Vorbemerkungen

Das Betriebssystem ist die zentrale Systemsoftware eines Computers: Es verwaltet dessen Hard- und Softwarekomponenten und stellt vielfältige Dienste bereit, über die Benutzer[1] und Programmierer auf das System zugreifen können. Erst durch das Betriebssystem wird also die Hard- und Software eines Rechensystems benutzbar.

Dem Thema „Betriebssysteme“ kann man sich auf zwei verschiedene Arten nähern:

- Man kann den Standpunkt eines Benutzers annehmen und Fragen der folgenden Art stellen: „Wie kann ich meine Audioclips speichern und nutzen?“, „Welche Möglichkeiten zur Tabellenkalkulation bietet mein Betriebssystem?“ und so weiter. Im Mittelpunkt stehen hier also die Dienste der Benutzeroberfläche, die das Betriebssystem bereitstellt.
- Man kann den Standpunkt eines Programmierers annehmen und Fragen der folgenden Art stellen: „Wie ist es möglich, dass mein Rechner gleichzeitig mehrere Aufgaben ausführen kann?“, „Wie kann ich selbst solche nebenläufigen Programme schreiben?“ und so weiter. Im Mittelpunkt stehen hier also der interne Aufbau des Betriebssystems und die Dienste der Programmierschnittstelle, die das Betriebssystem bereitstellt.

Diese Kurseinheit folgt dem zweiten Ansatz, stellt also Dinge in den Vordergrund, die „unter der Motorhaube“ eines Betriebssystems liegen: Wenn Sie die Kurseinheit durchgearbeitet haben,

- kennen Sie die zentralen Konzepte und Mechanismen, die einem modernen Betriebssystem zugrunde liegen,
- wissen Sie, wie diese in der Betriebssystem-Software implementiert sind, und
- können Sie aus Ihren Programmen heraus auf Dienste eines Betriebssystems zugreifen.

1 Trotz vielfacher Versuche bietet die deutsche Sprache leider immer noch keine elegante Möglichkeit, die weibliche und männliche Form eines Worts durch einen gemeinsamen Oberbegriff abzudecken. Aus Gründen der Lesbarkeit wird daher in diesem Begleittext jeweils nur die männliche Form verwendet. Selbstverständlich sind aber auch immer die Benutzer*innen*, Programmierer*innen* usw. gemeint.

Grundlage der Kurseinheit ist der Begleittext, den Sie gerade lesen. Er gibt Ihnen eine Anleitung zum Durcharbeiten des Lehrbuchs „Grundkurs Betriebssysteme“ von P. Mandl [1]. Im Literaturteil finden Sie Hinweise für die weiterführende Lektüre.

Am Anfang des Lehrbuchs stehen in Kapitel 1 eine Einführung in die grundlegenden Aufgaben und Konzepte von Betriebssystemen sowie historische Betrachtungen. Kapitel 2 stellt verschiedene Ansätze zur Strukturierung der Betriebssystem-Software (ihrer „Architektur“) sowie zur nebenläufigen und verteilten Ausführung von Anwendungen vor. In Kapitel 3 wird die grundlegende Technik der „Interrupts“ diskutiert, die insbesondere bei „Systemaufrufen“, also dem Aufruf von Betriebssystemdiensten aus Programmen heraus, eine Rolle spielt. Kapitel 4 führt die Begriffe „Prozess“ und „Thread“ ein – Programme und Programmstücke, die vom Betriebssystem „nebenläufig“ (also zur selben Zeit) ausgeführt werden. In diesem Zusammenhang ist die effiziente Verwaltung des Prozessors (das „CPU-Scheduling“) entscheidend, womit sich Kapitel 5 befasst. Nebenläufige Prozesse und Threads müssen zeitlich koordiniert („synchronisiert“) werden, und sie tauschen untereinander Daten aus („kommunizieren“), was Thema von Kapitel 6 ist. Kapitel 7 behandelt die Verwaltung der Hierarchie aus Haupt- und Plattenspeicher, um Prozessen Speicherplatz für ihre Daten und ihren Code zur Verfügung zu stellen. Die folgenden Lehrbuchkapitel 8 und 9 zu Geräte- und Dateiverwaltung bzw. Betriebssystemvirtualisierung sind nicht Bestandteil dieser Kurseinheit.

Das Lehrbuch beschreibt die genannten Betriebssystemkonzepte zunächst allgemein und illustriert sie dann anhand von Linux mit der Programmiersprache C und Windows mit C# sowie anhand von Java. Diese Kurseinheit beschränkt sich auf Linux mit C und auf Java. Vorausgesetzt wird also, dass Sie Grundwissen in diesen beiden Programmiersprachen haben. Darüber hinaus sind Grundkenntnisse der Benutzeroberfläche eines Betriebssystems sowie der Hardwarearchitektur von Computern hilfreich.

