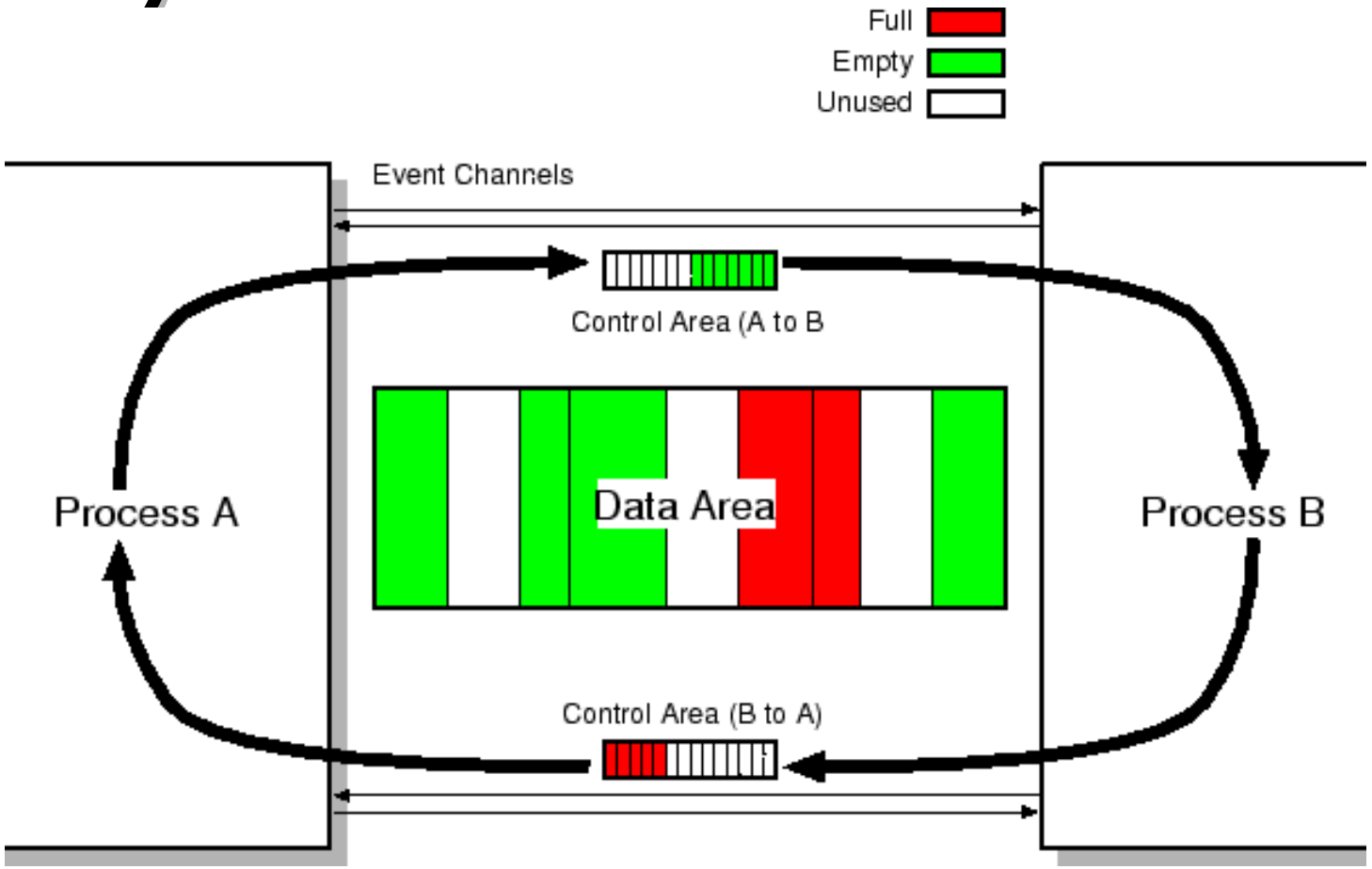
Events


Process A
Process B
sys_send_event(n)
process
event table
value of event n
n
updated value
m
process
event table
m
User
Kernel
pending event
circular buffer
fifo
process index
B
m
n
kernel
event dispatch
table for process A

