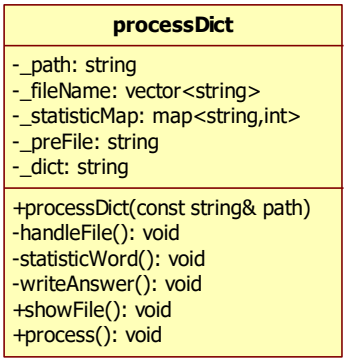
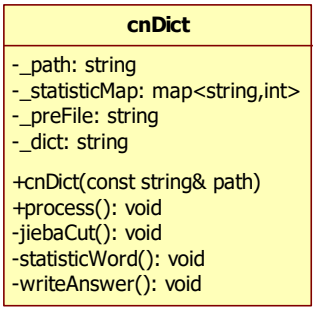
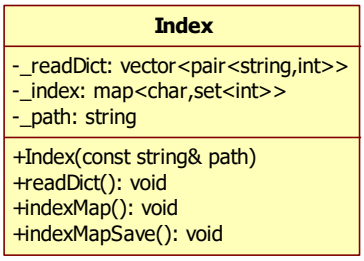
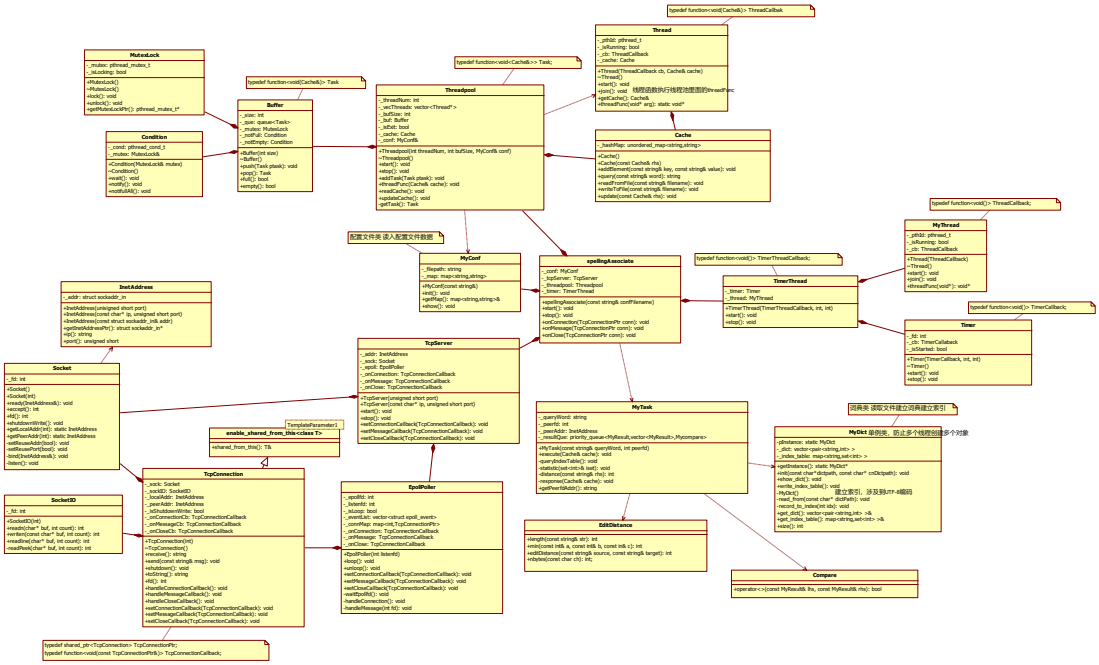
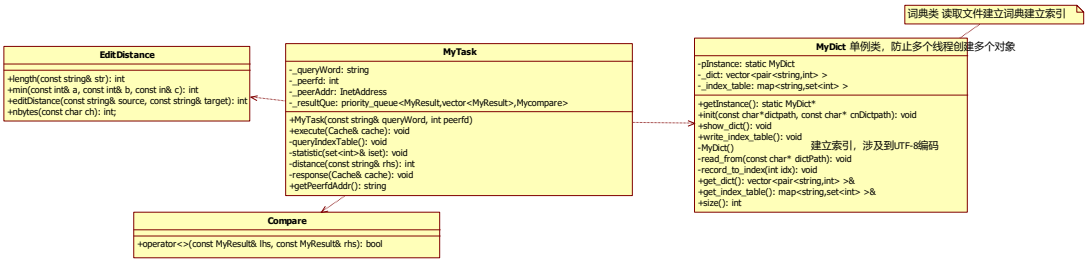
—

