Операционные системы. Практика 1
Мониторы

Михаил Пожидаев

14 сентября 2024 г.

Понятие монитора

Монитор — способ безопасного и вычислительноэффективного уведомления о наступлении события в параллельно выполняющихся процессах/потоках.

Примеры применения мониторов:

1. Выполнение части действий в другом контексте безопасности.
2. Ожидание освобождения ресурса.
3. Ожидание команды пользователя от потока-обработчика устройства ввода.

Составные части монитора

Монитор имеет смысл только в контексте двух параллельно выполняющихся потоков, которые условно назовём поток-поставщик и поток-потребитель. В общем случае потребителей может быть несколько.

Элементы монитора, общие для обоих потоков:

1. Формальный критерий проверки наступления события.
2. Разделяемые данные, передаваемые от потока-поставщика к потоку-потребителю.
3. Инструмент блокировки участков кода, имеющих доступ к разделяемым данным.
4. Возможность остановить выполнение потока-потребителя до получения уведомления от потока-поставщика.

Мьютексы

Функции мьютекса:

- гарантия доступа к разделяемым данным только в одном потоке;
- вычислительно эффективное ожидание возможности входа в защищённый фрагмент.

Схема функции-поставщика

1. Блокировка мьютекса.
2. Выполнение действий необходимых для наступления события.
3. Уведомление потока-потребителя о наступившем событии.
4. Освобождение мьютекса.

Схема функции-потребителя

1. Блокировка мьютекса.
2. Проверка наступления события.
3. Ожидание наступления события с временным освобождением мьютекса.
4. Обработка события.
5. Окончательное освобождение мьютекса.

Функция-поставщик для Java

```
void synchronized provide()
{
  if (ready)
    return;
  ready = true;
  notify();
}
```

Функция-потребитель для Java

```
void synchronized consume()
{
  while(!ready)
    wait();
  ready = false;
}
```

Переменные для pthread

```
pthread_cond_t cond1 = PTHREAD_COND_INITIALIZER;
pthread_mutex_t lock = PTHREAD_MUTEX_INITIALIZER;
int ready = 0;
```

Функция-поставщик для pthread

```
pthread_mutex_lock(&lock);
if (ready == 1)
{
  pthread_mutex_unlock(&lock);
  continue;
}
ready = 1;
printf("provided\n");
pthread_cond_signal(&cond1);
pthread_mutex_unlock(&lock);
```

Функция-потребитель для pthread

```
pthread_mutex_lock(&lock);
while (ready == 0)
{
  pthread_cond_wait(&cond1, &lock);
  printf("awoke\n");
}
ready = 0;
printf("consumed\n");
pthread_mutex_unlock(&lock);
```

Спасибо за внимание!

E-mail: msp@luwrain.org
Канал в Телеграм: https://t.me/MarigostraRu