Примеры многопоточных программ (потоки в ОС и RTL)

Востокин Сергей Владимирович

План

- Пример 0: применение Taskbag
- Модель потоков
- Пример 1: Реализация потоков в Windows
- Пример 2: Реализация потоков в Unix
- Пример 3: Реализация потоков в C++

Пример 0: применение Taskbag

Описание примера

- Цель: демонстрация применения шаблона Taskbag для распараллеливания сортировки
- Исходный алгоритм: алгоритм быстрой сортировки Хоара
- Реализации шаблона Taskbag
  - Потоковый пул Windows API
  - Потоковый пул POSIX Threads

Исходный алгоритм

```
void qSort(int* a, int size)
{
    long i = 0, j = size-1;
    int temp, p;

    p = a[ size>>1 ];
    do {
        while ( a[i] < p ) i++;
        while ( a[j] > p ) j--;

        if (i <= j) {
            temp = a[i]; a[i] = a[j]; a[j] = temp;
            i++; j--;
        }
    } while ( i<=j );

    if ( j > 0 ) qSort(a, j+1);
    if ( size > i ) qSort(a+i, size-i);
}
```

Структура кода, к которой требуется привести исходный алгоритм

```
struct task{/*1*/};
struct bag{/*2*/};
void proc(task*){/*3*/}
bool if_job(bag*){/*4*/return false;}
void put(task*,bag*){/*5*/}
void get(task*,bag*){/*6*/}

int _tmain(int argc, _TCHAR* argv[])
{
    task t; bag b;
    while(if_job(&b)){
        get(&t,&b); proc(&t); put(&t,&b);
    }
    return 0;
}
```

Принцип решения

- Задача (task) – сортировка подмассива
- Алгоритм разделяется на 2 стадии:
  - Формирование списка задач
  - Обработка списка
- Задачи формируются за счёт остановки рекурсии, когда разделяемый массив достигает некоторого заданного размера

Структуры данных

```
struct task{
        int*a; int size;
};

struct bag{
        queue<task> taskQueue;
};
```

Функции (1/2)

```
void proc(task*t){
    qSort(t->a,t->size);
}

bool if_job(bag*b){
    return !b->taskQueue.empty();
}
```

Функции (2/2)

```
void put(task*,bag*){
        // код не требуется
}

void get(task*t,bag*b){
        *t=b->taskQueue.front();
        b->taskQueue.pop();
}
```

Формирование списка задач

```
void qSort0(bag*b, int*a, int size)
{
    long i = 0, j = size-1;
    int temp, p;

    p = a[ size>>1 ];
    do {
        while ( a[i] < p ) i++;
        while ( a[j] > p ) j--;

        if (i <= j) {
            temp = a[i]; a[i] = a[j]; a[j] = temp;
            i++; j--;
        }
    } while ( i<=j );

    if ( j > 0 )    if(j+1 < T)    b->taskQueue.push(task(a,j+1));     else qSort0(b, a, j+1);
    if ( size > i ) if(size-i < T) b->taskQueue.push(task(a+i,size-i));else qSort0(b, a+i, size-i);
}
```

Запуск алгоритма

```
int _tmain(int argc, _TCHAR* argv[])
{
    task t; bag b;

    for(int i=0;i<N;i++) arr[i]=rand()%N;

    qSort0(&b, arr, N);

    while(if_job(&b)){
        get(&t,&b); proc(&t); put(&t,&b);
    }

    for(int i=0;i<N;i++) cout<<arr[i]<<' ';

    return 0;
}
```

Перенос блоков кода в шаблон Taskbag (1/4)

```
class TaskBag:public TEMPLET::TBag{
public:
    class TaskBagTask:public TBag::Task{
    public:
        TaskBagTask():TBag::Task(){}
        virtual~TaskBagTask(){}

        void send_task()  {}
        void recv_task()  {}
        void send_result(){}
        void recv_result(){}

        task t;//<----------------
    };
```

Перенос блоков кода в шаблон Taskbag (2/4)

```
public:
    TaskBag(int num_prc,int argc, char* argv[]):TBag(num_prc,argc,argv){
        for(int i=0;i<N;i++) arr[i]=rand()%N; //<----------
        qSort0(&b, arr, N);                   //<----------
    }
    virtual ~TaskBag(){}
    TBag::Task* createTask(){return new TaskBagTask;}
```

Перенос блоков кода в шаблон Taskbag (3/4)

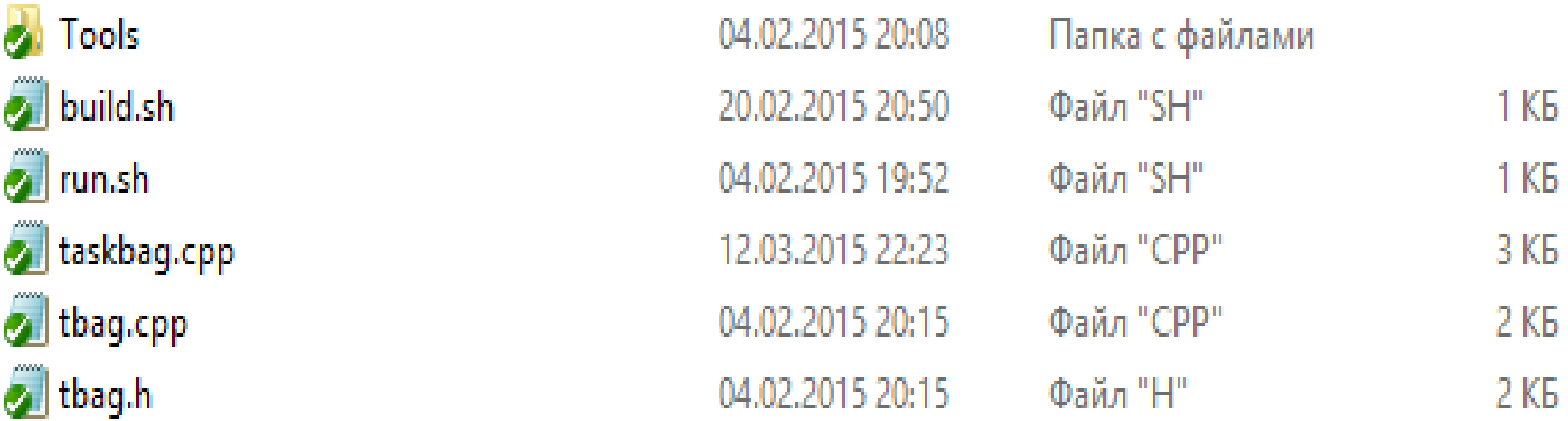
```
queue<task> taskQueue;//<-------------

bool if_job(){return !taskQueue.empty();}//<---------
void put(Task*t){}//<-------------
void get(Task*t){TaskBagTask* mt=(TaskBagTask*)t;
    tm->t=taskQueue.front(); //<-------------
    taskQueue.pop(); //<-------------
}
```