Inter Domain Communication (IDC)

- I/O Channels (RBuf):
  - Austausch von großen Datenmengen
  - Konzepte:
    - Events
    - Shared Memory
  - Aufbau:
    - Data Area
    - Control Areas
    - Event Channels

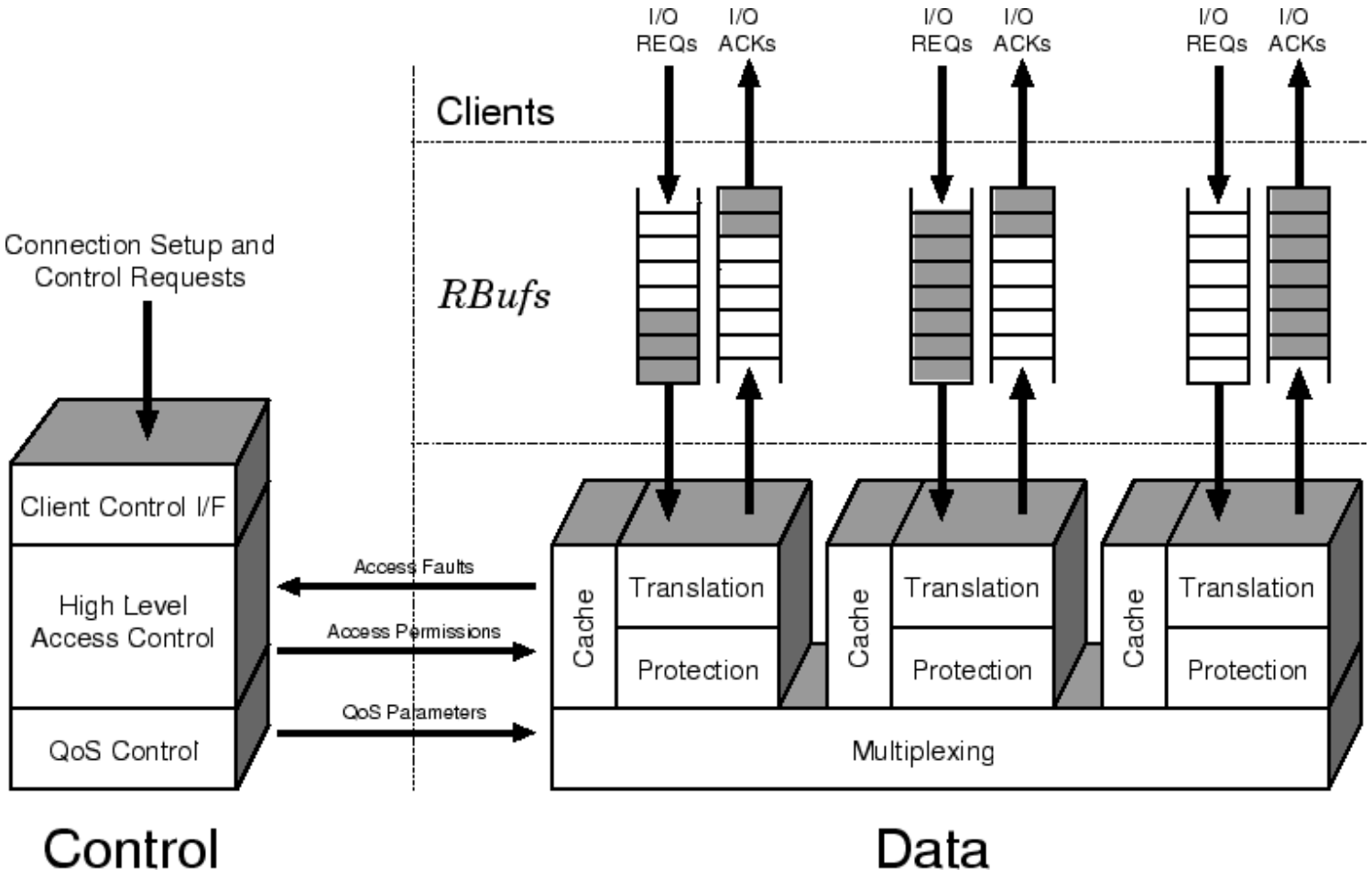
Inter Domain Communication (IDC)