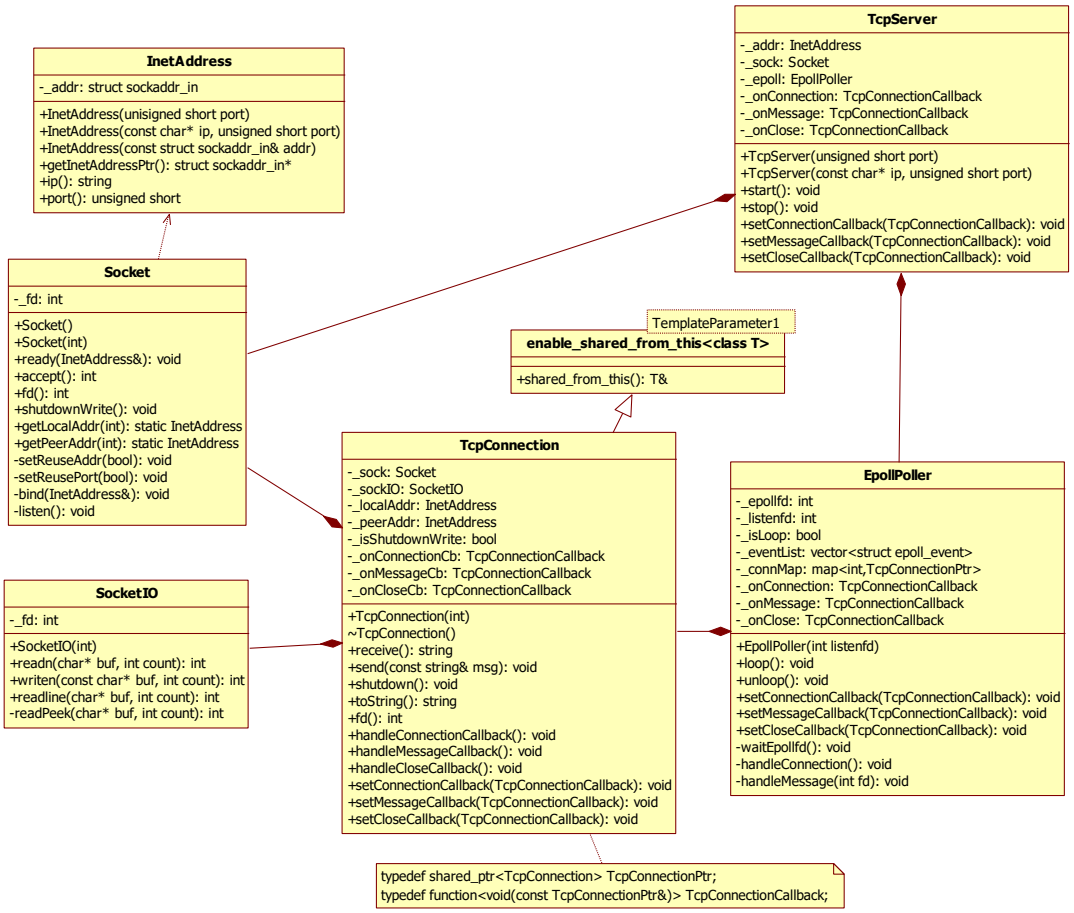

processDict
-_path: string
-_fileName: vector<string>
-_statisticMap: map<string,int>
-_preFile: string
-_dict: string
+processDict(const string& path)
-handleFile(): void
-statisticWord(): void
-writeAnswer(): void
+showFile(): void
+process(): void

