|  |  |
| --- | --- |
| 文档编号 | 2014082501 |
| 文档负责 | \*\*\* |
| 项目名称 | 机票查询引擎项目 |
| 文件状态 | 草稿 |
| 文件密级 | 绝密/保密/普通 |

**国内航班查询引擎项目设计说明书**

**技术研发中心**

**2014.08**

文档修订记录

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **版本** | **日期** | **撰写** | **审定** | **说明** |
| 1 | v0.1 | 2014-8-25 | \*\*\* |  |  |
| 2 | v0.2 | 2014-8-29 | \*\*\* |  |  |
| 3 | v0.3 | 2014-09-04 | \*\*\* |  | 根据PRD改动调整 |
| 4 | v1.0 | 2014-09-10 | \*\*\* |  | 通过架构评审，固化版 |
|  |  |  |  |  |  |

**目录**

[1 系统概述 5](#_Toc510986298)

[1.1 编写目的 5](#_Toc510986299)

[1.2 适用范围 5](#_Toc510986300)

[1.3 系统目标 5](#_Toc510986301)

[1.4 术语和缩写 5](#_Toc510986302)

[1.5 参考资料 5](#_Toc510986303)

[2 设计约定 5](#_Toc510986304)

[2.1 关键需求 6](#_Toc510986305)

[2.1.1 功能需求 6](#_Toc510986306)

[2.1.2 非功能需求 6](#_Toc510986307)

[2.2 约束 6](#_Toc510986308)

[3 概要设计 7](#_Toc510986309)

[3.1 设计目标和思路 7](#_Toc510986310)

[3.1.1 设计目标 7](#_Toc510986311)

[3.1.2 设计思路 7](#_Toc510986312)

[3.2 功能设计 8](#_Toc510986313)

[3.2.1 用例视图 8](#_Toc510986314)

[3.2.2 用例说明和用例场景 8](#_Toc510986315)

[3.3 外部依赖视图 10](#_Toc510986316)

[3.4 逻辑架构 11](#_Toc510986317)

[3.5 领域设计 12](#_Toc510986318)

[3.5.1 应用部署图 12](#_Toc510986319)

[3.6 接口设计 13](#_Toc510986320)

[3.6.1 查询航班数据接口 13](#_Toc510986321)

[3.7 数据库设计 14](#_Toc510986322)

[3.8 物理架构 16](#_Toc510986323)

[3.8.1 应用集群 16](#_Toc510986324)

[3.8.2 数据库服务器 16](#_Toc510986325)

[3.8.3 App设计 17](#_Toc510986326)

[3.8.4 域名 17](#_Toc510986327)

[3.8.5 Memcached服务器 17](#_Toc510986328)

[4 非功能性设计 17](#_Toc510986329)

[4.1 性能设计 17](#_Toc510986330)

[4.1.1 性能现状分析 17](#_Toc510986331)

[4.1.2 性能目标 17](#_Toc510986332)

[4.1.3 对应设计 18](#_Toc510986333)

[4.1.4 架构满足的性能目标 18](#_Toc510986334)

[4.2 安全设计 18](#_Toc510986335)

[4.3 可靠性设计 18](#_Toc510986336)

[4.4 扩展性设计 18](#_Toc510986337)

[4.5 应用监控及预警 19](#_Toc510986338)

[4.6 可管理及可维护设计 19](#_Toc510986339)

[4.6.1 应用被使用的密度,频度和高峰值 19](#_Toc510986340)

[4.6.2 应用所支持的业务数据重要性要求 19](#_Toc510986341)

[4.6.3 业务对应用可能发生中断的容忍程度（包括日常维护,意外中断) 20](#_Toc510986342)

# 系统概述

## 编写目的

本文档主要描述机票查询引擎项目的架构设计，作为该项目开发阶段的主要依据

## 适用范围

本文档适用于参与此次机票查询引擎项目的所有设计人员、开发及测试人员等。

## 系统目标

1. 将主平台政策以接口的形式接出；
2. 保证接出政策与供应商在主平台查询时获得政策一致；
3. 配合APP4.0版本将APP目前每日出票约100内增加至1000张。

## 术语和缩写

|  |  |  |
| --- | --- | --- |
| **术语与缩写** | **解释** | **备注** |
| Flt | Flight机票 |  |
| Engine | 引擎 |  |

