中期统一风控平台

《概要设计说明书（软件）》

|  |  |  |
| --- | --- | --- |
| 文件状态：   草稿   正式发布   正在修改 | 文件标识： |  |
| 当前版本： | V1.01 |
| 作 者： | 蔡国森 |
| 完成日期： | 2012-11-09 |
| 审 定： |  |

**目 录**

[1 引言 3](#_Toc340819285)

[1.1 编写目的 3](#_Toc340819286)

[1.2 背景 3](#_Toc340819287)

[1.3 定义 3](#_Toc340819288)

[1.4 参考资料 3](#_Toc340819289)

[2 任务概述 4](#_Toc340819290)

[2.1 目标 4](#_Toc340819291)

[2.2 需求概述 4](#_Toc340819292)

[3 总体设计 5](#_Toc340819293)

[3.1 风控客户端UI界面 5](#_Toc340819294)

[3.2 总体结构和模块划分 9](#_Toc340819295)

[3.3 数据管理 10](#_Toc340819296)

[4 模块划分 10](#_Toc340819297)

[5 模块详细设计 11](#_Toc340819298)

[5 尚未解决的问题 21](#_Toc340819299)

# 引言

## 编写目的

根据中期统一风控平台需求分析设计出支持风控平台的客户终端。

## 背景

[待开发软件名称] ：中期统一风控平台客户端

[任务提出者或项目名称] ：

[设计者] ：

[开发者] ：

[目标用户] ：

[版本] ：v1.01

## 定义

* 交易客户端(客户端)：指现有FastTrader为基础的多账号版本
* 中期统一风控平台管理服务器（风控服务器）
* 中期统一风控平台管理客户端（风控客户端）
* CTP服务器（服务器）：综合交易平台的交易和行情服务器

## 参考资料

a.中期统一风控平台需求分析

# 任务概述

## 目标

开发出中期统一风控平台

运行环境

软件：由C++，MFC编写

硬件：装载微软windows XP SP3的台式机

## 需求概述

2.2.1总体需求：

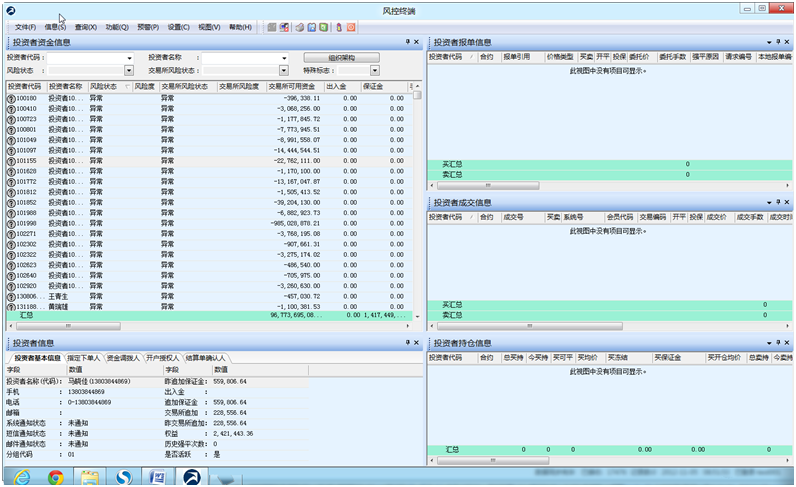
一个开放、分布、高可靠性、高效率、高兼容性的平台。总体功能可以概括为对资管单位的帐户、人员、理财产品、资金情况、持仓情况、日内实时交易监视和报单审核等。

