ISIM: ASK + SO

Ćwiczenia 7: "Architektura systemów operacyjnych"

Należy przygotować się do zajęć czytając następujące rozdziały książek:

- "Modern Operating Systems" (czwarta edycja); Tanenbaum; 1.7, 2.1, 2.2

Należy być przygotowanym do wytłumaczenia **wytłuszczonych** haseł.

Zadanie 1

Jaka była motywacja do wprowadzenia **wieloprogramowych systemów wsadowych**? Opisz w jaki sposób wieloprogramowe systemy wsadowe wyewoluowały w **systemy z podziałem czasu**. Czy **interaktywne** systemy operacyjne muszą być wieloprogramowe? Jeśli nie to podaj przykład takich systemów.

Zadanie 2

Zdefiniuj pojęcie **wywłaszczania** i podaj mechanizmy niezbędne do jego implementacji. Wyjaśnij jak **algorytm rotacyjny** (ang. *round-robin*) może być użyty w implementacji **wielozadaniowości** z wywłaszczaniem. Jaka jest różnica między **planistą** (ang. scheduler), a **dyspozytorem** (ang. dispatcher)? Co to znaczy, że jądro jest wywłaszczalne – czemu jest to pożądana cecha?

Zadanie 3

Z jakich głównych **komponentów** składa się **monolityczne** jądro Linuksa – znajdź i przedstaw uproszczony diagram. Wymień kilka **interfejsów** funkcjonujących wewnątrz jądra. Co to znaczy, że jądro jest zorganizowane w **warstwy**? Jądro systemu NetBSD jest łatwo **przenośne** (ang. portable) dzięki dobrej implementacji **warstwy abstrakcji sprzętu** (ang. *hardware abstraction layer*). Czym różni się ona od **sterowników urządzeń**?

Zadanie 4

Podaj przykłady usług i funkcjonalności **jądra monolitycznego**, które mogą być zrealizowane jako procesy poziomu użytkownika w systemie operacyjnym z **mikrojądrem**. Jaki jest minimalny zestaw funkcjonalności, które musi implementować każde mikrojądro[1]. System Windows NT został pierwotnie zaprojektowany jako mikrojądro, lecz w trakcie rozwoju zdecydowano o migracji do architektury **jądra hybrydowego**. Dlaczego tak się stało? Jakie korzyści to przyniosło?

Zadanie 5

Proces to nie tylko przestrzeń adresowa i **kontekst**, ale też cały zestaw **zasobów**. Zapoznaj się z zawartością **bloku kontrolnego procesu** (**PCB**) jądra systemu Linux (`task_struct`) lub FreeBSD (`proc`). Na tej podstawie podziel na klasy rodzaje zasobów / informacji skojarzonych z procesem. Zauważ, że Linux nie definiuje osobnych struktur danych dla wątku i procesu, dlaczego? Czemu każde **zadanie** (ang. *task*) w systemie Linux posiada osobny **stos w przestrzeni jądra**?

Zadanie 6

Opisz **siedmiostanowy model procesu**. Wyjaśnij, które przejścia w automacie opisującym ten model są rezultatem działań podejmowanych przez (a) system operacyjny (b) proces użytkownika. Podaj jakie akcje wymuszają zmianę stanów? Które z decyzji po stronie systemu operacyjnego będą podejmowane przez **planistę krótkoterminowego** i **długoterminowego**.

Zadanie 7

W systemach uniksowych proces może zakończyć swoje działanie wywołując funkcję `exit()` lub otrzymując **sygnał**. Czym różni się sygnał **synchroniczny** od **asynchronicznego**? Wskaż podobieństwa i różnice między sygnałami a przerwaniami. Jakie zdarzenia / sytuacje mogą spowodować wysłanie sygnału kończącego działanie procesu? Czy są takie zdarzenia, które normalnie uznaje się za błąd programu, ale mogą służyć do implementacji pewnych funkcjonalności aplikacji? Jeśli tak, to podaj przykład.

---

[1] Według publikacji: On µ-Kernel Construction, Jochen Liedtke.

Zadanie 8

Czym różni się tworzenie procesów w systemie Linux i Windows NT? Rozważ wady i zalety obu rozwiązań. Wytłumacz na czym polega mechanizm **kopiowania przy zapisie** (ang. copy-on-write) ? Jakie flagi należy przekazać do wywołania systemowego **`clone()`** aby utworzyć (a) proces (b) wątek? Czy inne wariantu użycia `clone()` mogłyby mieć sens?

Zadanie 9

W książce (§2.2.1) omówiono strukturę prostego serwera WWW. Niejawnie zastosowano tam popularny **wzorzec projektowy** (ang. *design pattern*) zwany **pulą wątków** (ang. *thread pool*). Wyjaśnij jak się stosuje taki wzorzec do projektowania oprogramowania. Jakie zalety ma zastosowanie serwera opartego na puli wątków w porównaniu do serwera, który dla każdego nowego połączenia tworzy wątek?

Zadanie 10 [bonus]

Do implementacji wydajnych serwerów wykorzystuje się wzorzec projektowy *Proactor*. Wytłumacz działanie jego prostszego wariantu zwanego ***Reactor***'em na podstawie obsługi żądania GET protokołu HTTP. Wyjaśnij rolę wywołania systemowego `select()`, które służacy do demultipeksowania zdarzeń, oraz **nieblokujących** operacji wejścia-wyjścia.