Lesen Sie bitte nun das Vorwort des Lehrbuchs auf den Seiten V bis VII.

Inhaltsverzeichnis

2 Betriebssystemarchitekturen und Betriebsarten

Kapitel 2 diskutiert fundamentale Konzepte von Betriebssystemen

- zum Schutz von sicherheitskritischen Diensten und Daten vor unberechtigter Benutzung,
- zur Strukturierung der Software eines lokalen (d.h. auf einen einzelnen Computer beschränkten) Betriebssystems,
- zur nebenläufigen Ausführung von Aufträgen und
- zur Realisierung von Architekturen und Systemsoftware für vernetzte Computer.

2.1 Zugriffsschutz in Betriebssystemen

Lesen Sie bitte die Einleitung zu Kapitel 2 sowie das Unterkapitel 2.1, Seiten 25 bis 27 des Lehrbuchs.

In diesem kurzen Unterkapitel wird ein grundlegender und daher sehr wichtiger Sicherheitsaspekt angesprochen: Das Betriebssystem verwaltet den Computer und muss dafür unbeschränkten Zugriff auf alle seine Ressourcen besitzen. Anwendungsprogramme dürfen dagegen keinen solchen Vollzugriff haben, da das die Sicherheit des Systems verletzen würde. Die damit erforderliche strikte Trennung wird durch zwei Ausführungsmodi durchgesetzt und durch entsprechende CPU-Bits kontrolliert – das Betriebssystem wird im „Kernel Mode“ mit unbeschränkten Rechten ausgeführt, Anwendungssoftware dagegen im „User Mode“, in dem Daten und Code des Betriebssystem-„Kerns“ nicht zugreifbar sind. Der Kern (engl. „kernel“) ist der Teil des Betriebssystems, der direkt auf die Hardware zugreift und sicherheitsrelevante Daten speichert und verarbeitet.

Der Kern des Betriebssystems stellt an einer Schnittstelle („Kernschnittstelle“, „Systemschnittstelle“, „System Call Interface“, „Application Programming Interface API“) Dienste bereit, die von Anwendungsprogrammen genutzt (d.h. als Funktion aufgerufen) werden können – z.B. zur Erzeugung einer neuen Datei. Ruft ein Anwendungsprogramm eine Funktion dieser Schnittstelle auf, so erfolgt ein Übergang vom User Mode in den Kernel Mode. Es erhält damit zwar uneingeschränkte Rechte, muss dann aber genau das tun,

was die Kernfunktion vorschreibt. Ein Programm kann also im privilegierten Kernel Mode nicht beliebigen „selbstgestrickten“ Programmcode ausführen, sondern nur solchen Code, der vom Betriebssystemhersteller vorgegeben wurde. Die Sicherheit ist damit gewährleistet. Bei der Rückkehr aus der Funktion (also vom Betriebssystemkern in den Anwendungscode) werden der Ausführungsmodus zurückgesetzt und die Rechte wieder beschränkt.

2.2 Betriebssystemarchitekturen

Lesen Sie bitte Unterkapitel 2.2, Seiten 27 bis 35 des Lehrbuchs.

Sie haben in diesem Unterkapitel gesehen, dass das Betriebssystem als großes Softwaresystem auf unterschiedliche Arten strukturiert werden kann. Die weit verbreiteten Betriebssysteme Windows, Unix/Linux und Android haben geschichtete Architekturen. Hier bauen die Schichten so aufeinander auf, dass jede Schicht „nach oben“ eine Schnittstelle mit Diensten und Funktionen anbietet, die von der jeweils darüberliegenden Schicht benutzt werden. Die unterste Schicht setzt dabei auf die Hardware auf; die oberste Schicht stellt Schnittstellen/Dienste für den Benutzer bereit.

Bei allen drei Architektur-Beispielen habe Sie das wiedergefunden, was in Unterkapitel 2.1 allgemein besprochen wurde: Der Betriebssystemkern stellt an seiner Kernschnittstelle (hier jeweils „System Call Interface“ genannt) Dienste bereit. Der Aufruf eines solchen Diensts führt zum Übergang vom User Mode in den Kernel Mode – also zur Ausführung von festgelegtem Betriebssystemcode mit vollen Rechten.

Interessant ist bei Unix und Linux die „C Library“, die C-Schnittstellen für die Kernfunktionen (und damit ein „Application Programming Interface API“) bereitstellt. Man kann so aus C-Programmen heraus die Dienste des Betriebssystems wie normale C-Funktionen aufrufen. Sie werden später eine Reihe von Beispielen dazu sehen.

Eine alternative Architektur haben Mikrokern-Betriebssysteme, bei denen der „leichtgewichtige“ Kern nur die nötigsten Funktionen enthält, während alles andere in separate Module ausgelagert wird. Dieser Ansatz ist insbesondere für Plattformen mit beschränkten Ressourcen interessant. Zudem können die

Module relativ leicht auf verschiedene Rechnerknoten in einem Netz verteilt werden.

