Задания на практические работы по курсу «Распределённые системы»

Цель работы: изучение методов алгоритмического описания распределённых алгоритмов

Порядок выполнения работы

1. Выполнить словесное описание алгоритма в терминах приёма и отправки сообщений. Для наглядности можно использовать поясняющие диаграммы.

2. Выполнить точное описание алгоритма из п.1 в терминах акторной модели:
   - определить необходимые типы акторов, их состояние, алгоритмы обработки поступающих сообщений;
   - определить необходимые типы сообщений;
   - определить начальное состояние вычислений (имеющиеся вначале работы алгоритма акторы; отправленные, но не принятые начальные сообщения);
   - определить критерий остановки (достижения требуемого состояния системой акторов).
   -

3. Построить программную модель алгоритма п.2 с использованием каркаса Templet

```
struct engine;
struct proc;
struct chan;

struct engine{
      std::vector<chan*> ready;
};

struct proc{
      virtual void recv(chan*, proc*);
};

struct chan{
      proc*p;
      bool sending;
};

inline void send(engine*e, chan*c, proc*p)
{
      if (c->sending) return;
      c->sending = true;
      c->p = p;
      e->ready.push_back(c);
}

inline bool access(chan*c, proc*p)
{
      return c->p == p && !c->sending;
}

inline void run(engine*e, int n = 1)
{
      size_t rsize;
      while (rsize = e->ready.size()){
            int n = rand() % rsize;    auto it = e->ready.begin() + n;
            chan*c = *it;   e->ready.erase(it); c->sending = false;
```

```
            c->p->recv(c, c->p);
        }
}
```

Определите акторы реализации алгоритма п.2 как расширения `struct proc`, а сообщения – как расширения `struct chan`.

4. Подготовить тестирующий код. Выполнить тестирование программной реализации алгоритма п.3.

5. Оформить письменный отчет.

Примерные варианты заданий

1. Алгоритм (по выбору) использующий распределённый портфель задач.
2. Распределённое умножение матриц с использованием циркуляции столбцов.
3. Нахождение простых чисел методом решета Эратосфена, реализованного в виде конвейера из фильтрующих процессов.
4. Вычисление простых чисел в архитектуре управляющий-рабочие (см. Г. Эндрюс с.266).
5. Сортировка массива из n чисел при помощи конвейера из n процессов. Каждый из процессов оперирует 2 числами: следующим введенным и текущим минимумом.
6. Реализовать алгоритм сортировки слиянием, для этого реализовать актор merge, объединяющий два отсортированных потока чисел в один отсортированный поток.
7. Кольцевой алгоритм голосования. Считать, что отказавший процесс пропускает через себя сообщение ГОЛОСОВАНИЕ, не добавляя себя к списку претендентов; процессы, запускающие голосование выбираются произвольно.
8. Алгоритм забияки. Считать, что отказавший процесс посылает в ответ на сообщение ГОЛОСОВАНИЕ специальное сообщение ОТКАЗ; процессы, запускающие голосование выбираются произвольно.
9. Двухфазный протокол подтверждения транзакции.
10. Трёхфазный протокол подтверждения транзакции.