## 参考资料

产品设计说明书：\*\*\*主平台政策接出接口.doc

# 设计约定

|  |
| --- |
| **规范** |
| 《\*\*\*技术研发中心-常用领域词汇》  《\*\*\*技术研发中心-设计评审过程规范》  《\*\*\*技术研发中心-系统设计文档规范》 |
| **约束** |
| 开发平台语言：VS2010/C#+Enterprise Library5.0 |
| 系统运行平台：Windows2008+.NetFramework4.0+IIS7.5 |
| 数据库系统：SQL Server2008 |
|  |

# 关键需求

### 功能需求

#### 功能清单

|  |  |  |
| --- | --- | --- |
| **功能模块** | **主要功能** | **功能描述** |
| 政策获取 | 航班查询（触发） |  |
|  | 航班查询 | 从黑屏及航班接口中查出符合条件航班 |
|  | 取政策 | 从政策库取出政策 |
|  | 取供应信息 | 出票标识，退票标识，出，退时间 |
|  | 计算舱位价格 | 获取不同舱位的价格 |
|  | 政策限制规则 | 地域限制、最高返点限制 |
|  | 显示查询结果 |  |
| 缓存策略管理 | 城市级别管理 | 城市级别，提供配置调整功能 |
|  | 缓存管理 | 不同城市级别，缓存时间调整管理 |
| APP采购贴点 | 贴点 |  |

### 非功能需求

参考产品需求说明书：

|  |  |
| --- | --- |
| **非功能需求** | **需求描述** |
| 安全性 |  |
| 可靠性 | 支持7\*24小时 |
| 容量 |  |
| 并发 |  |
| 扩展性 |  |

# 约束

# 概要设计

## 设计目标和思路

### 设计目标

1. 将分散到各应用的机票查询功能进行集中管理，对外提供统一的接口；
2. 减少中航信通讯流量费用20%-50%，查询结果正确率平均达到95%以上；
3. 提高查询速度，将原有10-25s，缩少到2-6秒；
4. 各接口方易于接入，平滑迁移。

### 设计思路

1. 国内机票查询引擎采用传统的WCF方式，采用SearchRequest/SearchResponse模式。此服务仅提供内部SOA核心查询功能，不包括OpenApi相关的逻辑，OpenApi需要自已在安全、流量监控方面进行再包装；
2. 将串行获取数据改为并行，获取数据源包括政策、贴点、价格、黑屏舱位库存等；
3. 递增式缓存，缓存时间可配置，设置维度包括热门城市对，时间维度，用户维度；
4. 优化算法和数据结构：对数据源包括政策、价格、贴点、黑屏舱位库存等进行数据结构优化，对合并计算的功能进行算法优化；
5. 其它后继三项优化：预加载热门城市对，采用job方式进行加载，预先缓存热门城市的数据；减少政策表中的运行数据，仅保留有效数据，过期等无效数据迁移到其它录入表；优化政策的数据结构。以上三项影响面比较大，考虑放入第二期。

## 功能设计

### 用例视图



### 用例说明和用例场景

#### **查询航班数据**

##### **用例说明**

##### **用例活动图**



## 外部依赖视图



## 逻辑架构



## 领域设计



### 应用部署图



## 接口设计

### 查询航班数据接口

* 接口名称：航班查询数据接口
* 接口描述：根据接入方的请求参数，合并计算并返回航班查询数据
* 接口提供者信息：国内机票查询引擎
* 接口使用者信息：平台、moblieOpenApi
* 接口方式：WCF
* 接口周期：实时