2.2.2 客户端需求：

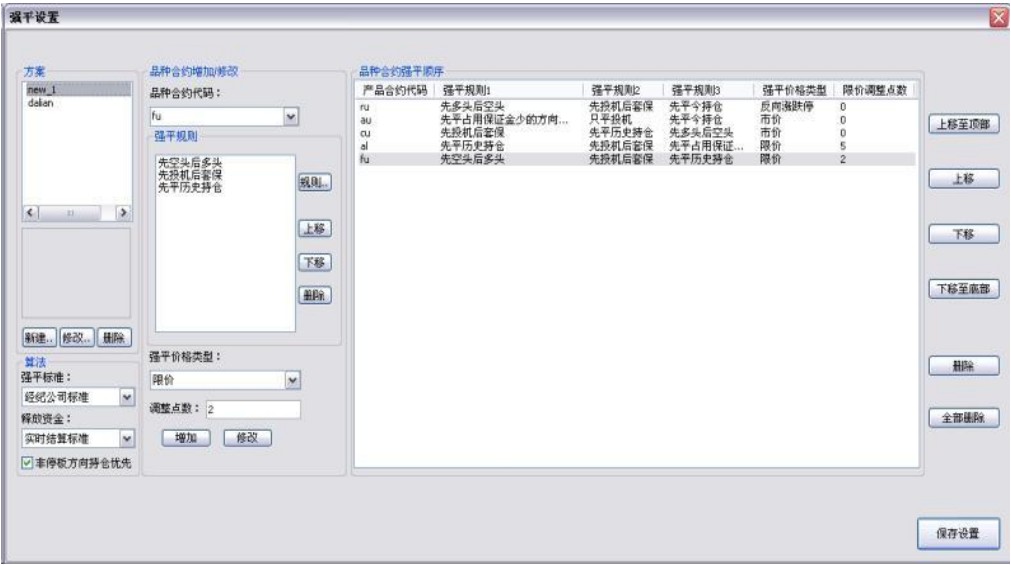
目前需要实现：包含资金查询、投资者信息、投资者报单信息、投资者持仓信息、投资者成交信息、风控指标设定、强平控制、预警控制、消息通知、行情显示、事件记录、系统一般功能。

# 总体设计

## 风控客户端UI界面



主界面

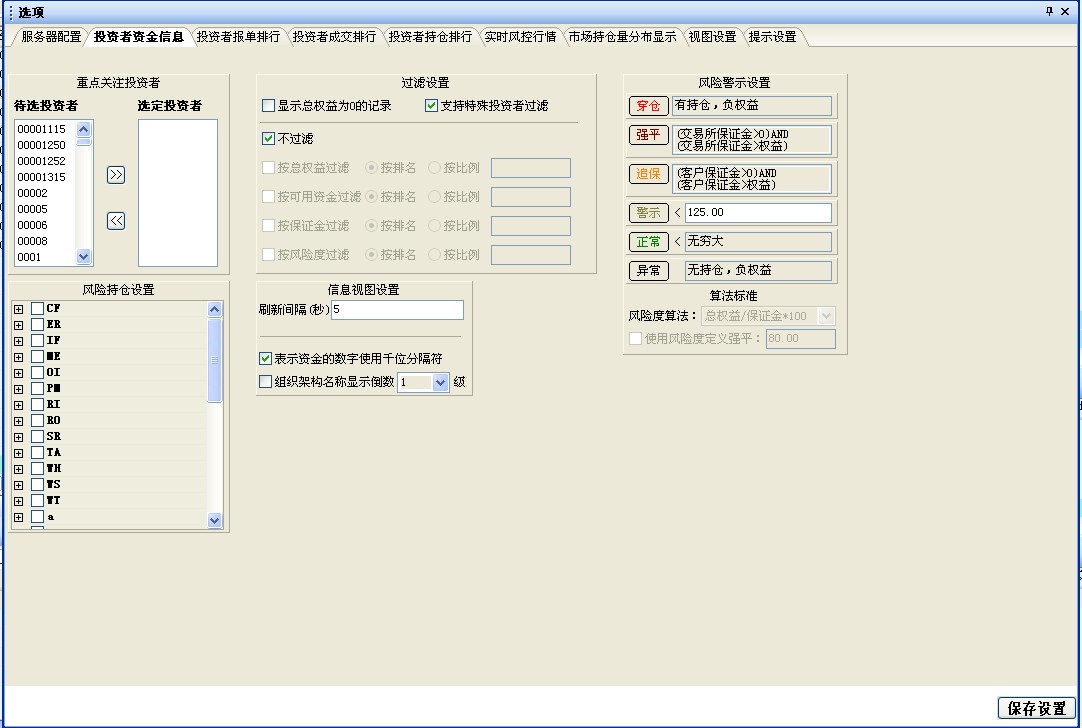
 强平设置窗口



快捷平仓窗口



手动强平窗口



选项

## 总体结构和模块划分

1. 总体结构图



1. 模块划分
2. 网络通讯层：对socket进行一次封装，完成客户端与服务端交互功能。
3. 数据处理模块：界面显示的主要数据结构都由在此模块显示
4. 数据接收线程：负责接收服务端推送消息，转发到主界面
5. 数据发送线程: 用户操作后，产生的服务端交互消息，由此线程负责按序发送
6. 主界面模块（采用多文档程序）

