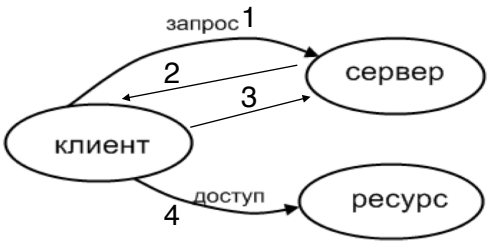
Лекция №7

Схема клиента и сервера предполагающая отделение ресурса от сервера



1. Запрос к серверу со стороны клиента
2. Разрешение сервера на обращение к ресурсу
3. Взаимодействие с ресурсом
4. Сообщение об освобождении ресурса

ТУТ ДОЛЖЕН БЫТЬ ФОРМАТ ТАБЛИЦЫ РЕСУРСОВ

Если ресурс может выделяться по частям (дискретный) нужно хранить количество оставшегося ресурса, если он неделимый то просто флаг занятости.

Очередь к ресурсу – массив m на n-1, где m – количество ресурсов, а n – количество клиентов

Информационные структуры сервера:
1. Таблица ресурсов (формат самостоятельно)
2. Очереди процессов к ресурсам (массив m*n-1)

Форматы сообщений при взаимодействии клиентов и сервера

Необходим идентификатор ресурса, который необходимо захватить/освободить.

Если ресурс дискретный (может быть разделен на части), то необходимо указать какая его часть должна быть захвачена/освобождена.

Идентификатор клиента (для захвата, для освобождений подумать зачем)

Запрос на захват ресурса

Захват			
Тип сообщения	Идентификатор ресурса	Количество запрашиваемого ресурса для дискретных	Идентификатор клиента, запросившего
Освобождение			
Тип сообщения	Идентификатор ресурса	Количество освобождаемого ресурса для дискретных	

В случае если клиент одновременно может запросить несколько ресурсов, тогда сообщение от сервера должно содержать идентификатор ресурса, доступ к которому разрешён.

Алгоритм функционирования сервера при получении запросов:
1. Идентификация положения ресурса в таблице
2. Проверка состояния ресурсов (занят/свободен), достаточно ли ресурса для удовлетворения запросов
3. В случае отсутствия достаточного количества ресурса – размещение процесса в очереди с указанием количества запрошенного им ресурса
4. Иначе состояние ресурса поменять и сформировать сообщение клиенту с указанием возможности обратиться к ресурсу

Алгоритм функционирования сервера при интерпретации сообщения об освобождении ресурса:

1. Идентификация положения ресурса в таблице
2. Изменение состояния ресурса (Свободное + Освобождаемое)
3. Определить наличие процессов в очереди на этот ресурс (простейшая стратегия – первый в очереди)
   - *3.1.*Извлечение первого процесса из головы очереди
   - *3.2.*Определение достаточности свободного ресурса для удовлетворения запроса этого процесса
   - *3.3.*В случае достаточного количества – оповещение соответствующего клиента и удаления его из головы очереди, изменить состояние доступности ресурса

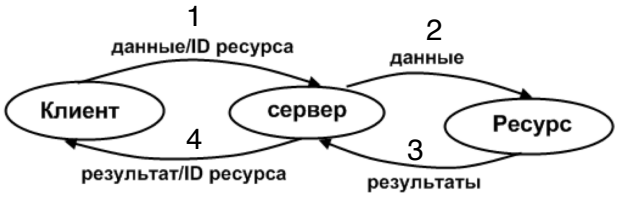
     Недостатками рассмотренной стратегии обработки очереди являются: блокирование процессов с малым количеством запрошенного ресурса в очереди тем процессом, которому ресурса недостаточно

     Альтернативным вариантом стратегии является идентификация в очереди того процесса, количество запрошенного ресурса которого соответствует количеству свободного ресурса

Недостатком рассмотренной схемы является блокирование клиента в ожидании требуемого для него ресурса.

Альтернативным вариантом является когда сервер выступает в роли посредника между клиентом и ресурсом.