Gerätetreiber

- Device Data-Path Module(DDM)
  - Translation
  - Protection
  - Multiplexing
- Device Control-Path Module (DCM)
  - Einrichtung & Verwaltung

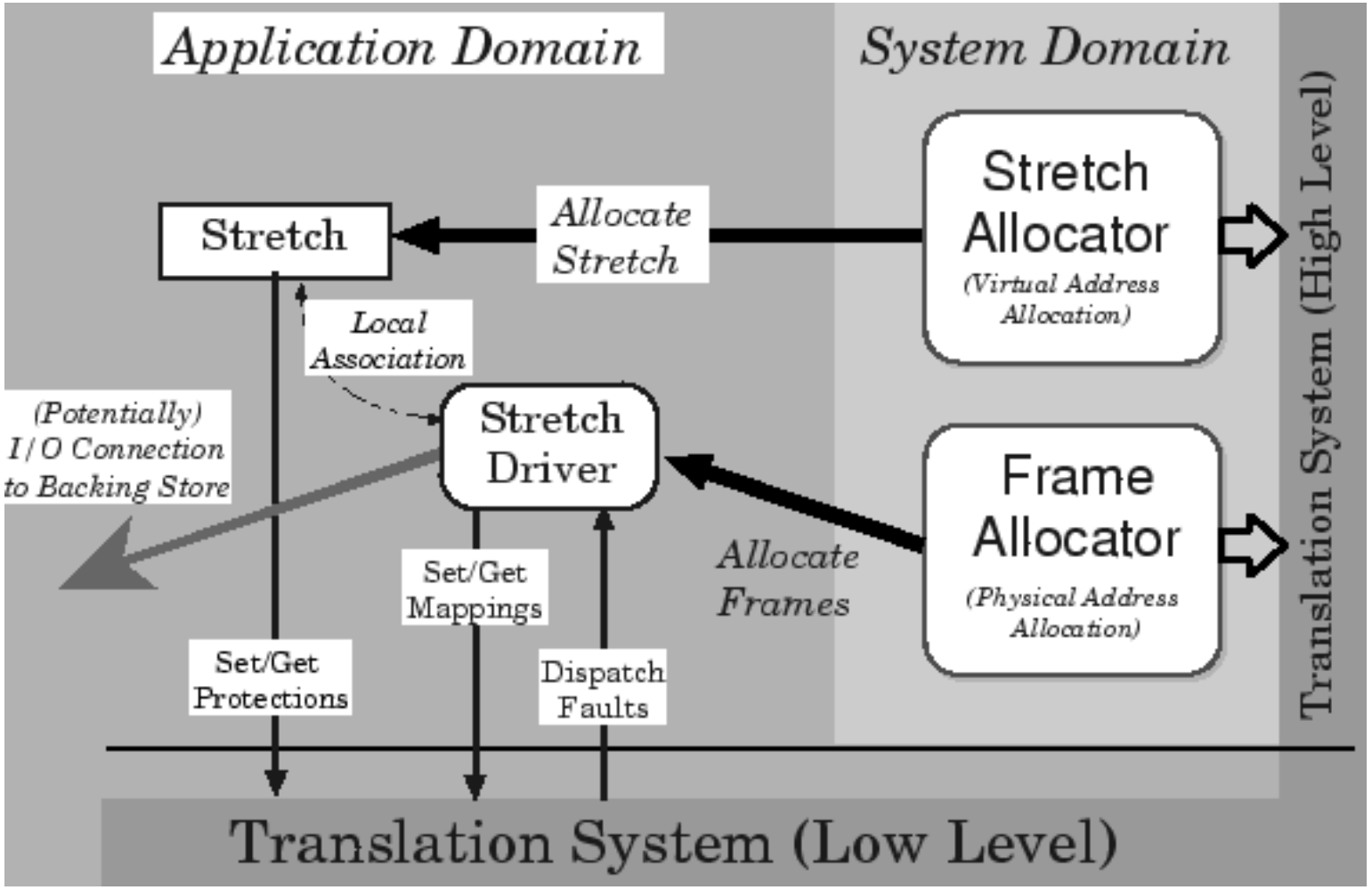
