

## Лабораторная работа №3 «Скалярное время Лэмпорта»

### Введение

В распределенных системах отсутствуют глобальные часы, отсчитывающие общее для всех процессов время, и к показаниям которых процессы могли бы получать мгновенный доступ. Существуют различные способы синхронизации физических часов: радио часы (WWV), протокол NTP и его производные — однако максимальная точность синхронизации составляет порядка нескольких миллисекунд, что для ряда приложений недостаточно.

Поэтому, как было показано на примере банковской системы в лабораторной работе №2, для правильной работы распределенных приложений необходимо определить отношение «произошло раньше», связывающее события процессов между собой, не опираясь на понятие единого физического времени. Поэтому распределенные приложения используют различные варианты логического времени.

### Исходные данные

Следует использовать исходные данные из лабораторной работы №2.

### Задание

В реализации банковской системы из лабораторной работы №2 необходимо заменить физическое время на скалярное время Лэмпорта. Для обмена отметками времени процессы должны использовать поле *s\_local\_time* структуры *MessageHeader*, которое должно содержать показания часов процесса-отправителя на момент отправки сообщения. Получение текущей отметки времени осуществляется посредством вызова функции *get\_lamport\_time()*, которую необходимо реализовать.

Логические часы инициализируются нулем. Считать, что у процессов отсутствуют внутренние события, т.е. линейному упорядочиванию подлежат только события отправки и получения сообщений. Перед выполнением любого события процесс увеличивает показания своих логических часов на единицу. Так при отправке сообщения показания часов сначала увеличиваются, а уже потом в сообщение вкладывается отметка времени. Отправка группового сообщения посредством функции *send\_multicast()* продвигает время на единицу вне зависимости от числа отправленных сообщений. В случае, когда два события имеют одинаковые временные метки, линейный порядок определяется на основе идентификаторов процессов, в которых произошли данные события.

В отличие от лабораторной работы №2 при подсчете полной суммы денег в каждый момент времени необходимо учитывать состояние каналов между процессами, т.е. сохранять информацию о переводах, которые были отправлены, но еще не были получены. Для этого необходимо заполнять поле *s\_balance\_pending\_in* структуры *BalanceState*. Обратите внимание, что в виду синхронности протокола данная задача является упрощенным вариантом задачи подсчета полной суммы, рассмотренной на лекции.

Никакие другие изменения в банковскую систему вносить не требуется.

### Требования к реализации и среда выполнения

Реализацию необходимо выполнить на языке программирования Си с использованием предоставленных заголовочных файлов и библиотеки из архива [pa23\\_starter\\_code.tar.gz](http://pa23.starter.code.tar.gz) из второй лабораторной работы.

Работа присылается в виде архива с именем *pa3.tar.gz*, содержащим каталог *pa3*. Все файлы с исходным кодом и заголовки должны находиться в корне этого каталога. Среда выполнения — Linux (Ubuntu версии  $\geq 12.10$ ). При автоматической проверке используется следующая команда: *gcc -std=c99 -Wall -pedantic \*.c -L. -lruntime*. При наличии варнингов работа не принимается. При успешном выполнении запущенные процессы не должны использовать *stderr*, код завершения программы должен быть равен 0.