Схема клиент-серверного взаимодействия при реализации сервера как посредника



1. Сообщение с указанием идентификатора ресурса и тех данных, которые должны быть им обработаны
2. Сообщение с данными, передаваемыми ресурсу на обработку (сервер идентифицирует процесс (хранит ID), данные которого он передал ресурсу для обработки)
3. Результаты обработки данных
4. Ретрансляция результатов обработки конкретному клиенту

В случае занятости ресурса данные буферизируются в ожидании освобождения ресурса. В этом случае буферизируются в очереди к ресурсу не только ID процесса, но и данные которые должны быть обработаны.

Алгоритмы распределённой синхронизации

1. Распределённые семафоры
2. Распределённые взаимоисключения

С реализацией механизма распределённых семафоров связаны понятия:

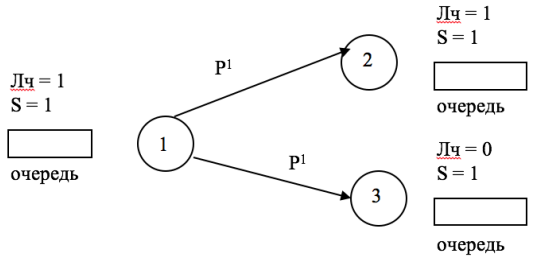
1. Логические часы – счётчик событий, связанных с взаимодействием распределено выполняющихся процессов (любая передача сообщений приводит к изменению значения логических часов (+1))
2. Локальная очередь процессов к разделяемому ресурсу (для каждого ресурса своя)

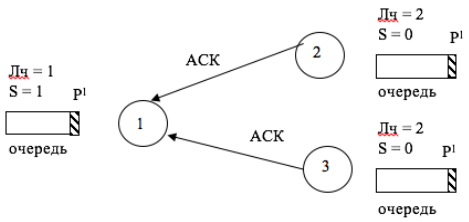
Форматы сообщений

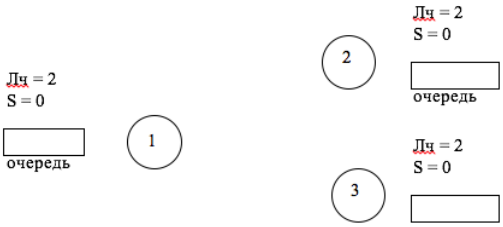
Идентификатор процесса, являющегося источником сообщения	Идентификатор вида операции P или V	Метка времени (значение локальных часов процесса, являющегося источником сообщения)

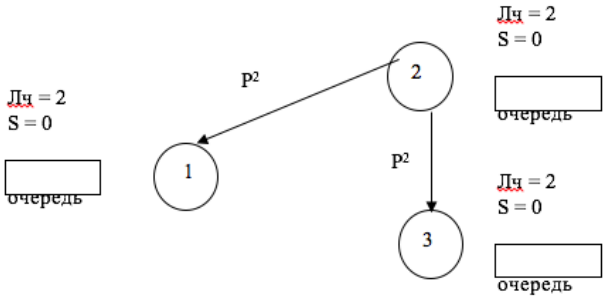
Обозначим формат соответствующей операции в виде:
P^i_j, V^i_j, где i – это значение логических часов, j – идентификатор процесса, являющегося источником данного сообщения

Начальное состояние: логические часы всех процессов равны 1, все ресурсы свободны (семафор 1).

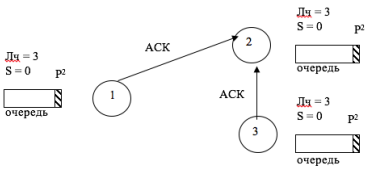






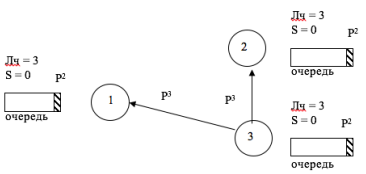


79

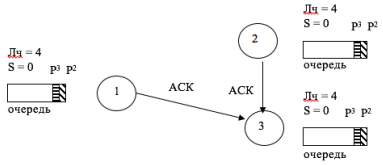


Т.к. S = 0, то процесс 2 блокируется в ожидании разрешения доступа к ресурсу. Сообщение P^2 остается в очереди.

6. Запрос процессом 3 ресурса. Отправка сообщения с идентификатором P-операции.

