Übungen zu Vorlesung
Betriebssysteme I
Sommersemester 1999

AG BS
Oliver Meyer/ Jan Peleska

Serie 4

IPC unter Windows-NT

Aufgabe 1: UNIX-Semaphore unter Windows-NT

Entwerfen und implementieren Sie eine Menge von Routinen, welche die Basissemantik der UNIX-Semaphore (nachzulesen mit „`man 2 semget`“ bzw. mit „`man 2 semop`“) unter Windows-NT nachbildet. Dies bedeutet, daß nicht die korrekte Reaktion auf die vollständige Menge der unter UNIX definierten Flags realisiert werden muß, jedoch sollen die Kommandos `semget` und `semop` soweit nachgebildet werden, daß die unter UNIX zur Verfügung stehende Möglichkeit, mehrere Semaphoren gleichzeitig anzusprechen, bereitgestellt wird. Auch die Auswirkungen des `IPC_NOWAIT`-Flags sollen umgesetzt werden. Außerdem sollen die Signaturen der UNIX-Aufrufe unverändert übernommen werden (es soll im Prinzip eine kleine Bibliothek implementiert werden, welche eine einfache Portierung von UNIX-Programmen mit Semaphore-Verwendung nach Windows-NT erlaubt). Insbesondere soll die Realisierung so vollständig sein, daß die folgende Aufgabe komplett mit den „neuen“ Semaphorkommandos gelöst werden kann.

Aufgabe 2: Producer-Consumer unter Windows-NT

In der Vorlesung wurde eine Lösung des Producer-Consumer-Problems skizziert, die mit drei Semaphoren arbeitet: Zwei davon sind zählende Semaphore, die den Vollheitsgrad bzw. den Leerheitsgrad des Puffers protokollieren während die dritte Semaphore binär ist und den kritischen Abschnitt – den Zugriff auf den Puffer zum Hineinschreiben/Auslesen – schützt. Beim Zugriff auf die Semaphore konnte ein Deadlock entstehen, wenn die `down`-Operationen in der falschen Reihenfolge ausgeführt wurden. Diese Fehlerquelle kann umgangen werden, wenn auf die zwei jeweils relevanten Semaphore (Füllstand + Mutex) gleichzeitig gewartet werden kann, wie es die UNIX-Semaphore erlauben.

Implementieren Sie eine Lösung des Producer-Consumer-Problems unter Windows-NT mit den in Aufgabe 1 entwickelten UNIX-Semantik Semaphoren. Dabei soll die vorgeschlagene Erweiterung – warten auf mehrere Semaphore gleichzeitig – genutzt werden. Der Puffer soll als Array im Shared-Memory realisiert werden; Produzent und Konsument sollen eigenständige Prozesse sein.

Abgabe: Bis Mo, 28. Juni 1999, in der Übung