* 持有资金查询、投资者信息、投资者报单信息、投资者持仓信息、投资者成交信息等子界面。
* 持有数据处理模块对象
* 接收处理来自数据接收线程发送的消息

1. 事件记录模块：负责记录风险事件处理过程及结果，写日志文件。
2. 选项配置模块：服务器配置，投资者资金信息等
3. 强平设置，以及快捷强平窗口，手动强平窗口
4. 类设计示意图



## 数据管理

1. 软件运行后，从本地文件读取组织机构的设定，向服务端请求组织机构下的监控账户信息
2. 点击某个账户信息，订阅该账户的投资者信息、投资者报单信息、投资者持仓信息、投资者成交信息。
3. CDataInfo的数据组织方式

struct SInvestorCapitalInfo

{

String strIvestorID;

String strInvestorName;

….

}

Vector<struct SInvestorCapitalInfo> m\_vecICInfo;//投资者资金信息数组

Vector<struct SDealInfo> m\_vecDealInfo; //投资者成交信息数组

Vector<struct SHoldInfo> m\_vecHoldInfo; //投资者持仓信息数组

Vector<struct SAskInfo> m\_vecAskInfo; //投资者报单信息数组

SInvestorInfo m\_vecInvestorInfo； //投资者信息

Vector<struct SForceDeal> m\_vecForceDeal; //强平设置方案数组

Map<string, Vector<struct SForceDeal>&> m\_mapName2VecForceDeal;//具体某个强平方案名字对应的强平方案数组

# 模块划分

|  |  |
| --- | --- |
| **模块名称** | **具体功能** |
| 主框架 |  |
| 通信接口 |  |
| 数据管理模块 |  |
| 参数管理 |  |
| 监控对象选择模块 |  |
| 实时委托列表模块 |  |
| 实时持仓列表模块 |  |
| 实时成交列表模块 |  |
| 实时资金列表模块 |  |
| 实时风险事件列表 |  |
| 风险事件处理列表 |  |
| 地图展现模块 |  |
| 出入金查询 |  |
| 投资者信息查询 |  |
| 消息通知查询 |  |
| 风险事件查询 |  |
| 交易数据查询 | 包括成交、持仓、委托单、资金的查询 |
| 强平设置 |  |
| 快捷强平功能 |  |
| 单客户强平功能 |  |
| 报表输出功能 |  |
| 风险事件处理结果记录 |  |

# 协议

发送函数原型：bool Send(int nCmdId, void\* pData, int nDataLen)

接收函数原型：bool Recv(int nCmdId, void\* pData, int nDataLen)

* 订阅风险事件

nCmdId= CMDID\_RISK\_MANAGER\_QUERY

nDataLen=sizeof(int)

pData为1×int数据指针，存储组织架构ID

返回订阅风险事件

nCmdId= CMDID\_RISK\_MANAGER\_QUERY\_RSP

nDataLen=sizeof(SRiskEvent)

pData为1×SRiskEvent数据指针

* 订阅风险事件处理列表

nCmdId= CMDID\_RISK\_EVENT\_PROCESS\_QUERY

nDataLen=sizeof(int)

pData为1×int数据指针，存储组织架构ID

返回订阅的风险事件处理列表

nCmdId= CMDID\_RISK\_MANAGER\_QUERY\_RSP

nDataLen=sizeof(SRiskEventProcess)

pData为1×SRiskEventProcess数据指针

* 出入金查询

nCmdId= CMDID\_INOUT\_MONEY\_QUERY

nDataLen=sizeof(SInOutMoneyCondition)

pData为1×SInOutMoneyCondition数据指针

返回出入金查询

nCmdId= CMDID\_INOUT\_MONEY\_RSP

nDataLen=sizeof(SInOutMoney)