Перенос блоков кода в шаблон Taskbag (4/4)

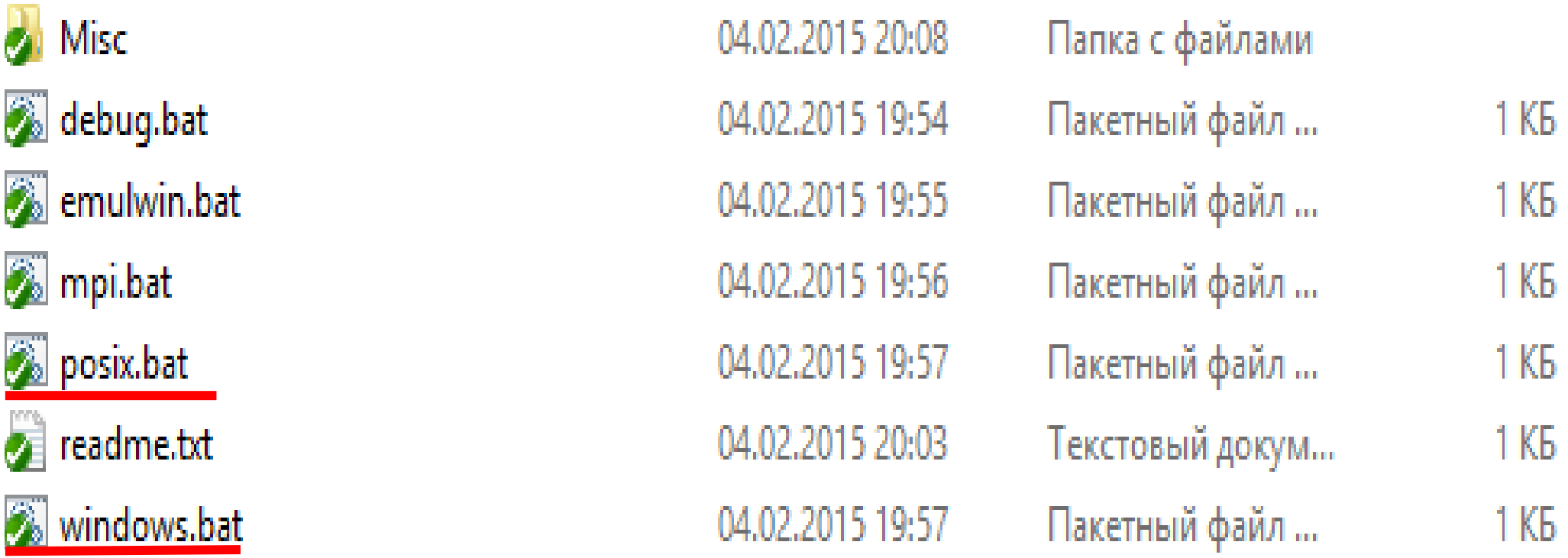
```
void proc(Task*t){TaskBagTask* mt=(TaskBagTask*)t;
    qSort(mt->t.a,mt->t.size); //<------
}
```

Выбор способа выполнения (1/2)



Tools	04.02.2015 20:08	Папка с файлами	
build.sh	20.02.2015 20:50	Файл "SH"	1 КБ
run.sh	04.02.2015 19:52	Файл "SH"	1 КБ
taskbag.cpp	12.03.2015 22:23	Файл "CPP"	3 КБ
tbag.cpp	04.02.2015 20:15	Файл "CPP"	2 КБ
tbag.h	04.02.2015 20:15	Файл "H"	2 КБ

Выбор способа выполнения (2/2)



Misc	04.02.2015 20:08	Папка с файлами	
debug.bat	04.02.2015 19:54	Пакетный файл ...	1 КБ
emulwin.bat	04.02.2015 19:55	Пакетный файл ...	1 КБ
mpi.bat	04.02.2015 19:56	Пакетный файл ...	1 КБ
posix.bat	04.02.2015 19:57	Пакетный файл ...	1 КБ
readme.txt	04.02.2015 20:03	Текстовый докум...	1 КБ
windows.bat	04.02.2015 19:57	Пакетный файл ...	1 КБ

Модель потоков

Псевдокод API многопоточного исполнения

```
struct thread{void(*tfunc)(thread*);};

struct mutex{};     //конкурентная синхронизация
void lock(mutex*);
void unlock(mutex*);

struct event{};     //условная синхронизация
void wait(event*,mutex*);
void notify(event*);
```