Betriebssysteme und Microkern

Andreas Steinel

16. September 2004

Überblick

Teil I

Einführung

Was ist ein Betriebssystem?

engl. *Operating System*:

- **Betriebsmittel:**
  CPU's, Speicher (physikalischer, virtueller)
- **Geräte:**
  - Eingabegeräte (Maus, Tastatur, ...)
  - Ausgabegeräte (Bildschirm, Drucker, ...)
  - Datenmedien (Festplatten, DVD, CD, Disketten, ...)
  - Multimediageräte
  - Netzwerkgeräte
- **API**
- **Tools**

Unterscheidung

- **technische Architektur**
  - Echtzeit
  - Time-Sharing
- **anwendungsorientiert**
  - Miniatursysteme (engl. *embedded systems*)
    jap. Echtzeitsystem ITRON ($>$ 3.000.000.000)
  - Einzel- und Mehrbenutzersysteme
    amerik. Time-SharingWindows ($>$ 500.000.000)
  - Grossrechner (physikalisch als auch virtuell)
    UNIX-derivate ($>$ 100.000.000)

Teil II

Betriebssysteme

Einführung

Kern (engl. *Kernel*) stellt Funktionalität:

- **monolithischer Ansatz**
- **microkernscher Ansatz**

Zugriff auf Funktionalität mittles **Systemaufruf** (*Systemcall*)

Eine Grundsatz-Diskussion über die beiden Ansätze führten A. Tanenbaum und L. Torvalds im usenet. Nähere Infos unter:

`http://www.dina.dk/~abraham/Linus_vs_Tanenbaum.html`

Kernpunkte

- Speicherverwaltung
- Prozess/Task und Threads
- Zwischenprozesskommunikation (*IPC*, Eyad Alkassar)
- Prozessumschaltung, Kontextwechsel
- Benutzer- und System-Modus (*User-*, *Kernel-Space*)
- Interrupt-Behandlung
- Abstraktion und Virtualisierung

Teil III

Monolith

ältere Monolithen

- Lange etabliert
- Alle Treiber vereint im Kernel-Space
- geringe Flexibilität
- hohe Komplexität
- geringe Portierbarkeit
- wenig Robust
- schlechte Sicherheitsmechanismen
- gute Performance

Beispiele:

- Ur-Systeme

Weiterentwicklung

Viele Ansätze:

- Schichtenmodell
- Kapselung von Diensten
- Client-Server Modell
- Sandboxing (Modula)

neuere Monolithen

- Schichtprinzip
- Alle Treiber vereint im Kernel-Space
- mittlere Flexibilität
- mittlere Komplexität
- gut Portierbarkeit
- weniger Robust
- schlechte Sicherheitsmechanismen
- gute Performance

Beispiele:

- Linux
- Solaris

Teil IV

Microkern

Microkern

- **Ende 80er**
- **Kapseln von Diensten in Servern** (Flexibilität, Komplexität)
- **Leichte Erweiterbarkeit** (Hybride)
- **Verteilte Systeme**
- **bessere Sicherheitsmerkmale** (Verifizierung)
- **Bereitstellung von Mechanismen** (IPC, keine Strategien)

Serverarchitektur

Client-Server-Architektur bietet:

- **Single-Server**
- **Multi-Server** (horizontal, linear)

(Gleich im Anschluss: Manuel Gorius)

- **Betriebssystemserver** (OS personality, Hybrid)
- **Platformunabhängig**
- **User-Space** (Prozess)
- **IPC** (syncron, asyncron, Stefan Knopp)