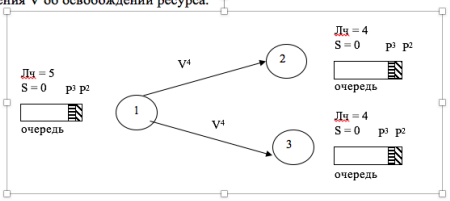


7. Размещение P сообщения в очереди. Подтверждение принятия P сообщения

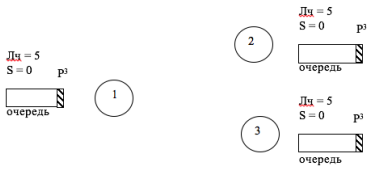


80

8. Обработка процессом 3 значения переменной S. Посылка процессом 1 сообщения V об освобождении ресурса.



9. Обработка очереди: изменение значения переменной S на 1, выборка операции P^1 из очереди, изменение переменной S в значение 0. Разрешение доступа процессу, сгенерировавшему операцию P (процесс 2) доступа к ресурсу. Удаление операции P^1 из очереди.



Рисунки 15.1- 15.9 - Пример реализации алгоритма взаимодействия процессов с использованием распределенных семафоров

15.2 Алгоритм передачи маркера

Недостатком распределенных семафоров является большое количество сообщений , передаваемых внутри кластера. Алгоритм передачи маркера позволяет уменьшить число сообщений. Алгоритм передачи маркера также предназначен для синхронизации доступа распределения выполняющихся процессов к общему ресурсу. В данном случае маркер – это сообщение специального вида, которое предназначено для передачи разрешения на доступ к ресурсу.

Особенностью построения алгоритма передачи маркера является то, что с каждым из распределено выполняющихся процессов, реализующих вычисления, сопоставлен специальный вспомогательный процесс, реализующий передачу маркера. Вспомогательные процессы образуют кольцо, т.е. процесс 1 передает

Алгоритм синхронизации разделения ресурсов посредством передачи маркера

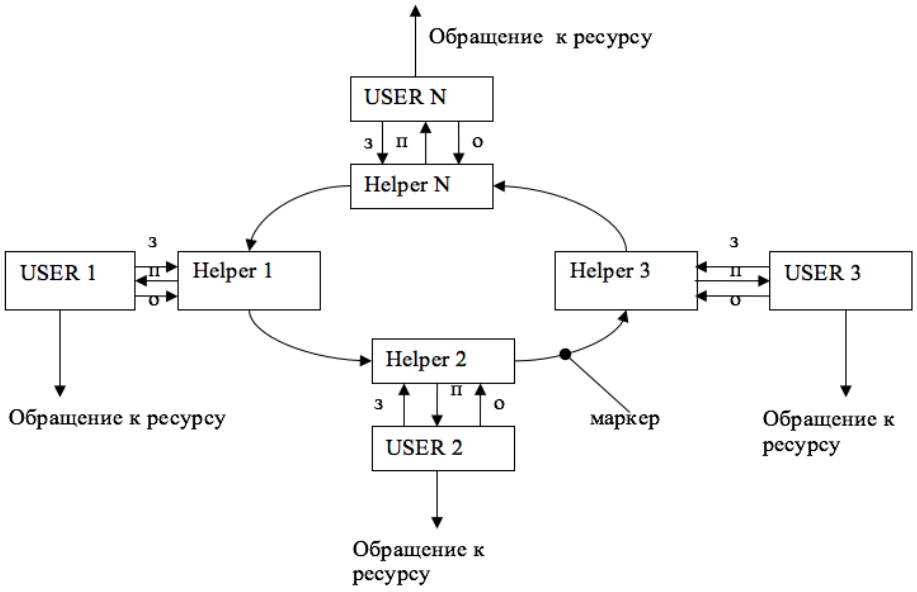


Рисунок 15.2 - Пример реализации алгоритма передачи маркера USER – вычислительный процесс, Helper – вспомогательный процесс, З – запрос ресурса, П – подтверждение возможности использования ресурса, О – сигнал освобождения ресурса

Маркер И !Запрос -> передача маркера
!Маркер И Запрос -> блокирование юзера
Маркер И Запрос -> доступ к ресурсу