LBS位置相关服务
实验环境

系统环境

- 服务器
  - 123.57.237.171
  - 用户名: st1~st8
  - 密码: 12345678

系统环境

- 登陆
  - mac/linux
    - ssh st1@123.57.237.171
  - windows
    - 通过SSH Secure Shell Client登陆

    http://ultra.pr.erau.edu/~jaffem/tutorial/SSH_secure_shell_client.htm

系统环境

- 数据拷贝
  - **mac/linux**
    - 从服务器拷贝到本机：
      - scp st1@123.57.237.171:~/lbs .
    - 从本机拷贝到服务器
      - scp –r lbs st1@123.57.237.172:~/
  - **windows**
    - 通过SSH Secure Shell Client上下传数据

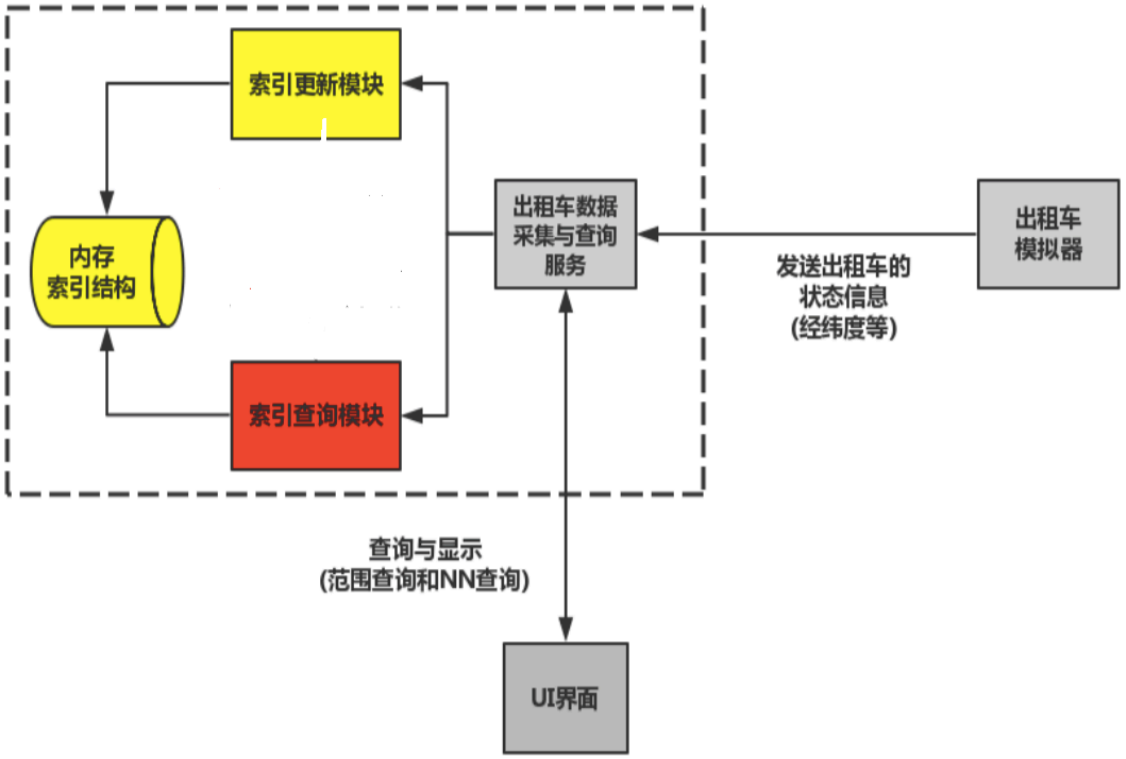
系统环境

- 开发环境
  - **mac/linux**
    - vim
    - gdb
  - **windows**
    - dev-c++

    http://www.bloodshed.net/dev/devcpp.html

系统环境

- 模拟器起停
  - 启动
    - cd lbs
    - ./run-simulator.sh
  - 停止
    - cd lbs
    - ./stop-simulator.sh
- lbs服务运行
  - 启动
    - cd lbs
    - ./run-server.sh
  - 停止
    - cd lbs
    - ./stop-server.sh



工程代码结构

- 代码结构


common 公共模块
conf 配置文件
simulator 模拟器
server 服务器
grid 网格索引与查询模块

位置更新

- 需要什么模块
  - lbs_hashtable
  - lbs_grid

位置更新

- **Hashtable的结构定义**

```
#include <pthread.h>

#include "server/grid/lbs_defs.h"

typedef struct lbs_hashnode_s {
  // 链表
  lbs_queue_t queue;
  // 节点
  lbs_mov_node_t* mov_node;
  // cell id
  int cell_id;
} lbs_hashnode_t;

typedef struct lbs_hashtable_s {
  // 锁
  pthread_mutex_t mutex;
  // 已占用
  int size;
  // 容量
  int capacity;
  // 哈西-链地址
  lbs_hashnode_t* hash_nodes;
} lbs_hashtable_t;
```

位置更新

- **Hashtable函数定义**

```
// 初始化
int lbs_hashtable_init(lbs_hashtable_t* lbs_hashtable);
// 销毁
int lbs_hashtable_destroy(lbs_hashtable_t* lbs_hash_table);
// 设置
int lbs_hashtable_set(lbs_hashtable_t* lbs_hashtable, uint32_t id, lbs_mov_node_t* lbs_mov_node, int cell_id);
// 提取
lbs_hashnode_t* lbs_hashtable_get(lbs_hashtable_t* lbs_hash_table, uint32_t id);
```

位置更新

- Grid的结构定义

```
#include <pthread.h>

#include "server/grid/lbs_defs.h"
#include "server/grid/lbs_hashtable.h"

typedef struct lbs_cell_s {
  // dammy node
  lbs_mov_node_t dammy_node;
  // 锁
  pthread_mutex_t mutex;
} lbs_cell_t;

typedef struct lbs_grid_s {
  // row num of grid
  int row_num;
  // col num of grid
  int col_num;
  // cell width
  double cell_width;
  // cell height
  double cell_height;
  // grid lon minimum value
  double lon_min;
  // grid lat minimum value
  double lat_min;
  // 哈西表
  lbs_hashtable_t hash_table;
  // 所有的Cells
  lbs_cell_t* cell;
} lbs_grid_t;
```

位置更新

- Grid的函数定义

```
// 网格的初始化
int lbs_grid_init(lbs_grid_t* lbs_grid, double lon1, double lon2, double lat1, double lat2, int row_num, int col_num);
// 网格的删除
int lbs_grid_destroy(lbs_grid_t* lbs_grid);
// 更新移动位置
int lbs_grid_update(lbs_grid_t* lbs_grid, double lon, double lat, uint64_t timestamp, uint32_t id);
// 计算Cell Row
int lbs_grid_cell_row(lbs_grid_t* lbs_grid, double lat);
// 计算Cell Col
int lbs_grid_cell_col(lbs_grid_t* lbs_grid, double lon);
// 计算Cell Id
int lbs_grid_cell_id(lbs_grid_t* lbs_grid, int cell_row, int cell_col);
// 计算row和col
void lbs_grid_cell_row_col(lbs_grid_t* lbs_grid, int cell_id, int* cell_row, int* cell_col);
// 获取Cell Id里面的Cell
lbs_cell_t* lbs_grid_cell(lbs_grid_t* lbs_grid, int cell_id);
```