2.3 Klassische Großrechnerbetriebsarten

Lesen Sie bitte Unterkapitel 2.3, Seiten 35 bis 40 des Lehrbuchs.

In einem Computersystem stehen meist viele Aufträge zur Bearbeitung an, die auf dem einen Prozessor oder den wenigen Prozessoren ausgeführt werden müssen. Die Betriebsart eines Betriebssystems bestimmt insbesondere, wie viele Benutzer und wie viele Programmausführungen auf einem Computer gleichzeitig aktiv sein können. Charakteristisch für heutige Betriebssysteme ist der Mehrprogrammbetrieb, bei dem der Prozessor im meist raschen Wechsel zwischen den ausführbereiten Programmen hin- und hergeschaltet wird. Hierdurch entsteht der Eindruck der gleichzeitigen Ausführung. Wie man dies implementiert, welche Probleme dabei auftreten und wie man diese Probleme löst, wird in dieser Kurseinheit noch ausführlich behandelt.

2.4 Terminalserver-Betrieb *und*

2.5 Verteilte Verarbeitung

Lesen Sie bitte die Unterkapitel 2.4 und 2.5, Seiten 40 bis 47 des Lehrbuchs.

In heutigen Rechensystemen werden Dienste und Daten oft nicht nur vor Ort bei den Benutzern realisiert, sondern auch an geografisch weiter entfernten Stellen vorgehalten und über ein Kommunikationsnetz zugegriffen. Im einfachsten Fall liegen sie alle auf einem zentralen Server und werden von „dummen“ Benutzerterminals aus genutzt.

Flexibler ist ein „Verteiltes System“, bei dem sich die Funktionalität auf mehrere Rechnerknoten im Netz verteilt. Hier sind die Rollen von Dienstanbietern und -nutzern entweder klar voneinander getrennt („Client-Server-Architektur“), oder die Knoten können beides zugleich sein („Peer-to-Peer-Architektur“).

Ein wichtiges Ziel bei der Implementierung verteilter Systeme ist „Transparenz“: Benutzer (und möglichst auch die Anwendungssoftware) sollen nicht wissen müssen, wo im Netz ein Dienst realisiert ist. Auch sollten die Unterschiede in den lokalen Hardware- und Betriebssystemplattformen verborgen bleiben. Um dies zu erreichen, implementiert man eine zusätzliche Softwareschicht oberhalb der lokalen Betriebssysteme – die so genannte „Middleware“.

Lesen Sie die Unterkapitel 2.6 und 2.7 des Lehrbuchs nur bei Interesse.

Übungsaufgaben

Lösen Sie bitte die Übungsaufgaben zu Kapitel 2 des Lehrbuchs auf Seite 49. Die Aufgaben 2, 6, 11 und 15 müssen Sie nicht unbedingt bearbeiten. Die zugehörigen Lösungen finden Sie im Lehrbuch ab Seite 322.

Zusammenfassung

In diesem Kapitel haben Sie eine Reihe grundlegender Begriffe und Konzepte kennengelernt:

- **Zugriffsschutz:** Basis für die Sicherheitsarchitektur eines Computers ist die Unterscheidung von Kernel Mode und User Mode. Vertrauenswürdiger Code des Betriebssystemkerns wird im privilegierten Kernel Mode ausgeführt, potentiell gefährlicher sonstiger Code im eingeschränkten User Mode. Allerdings kann ein Anwendungsprogramm Dienste des Betriebssystems aufrufen und erhält dann während deren Ausführung temporär höhere Rechte.

- **Betriebssystemarchitekturen:** Die Software eines Betriebssystems ist oft hierarchisch oder auch nach dem Mikrokern-Prinzip strukturiert. Bei der hierarchischen Strukturierung setzen die einzelnen Schichten an ihren jeweiligen Schnittstellen aufeinander auf. Wichtig ist hier der Kern mit seiner Systemschnittstelle, die Dienste und Funktionen für die darüberliegenden Betriebssystemschichten und auch Anwendungsprogramme bietet.
- **Betriebsarten:** Anhand seiner Betriebsart steuert das Betriebssystem die Ausführung der Aufträge, die zur Bearbeitung anstehen. Moderne Betriebssysteme ermöglichen ein Multitasking (einen Mehrprogrammbetrieb), bei dem der Prozessor im raschen Wechsel zwischen den Aufträgen umgeschaltet wird und somit die Illusion der Gleichzeitigkeit entsteht.
- **Computer im Netz:** Konzepte wie Terminal Server oder verteilte Systeme ermöglichen es Benutzern und Anwendungsprogrammen auf Daten und Dienste zuzugreifen, die nicht durch den eigenen Computer, sondern durch einen anderen Computer im Rechnernetz angeboten werden.