pData为1×SInOutMoney数据指针

* 投资者信息

nCmdId= CMDID\_INVESTOR\_INFO\_QUERY

nDataLen=sizeof(int)

pData为1×int数据指针，存储投资者ID

返回投资者信息

nCmdId= CMDID\_INVESTOR\_INFO\_QUERY\_RSP

nDataLen=sizeof(SInvestorInfo)

pData为1×SInvestorInfo数据指针

* 消息通知查询

nCmdId= CMDID\_MESSAGE\_QUERY

nDataLen=sizeof(SMessageCondition)

pData为1×SMessageCondition数据指针

返回消息通知查询结果

nCmdId= CMDID\_MESSAGE\_QUERY\_RSP

nDataLen=sizeof(SMessage)

pData为1×SMessage数据指针

* 投资者成交信息

nCmdId= CMDID\_INVESTOR\_TRADE\_QUERY

nDataLen=sizeof(int)

pData为1×int数据指针，存储投资者ID

返回投资者成交信息

nCmdId= CMDID\_INVESTOR\_TRADE\_QUERY\_RSP

nDataLen=sizeof(CShfeFtdcSequencialTradeField)

pData为1×CShfeFtdcSequencialTradeField数据指针

* 投资者资金信息

nCmdId= CMDID\_INVESTOR\_CAPITAL\_QUERY

nDataLen=sizeof(int)

pData为1×int数据指针，存储投资者ID

返回投资者资金信息

nCmdId= CMDID\_INVESTOR\_CAPITAL\_QUERY\_RSP

nDataLen=sizeof(CShfeFtdcSeqPreRiskAccountField)

pData为1×CShfeFtdcSeqPreRiskAccountField数据指针

* 投资者持仓信息

nCmdId= CMDID\_INVESTOR\_POSITION\_QUERY

nDataLen=sizeof(int)

pData为1×int数据指针，存储投资者ID

返回投资者持仓信息

nCmdId= CMDID\_INVESTOR\_ POSITION\_QUERY\_RSP

nDataLen=sizeof(CShfeFtdcSequencialPositionField)

pData为1×CShfeFtdcSequencialPositionField数据指针

* 投资者成交信息

nCmdId= CMDID\_INVESTOR\_TRADE\_QUERY

nDataLen=sizeof(int)

pData为1×int数据指针，存储投资者ID

返回投资者成交信息

nCmdId= CMDID\_INVESTOR\_TRADE\_QUERY\_RSP

nDataLen=sizeof(CShfeFtdcSequencialTradeField)

pData为1×CShfeFtdcSequencialTradeField数据指针

* 投资者委托信息

nCmdId= CMDID\_INVESTOR\_ORDER\_QUERY

nDataLen=sizeof(int)

pData为1×int数据指针，存储投资者ID

返回投资者委托信息

nCmdId= CMDID\_INVESTOR\_ORDER\_QUERY\_RSP

nDataLen=sizeof(CShfeFtdcSequencialOrderField)

pData为1×CShfeFtdcSequencialOrderField数据指针

* 强平方案(每一种方案，对应好几种合约，返回时候，一条条的返回)

nCmdId= CMDID\_FORCE\_DEAL\_QUERY

nDataLen=sizeof(int)

pData为1×int数据指针，存储投资者ID

返回某一合约的强平规则

nCmdId= CMDID\_FORCE\_DEAL\_QUERY\_RSP

nDataLen=sizeof(SForceDeal)

pData为1×SForceDeal数据指针

* 快捷下单

nCmdId= CMDID\_QUICK\_DEAL\_QUERY

nDataLen=sizeof(QuickDeal)

pData为1×QuickDeal数据指针，存储快捷下单信息

返回快捷下单结果

nCmdId= CMDID\_QUICK\_DEAL\_QUERY\_RSP

nDataLen=0

pData为NULL

返回值：0则下单成功

-1则下单失败

* 生成强平单

nCmdId= CMDID\_GENERATE\_DEAL\_QUERY

nDataLen=sizeof(SGenerateForceDealCondition)

pData为1×SGenerateForceDealCondition数据指针，SGenerateForceDealCondition某具体强平方案

返回快捷下单结果

nCmdId= CMDID\_GENERATE\_DEAL\_QUERY\_RSP

nDataLen=sizeof(SGenerateForceDeal)