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **请求消息对象** | | | | | |
| **节点名** | **英文名称** | **中文名称** | **数据类型** | **必填** | **备注** |
| FlightSearchRequest | UserName | 用户名 | String | T | 用户名 |
|  | RequestType | 请求类型 | String | T | 请求类型（OpenApi,Web） |
|  | Scity | 出发城市 | String | T | 出发城市三字码(CSX) |
|  | Ecity | 到达城市 | String | T | 到达城市三字码(CSX) |
|  | RateFlag | 政策标记 | String | F | 获取政策True  不获取政策Flase(暂不提供) |
|  | SDate | 出发日期 | String | T | 出发日期yyyy-MM-dd |
|  | AirLine | 航司二字码 | String | F | 航司二字码(CA) |
|  | FlightNo | 航班号 | String | F | 航班号(MU1234) |
|  | PolicyNum | 舱位返回政策条数 | Int | F | 返回政策条数，默认1个 |
|  | Cabin | 舱位类型 | String | F | 舱位类型 A：所有舱位(默认) B：经济舱 C：头等舱 D：全价舱 E：公务舱 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **响应消息对象** | | | | | |
| **节点名** | **英文名称** | **中文名称** | **数据类型** | **必填** | **备注** |
| FltSearchResponse | Success | 成功 | bool | T | 成功标志 |
|  | ErrMsg | 错误信息 | String | F | 错误描述 |
| Flight | **Flights** | 航班集合 | List<Flight> | T |  |
|  | SCity | 出发城市 | String | T | PEK |
|  | Ecity | 到达城市 | String | T | CSX |
|  | SDate | 出发日期 | String | T | yyyy-MM-dd |
|  | AirLine | 航司二字码 | String | T | MF |
|  | FlightType | 机型 | String | T | 738 |
|  | Stime | 起飞时间 | String | T | 06:00 |
|  | Etime | 到达时间 | String | T | 08:20 |
|  | Stop | 经停 | String | T | 0 |
|  | Tax | 机建燃油 | String | T | 150 |
|  | AirT | 航站楼 | String | T | T1,T2 |
|  | **Cabins** | 舱位集合 | List<Cabin> | T |  |
| Flight》Cabin | C | 舱位代码 | String | T | S |
|  | CN | 舱位描述 | String | T | 经济舱 |
|  | CL | 舱位等级 | String | T | B |
|  | D | 折扣 | String | T | 40 |
|  | P | 票面价 | String | T | 500 |
|  | N | 舱位数量 | String | T | 0-9 A大于9 |
|  | AlterNote | 变更说明 | String | F | 变更说明 |
|  | EndNote | 改签说明 | String | F | 改签说明 |
|  | RefundNote | 退票说明 | String | F | 退票说明 |
|  | **Policys** | 政策集合 | List<Policy> | T | 政策集合 |
| Cabin》Policy | PolicyID | 政策ID | String | T | 政策ID |
|  | ProviderID | 供应商ID | String | T | 供应商ID |
|  | K | 返点 | String | T | 3.5 |
|  | KT | 返点类型 | String | T | 0,1,2 |
|  | \*\*\* | 紧急通道标志 | bool | T | 紧急通道标志 |
|  | NoSeat | 60秒出票，NO位全陪标签 | bool | T | 60秒出票NO位全陪标签 |
|  | AllDay | 24小时出废标签 | bool | T | 24小时出废标签 |
|  | OutWorkTime | 出票时间 | String | T | (08:00-21:00) |
|  | RefundWorkTime | 废退票时间 | String | T | (08:00-21:00) |

## 数据库设计

数据库：【\*\*\*2】

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | CityLevel （城市级别） | | | |
| 字段名 | 中文名 | 类型 | 可空 | 备注 |
| CityLevelID | 编号 | int |  |  |
| SCity | 出发城市 | Varchar |  |  |
| ECity | 到达城市 | Varchar |  |  |
| Type | 缓存类型 | int |  |  |
| IsUse | 是否启用 | int |  | （0启用，1关闭） |
| 索引信息 | | | | |
| 索引类型 | 索引名字 | 索引列 | 唯一索引 | 备注 |
| PRIMARY | PK\_CityLevel | CityLevelID | T |  |
| NONCLUSTERED | IDX\_CityLevel\_Scity\_Ecity | Scity,Ecity | T |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | CityCacheRules（缓存规则） | | | |
| 字段名 | 中文名 | 类型 | 可空 | 备注 |
| CityCacheRulesID | 编号 | int |  |  |
| Type | 缓存类型 | int |  |  |
| FromDay | 航程起始日期 | int |  |  |
| ToDay | 航程结束日期 | int |  |  |
| FromDate | 航程起始日期 | datetime |  |  |
| ToDate | 航程结束日期 | datetime |  |  |
| FromTime | 查询起始时间 | int |  |  |
| ToTime | 查询结束时间 | int |  |  |
| CacheTime | 缓存时间 | int |  | 单位 （分钟） |
| Sort | 排序字段 | int |  |  |
| IsUse | 是否启用 | int |  | （0启用，1关闭） |
| DaysAhead | 提前天数 | int |  |  |
| 索引信息 | | | | |
| 索引类型 | 索引名字 | 索引列 | 唯一索引 | 备注 |
| PRIMARY | PK\_CityCacheRules | CityCacheRulesID | T |  |
| NONCLUSTERED | IDX\_CityCacheRules\_FromDate\_ToDate | FromDate, ToDate |  |  |
| NONCLUSTERED | IDX\_CityCacheRules\_ FromDay\_ToDay | FromDay,ToDay |  |  |
| NONCLUSTERED | IDX\_CityCacheRules\_ FromTime\_ToTime | FromTime, ToTime |  |  |