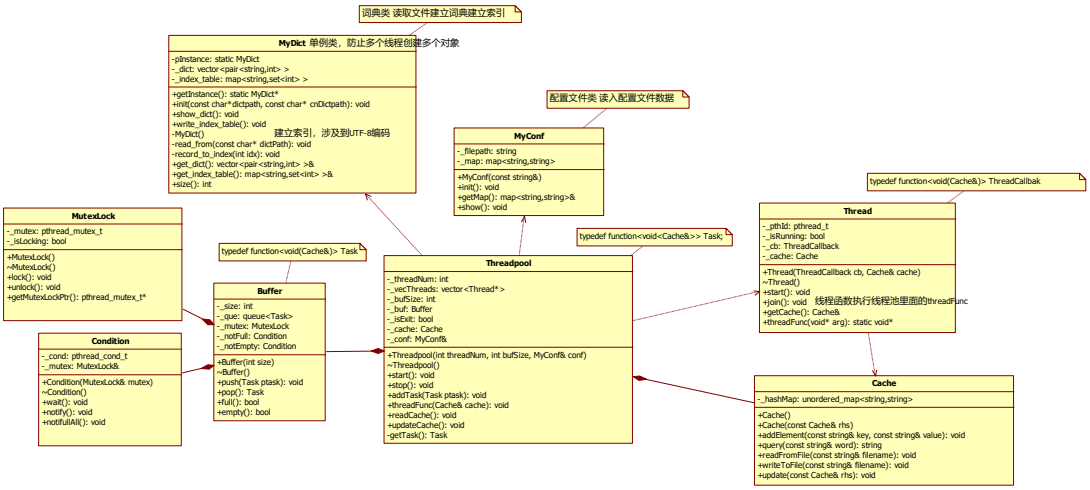

cnDict
-_path: string
-_statisticMap: map<string,int>
-_preFile: string
-_dict: string
+cnDict(const string& path)
+process(): void
-jiebaCut(): void
-statisticWord(): void
-writeAnswer(): void

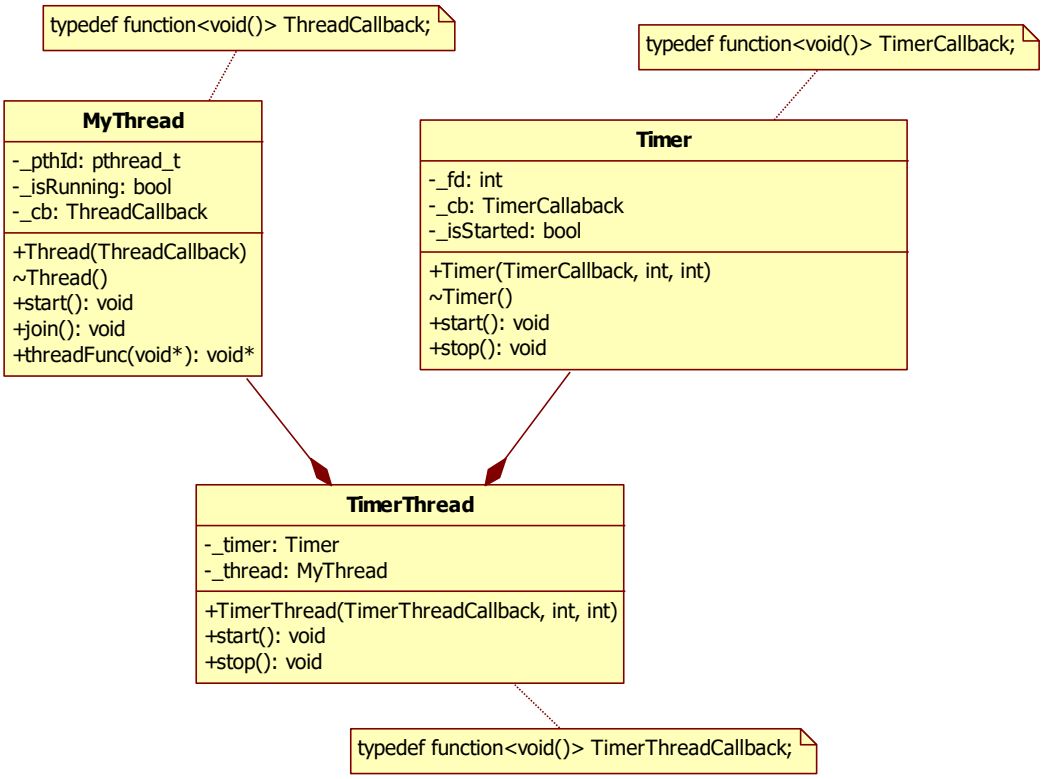

Index
-_readDict: vector<pair<string,int>>
-_index: map<char,set<int>>
-_path: string
+Index(const string& path)
+readDict(): void
+indexMap(): void
+indexMapSave(): void