pData为1×SGenerateForceDeal数据指针

# 模块详细设计

**5.1投资者资金信息**

功能实现：1》根据组织架构里面选择的组织，动态从服务器端请求投资者资金信息详情。

2》根据投资者代码，投资者名称，风险状态，交易所风险状态，特殊标志过滤列表里面的数据。

**5.1.2数据定义**

引用CDataInfo里面的：

Vector<struct SInvestorCapitalInfo> m\_vecICInfo;//投资者资金信息数组

**5.1.3 客户端发送的请求**

bool Send(int nQueryId, void\* pData, int nDataLen)；

nQueryId = QUERY\_INVESTOR\_CAPITAL\_INFO

pData：为组织架构的ID组合，以int类型分开请求，一次请求一个。

nDataLen：为ID组合的int类型长度

**5.1.4 客户端收到服务端反馈的信息**

nQueryId = QUERY\_INVESTOR\_CAPITAL\_INFO\_RSQ

nDataLen = sizeof（SInvestorCapitalInfo）

**5.2 投资者信息**

引用CDataInfo里面的：

SInvestorInfo m\_vecInvestorInfo； //投资者信息

5.2.1 客户端发送的请求

bool Send(int nQueryId, void\* pData, int nDataLen)；

nQueryId = QUERY\_INVESTOR\_INFO

pData：为投资者代码ID。例如：000079

nDataLen：为ID组合的字符串长度

5.2.2 客户端收到服务端反馈的信息

nQueryId = QUERY\_INVESTOR\_INFO\_RSQ

nDataLen = sizeof（SInvestorInfo）

**5.3 投资者成交信息**

引用CDataInfo里面的：

Vector<struct SDealInfo> m\_vecDealInfo;

5.3.1 客户端发送的请求

bool Send(int nQueryId, void\* pData, int nDataLen)；

nQueryId = QUERY\_DEAL\_INFO

pData：为投资者代码ID。例如：000079

nDataLen：为ID组合的字符串长度

5.3.2 客户端收到服务端反馈的信息

nQueryId = QUERY\_ DEAL\_INFO\_RSQ

nDataLen = sizeof（SDealInfo）

**5.4 投资者报单信息**引用CDataInfo里面的：

Vector<struct SAskInfo> m\_vecAskInfo; //投资者报单信息数组

5.4.1 客户端发送的请求

bool Send(int nQueryId, void\* pData, int nDataLen)；

nQueryId = QUERY\_ASK\_INFO

pData：为投资者代码ID。例如：000079

nDataLen：为ID组合的字符串长度

5.4.1 客户端收到服务端反馈的信息

nQueryId = QUERY\_ASK\_INFO\_RSQ

nDataLen = sizeof（SAskInfo）

**5.5 投资者持仓信息**

引用CDataInfo里面的：

Vector<struct SHoldInfo> m\_vecHoldInfo; //投资者持仓信息数组

5.5.1 客户端发送的请求

bool Send(int nQueryId, void\* pData, int nDataLen)；

nQueryId = QUERY\_HOLD\_INFO

pData：为投资者代码ID。例如：000079

nDataLen：为ID组合的字符串长度

5.5.2 客户端收到服务端反馈的信息

bool recv(int nQueryId, void\* pData, int nDataLen)；

nQueryId = QUERY\_HOLD\_INFO\_RSQ

nDataLen = sizeof（SHoldInfo）

**5.6 强平设置窗口**

引用CDataInfo里面的：

Vector<struct SForceDeal> m\_vecForceDeal; //强平设置方案数组

Map<string, Vector<struct SForceDeal>&> m\_mapName2VecForceDeal;//具体某个强平

**5.7单客户强平**

功能：根据投资者ID，从主界面获取ID的资金信息，当前报单，当前持仓。

并根据服务端强平规则生成强平单。

**5.8 报单录入窗口**

功能：提供快捷强平功能。

**5.9 选项设置**

存取本地配置文件。

**5.10 线程设计**

发送线程：list消息队列

# 5 尚未解决的问题

* 风控中涉及具体哪些数据还不是很清楚，需完善
* 跟服务端通讯协议还需解决
* 列表控件还得支持“汇总”的功能，拖拽自动停靠