Gerätetreiber



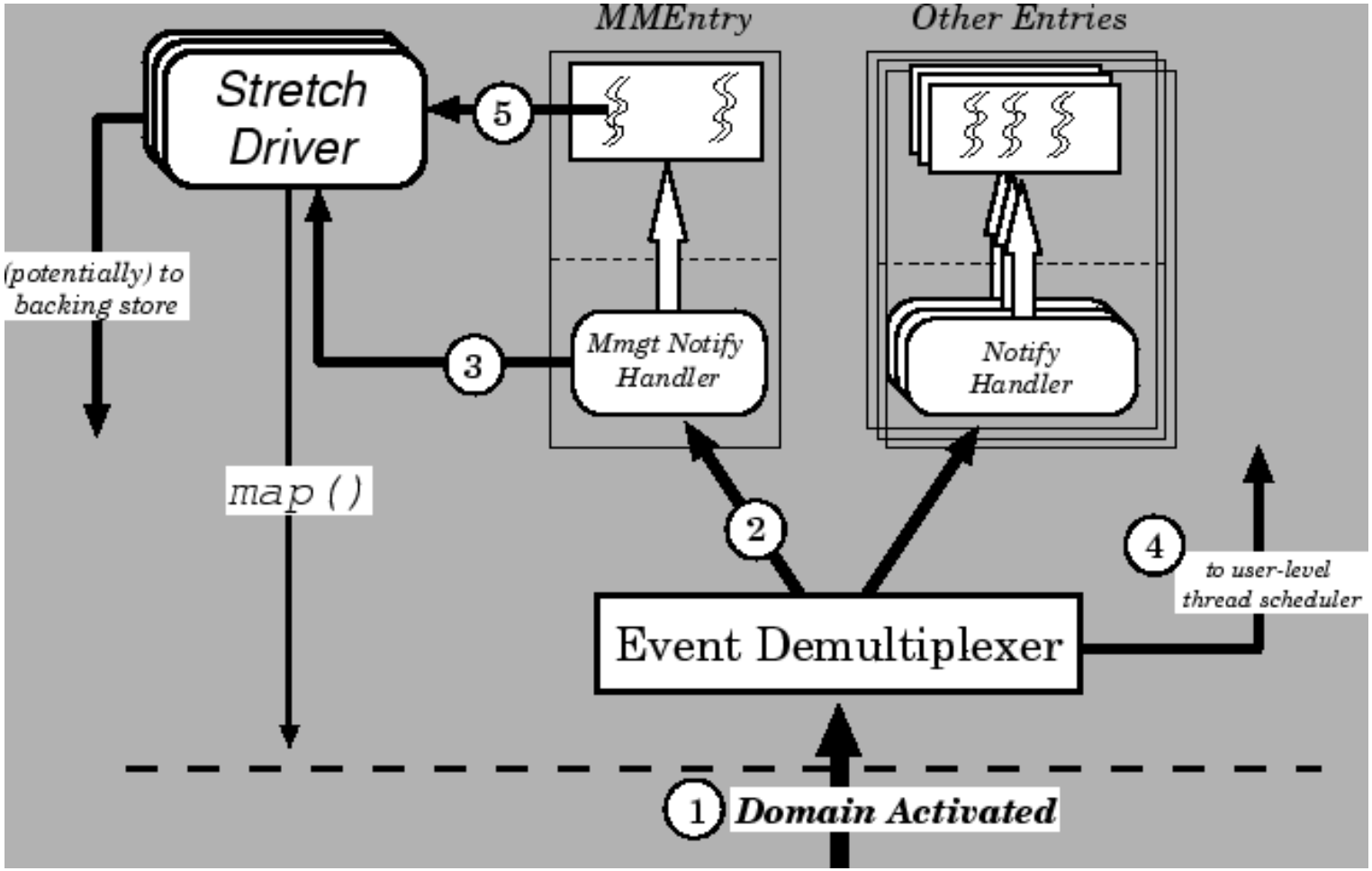
Zusammenfassung

- Nebenläufige QoS-Anwendungen
- Dynamische Ressourcenzuteilung
- Geringer QoS-Crosstalk

Selfpaging(1)



Selfpaging(2)



Memory Allocation