MutexLock
Condition
Buffer
Threadpool
Thread
Cache
MyConf
spellingAssociate
TimerThread
MyThread
Timer
InetAddress
Socket
TcpServer
MyTask
TcpConnection
EpollPoller
SocketIO
EditDistance
MyDict
Compare


词典类 读取文件建立词典建立索引
EditDistance
+length(const string& str): int
+min(const int& a, const int& b, const in& c): int
+editDistance(const string& source, const string& target): int
+nbytes(const char ch): int;
MyTask
-_queryWord: string
-_peerfd: int
-_peerAddr: InetAddress
-_resultQue: priority_queue<MyResult,vector<MyResult>,Mycompare>
+MyTask(const string& queryWord, int peerfd)
+execute(Cache& cache): void
-queryIndexTable(): void
-statistic(set<int>& iset): void
-distance(const string& rhs): int
-response(Cache& cache): void
+getPeerfdAddr(): string
MyDict 单例类，防止多个线程创建多个对象
-pInstance: static MyDict
-_dict: vector<pair<string,int> >
-_index_table: map<string,set<int> >
+getInstance(): static MyDict*
+init(const char*dictpath, const char* cnDictpath): void
+show_dict(): void
+write_index_table(): void
-MyDict()
建立索引，涉及到UTF-8编码
-read_from(const char* dictPath): void
-record_to_index(int idx): void
+get_dict(): vector<pair<string,int> >&
+get_index_table(): map<string,set<int> >&
+size(): int
Compare
+operator<>(const MyResult& lhs, const MyResult& rhs): bool


InetAddress
-_addr: struct sockaddr_in
+InetAddress(unisigned short port)
+InetAddress(const char* ip, unsigned short port)
+InetAddress(const struct sockaddr_in& addr)
+getInetAddressPtr(): struct sockaddr_in*
+ip(): string
+port(): unsigned short
TcpServer
-_addr: InetAddress
-_sock: Socket
-_epoll: EpollPoller
-_onConnection: TcpConnectionCallback
-_onMessage: TcpConnectionCallback
-_onClose: TcpConnectionCallback
+TcpServer(unsigned short port)
+TcpServer(const char* ip, unsigned short port)
+start(): void
+stop(): void
+setConnectionCallback(TcpConnectionCallback): void
+setMessageCallback(TcpConnectionCallback): void
+setCloseCallback(TcpConnectionCallback): void
Socket
-_fd: int
+Socket()
+Socket(int)
+ready(InetAddress&): void
+accept(): int
+fd(): int
+shutdownWrite(): void
+getLocalAddr(int): static InetAddress
+getPeerAddr(int): static InetAddress
-setReuseAddr(bool): void
-setReusePort(bool): void
-bind(InetAddress&): void
-listen(): void
TemplateParameter1
enable_shared_from_this<class T>
+shared_from_this(): T&
TcpConnection
-_sock: Socket
-_sockIO: SocketIO
-_localAddr: InetAddress
-_peerAddr: InetAddress
-_isShutdownWrite: bool
-_onConnectionCb: TcpConnectionCallback
-_onMessageCb: TcpConnectionCallback
-_onCloseCb: TcpConnectionCallback
+TcpConnection(int)
~TcpConnection()
+receive(): string
+send(const string& msg): void
+shutdown(): void
+toString(): string
+fd(): int
+handleConnectionCallback(): void
+handleMessageCallback(): void
+handleCloseCallback(): void
+setConnectionCallback(TcpConnectionCallback): void
+setMessageCallback(TcpConnectionCallback): void
+setCloseCallback(TcpConnectionCallback): void
EpollPoller
-_epollfd: int
-_listenfd: int
-_isLoop: bool
-_eventList: vector<struct epoll_event>
-_connMap: map<int,TcpConnectionPtr>
-_onConnection: TcpConnectionCallback
-_onMessage: TcpConnectionCallback
-_onClose: TcpConnectionCallback
+EpollPoller(int listenfd)
+loop(): void
+unloop(): void
+setConnectionCallback(TcpConnectionCallback): void
+setMessageCallback(TcpConnectionCallback): void
+setCloseCallback(TcpConnectionCallback): void
-waitEpollfd(): void
-handleConnection(): void
-handleMessage(int fd): void
SocketIO
-_fd: int
+SocketIO(int)
+readn(char* buf, int count): int
+writen(const char* buf, int count): int
+readline(char* buf, int count): int
-readPeek(char* buf, int count): int
typedef shared_ptr<TcpConnection> TcpConnectionPtr;
typedef function<void(const TcpConnectionPtr&)> TcpConnectionCallback;