Kommunikation der Server

- **keine direkte Kommunikation**
- **lange/kurze Kommunikationswege**
- **Beispiel 1**
  L4 mit Linux als Server und Benutzer-Programm (Client)
  - **Client** $\xrightarrow{Systemaufruf}$ Microkern
  - Microkern $\xrightarrow{IPC}$ **Linux-Server**
  - **Linux-Server** $\xrightarrow{IPC}$ Microkern
  - Microkern $\xrightarrow{return}$ **Client**
- **Beispiel 2**
  Seitenfehler: Benutzer-Programm (Client), externer Pager
  - **Client** Zugriff auf Speicher
  - Microkern hat Seitenfehler $\xrightarrow{IPC}$ **Pager**
  - **Pager** Seite nachladen $\xrightarrow{IPC}$ Microkern
  - **Client** kann fortfahren

erste Generation

Mach, 1989 Carnegie Mellon University (CMU)

- basiert auf 4.2BSD
- 3. Release Microkern
- viele Architekturen unterstützen
- Netzwerktransparenz
- Parallelismus bei Programmen und im System
- größere Adressräume
- Basis für die Entwicklung weiterer Betriebssysteme
- langsam und zu gross

Besonderheiten:

- Port (Kommunikationskanal, Port Rights, Port Sets)
- Nachrichten (auch indirekt per Zeiger)
- Memory Objects (Speicher, Dateien, Pipes)
- I/O System im Kern

zweite Generation

Der Microkern **L4**, Jochen Liedtke 1995:

- externer Pager
- sehr kleiner Kern ($<$ 12KB)
- 3 Abstraktionen
- 7 Systemaufrufe
- reintegrieren von Servern
- leichte Zwischenprozesskommunikation (LIPC)
- Clans und Chiefs (Kommunikationsgruppen, alt)
- Flexpages (Frank Reolon)

Erweiterungen

- **Sigma0** Speicher-Dienst verwendet:
  - erster Prozess
  - alloziert gesamten Speicher
  - mapping 1:1
  - fungiert als Pager
- **Omega0** Interrupt-Dienst
- L^{4}Linux

Merkmale

- **Performance:**
  - Nachrichten (Overhead, langsam)
  - mehr Kontextwechsel
- **Konfiguration:** (Treiber)
- **Flexibilität:** (Diensteintegration)
- **Robustheit:**
  - Dienste neustarten
  - eigener Adressraum
- **Portierbarkeit:** (Server unabhängig)
- **Sicherheit:** (pro Server)

Zusammenfassung: Microkern

Ein **Mikrokern** umfasst folgende Merkmale:

- Auslagerung auf User-Space
- hohe Flexibilität
- niedrige Komplexität
- hohe Portierbarkeit der Dienste
- sehr Robust
- verbesserte Sicherheitsmechanismen
- schlechte Performance (1. Generation)
- mittlere bis gute Performance (2. Generation)

Teil V

Anhang

L4 Systemaufrufe

- ipc
- id_nearest
- fpage_unmap
- thread_switch
- thread_schedule
- lthread_ex_regs
- task_new

Einführung

Amoeba ist ein Betriebssystem, das darauf ausgelegt war, auf vielen Rechnern zu laufen und diese als einziges System erscheinen zu lassen. Es entstand 1981 an der *Vrije Universiteit* in Amsterdam. Entwickelt wurde es vorallem von *A. Tanenbaum*. Es basiert nicht auf Unix, sondern wurde komplett neu entwickelt. Ziel war es:

- **Transparenz**
- **komplett Verteiltes Betriebssystem** (ein System)
- **Gruppenkommunikation bei IPC**
- **Low-Level I/O-Kontrollen** Jedes eingebaute I/O-Gerät hat nicht wechselbaren Treiber der in den Kern gelinkt ist.

Architektur

Es wurden viele, damals neuen, Konzepte verwendet um einen Superechner zu erschaffen:

- **Prozessor-Pools**
  - Unterstützung von vielen CPU's
  - Bündelung von CPU's und Speicher in Prozessor-Pools
  - Optimierung von Message-Passing durch Shared Memory möglich
  - heterogene Systeme können in einem Pool existieren
  - Kindprozesse können auf anderen Architekturen laufen
- **Terminal**
- **spezialisierte Server** (Fileserver)