词典类 读取文件建立词典建立索引
MyDict 单例类，防止多个线程创建多个对象
-pInstance: static MyDict
-_dict: vector<pair<string,int> >
-_index_table: map<string,set<int> >
+getInstance(): static MyDict*
+init(const char*dictpath, const char* cnDictpath): void
+show_dict(): void
+write_index_table(): void
-MyDict()
建立索引，涉及到UTF-8编码
-read_from(const char* dictPath): void
-record_to_index(int idx): void
+get_dict(): vector<pair<string,int> >&
+get_index_table(): map<string,set<int> >&
+size(): int
配置文件类 读入配置文件数据
MyConf
-_filepath: string
-_map: map<string,string>
+MyConf(const string&)
+init(): void
+getMap(): map<string,string>&
+show(): void
typedef function<void(Cache&)> ThreadCallbak
MutexLock
-_mutex: pthread_mutex_t
-_isLocking: bool
+MutexLock()
~MutexLock()
+lock(): void
+unlock(): void
+getMutexLockPtr(): pthread_mutex_t*
typedef function<void(Cache&)> Task
typedef function<void<Cache&>> Task;
Thread
-_pthId: pthread_t
-_isRunning: bool
-_cb: ThreadCallback
-_cache: Cache
+Thread(ThreadCallback cb, Cache& cache)
~Thread()
+start(): void
+join(): void 线程函数执行线程池里面的threadFunc
+getCache(): Cache&
+threadFunc(void* arg): static void*
Buffer
-_size: int
-_que: queue<Task>
-_mutex: MutexLock
-_notFull: Condition
-_notEmpty: Condition
+Buffer(int size)
~Buffer()
+push(Task ptask): void
+pop(): Task
+full(): bool
+empty(): bool
Threadpool
-_threadNum: int
-_vecThreads: vector<Thread*>
-_bufSize: int
-_buf: Buffer
-_isExit: bool
-_cache: Cache
-_conf: MyConf&
+Threadpool(int threadNum, int bufSize, MyConf& conf)
~Threadpool()
+start(): void
+stop(): void
+addTask(Task ptask): void
+threadFunc(Cache& cache): void
+readCache(): void
+updateCache(): void
-getTask(): Task
Condition
-_cond: pthread_cond_t
-_mutex: MutexLock&
+Condition(MutexLock& mutex)
~Condition()
+wait(): void
+notify(): void
+notfuliAll(): void
Cache
-_hashMap: unordered_map<string,string>
+Cache()
+Cache(const Cache& rhs)
+addElement(const string& key, const string& value): void
+query(const string& word): string
+readFromFile(const string& filename): void
+writeToFile(const string& filename): void
+update(const Cache& rhs): void


typedef function<void()> ThreadCallback;
typedef function<void()> TimerCallback;
MyThread
-_pthId: pthread_t
-_isRunning: bool
-_cb: ThreadCallback
+Thread(ThreadCallback)
~Thread()
+start(): void
+join(): void
+threadFunc(void*): void*
Timer
-_fd: int
-_cb: TimerCallaback
-_isStarted: bool
+Timer(TimerCallback, int, int)
~Timer()
+start(): void
+stop(): void
TimerThread
-_timer: Timer
-_thread: MyThread
+TimerThread(TimerThreadCallback, int, int)
+start(): void
+stop(): void
typedef function<void()> TimerThreadCallback;