## 物理架构



**由于没有SOA集群，暂不部署在国内机票平台集群中。**

### 应用集群

|  |  |  |  |
| --- | --- | --- | --- |
| **集群** | **部署应用** | **新建** | **说明** |
| 国内前台集群 | FlightSearchEngineWS | T | 国际机票查询引擎服务 |

### 数据库服务器

|  |  |  |
| --- | --- | --- |
| **数据库** | **新建** | **说明** |
| \*\*\* | F |  |
| \*\*\*2 | F |  |
| \*\*\*Rate | F |  |
| FlightPrice | F |  |

### App设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **应用编号** | **应用名称** | **虚拟目录** | **应用类型** | **新建** | **说明** |
| 010201 | FlightSearchEngineWS | FlightSearchEngineWS | SOA | T |  |

### 域名

|  |  |  |
| --- | --- | --- |
| **域名** | **新建** | **说明** |
| \*.\*.\*.231 | F | SOA与OpenApi网络级别隔离，存在单点问题。  服务器要求：8核 |

### Memcached服务器

|  |  |  |
| --- | --- | --- |
| **Memcached实例** | **容量** | **说明** |
| 114.80.75.18:20003 | 4G |  |
| 114.80.75.25:20004 | 4G |  |

为了简化管理和资源的利用率，可暂时只考虑一个实例。

# 非功能性设计

## 性能设计

### 性能现状分析

* 获取黑屏舱位数据：1-2秒左右
* 获取政策数据：1-2秒在右
* 计算舱位价格、获取特价：1秒左右
* 区域划分、最高返点限制：3-4秒左右
* 接口响应时间：10-25秒
* 中航信通讯流量费用：已缓存10分钟
* 查询结果正确率：已达到90%以上

### 性能目标

* 业务量：5万单/天\*3=15万单/天
* 接口访问量pv：订单量5万\*3\*10=150万pv
* 接口响应时间：一般3s以内，最大7s
* 中航信通讯流量费用：再次减少20%-50%
* 查询结果正确率：达到95%以上

### 对应设计

* 负载均衡：通过硬件的负载均衡器，使请求指向不同WEB Server，分散压力。
* 数据库读写分离：建立逻辑库，应用上实现读写分离。物理上采用主从结构实现数据库读写分离，提高数据库访问效率。
* Memcached缓存：信息缓存到memcache，加快信息的查询速度。
* 数据库并发：

数据读操作采用NOLOCK方式提高并发性能；

数据写操作采用无事务、乐观锁、事务的递进锁定机制。

* 数据库索引优化：常用查询关键字设置组合索引，保证查询时索引的使用顺序，减少数据库的数据扫描时间；
* 热门数据预加载：JOB预先加载热门数据，按时间段和城市，提高服务的首次响应速度；
* 减少运行数据和政策数据优化。

### 架构满足的性能目标

## 安全设计

* 数据库连接字符串采用加密方式统一保存；
* 提供的SOA接口被调用时通过AppID进行隔离；
* 防止SQL注入（开发用代码实现，如参数化sql等技术）。
* 所有敏感业务操作均记录日志

## 可靠性设计

* 数据库双机热备
* 失败补偿，重试机制，如查询舱位
* 国内前台1个15台服务器的集群，以防止SPOF问题

国内前台目前是这样分布的：  
上海 lvs 下面 5台  
苏州 lvs 下面 5台  
北京BGP   5台

## 扩展性设计

* 无状态服务，支持应用横向扩展
* 接口采用Request/Response模式进行设计，以支持接口的参数和返回的可扩展性

## 应用监控及预警

* 查询正确率数指标监控

Metrics记录查询匹配率，用dashboard查看，

* 缓存失效率指标监控

Metrics记录缓存失效率，用dashboard查看

## 可管理及可维护设计

* 发生异常时，系统将记录异常日志详情，便于排错；
* 接入Metrics，用dashboard查看，实现系统的监控。

### 应用被使用的密度,频度和高峰值

#### 预计初始数据量和月增长量

* 初始数据量：
* 日增长量：
* 1年的数据量：

#### 预计数据/日志对磁盘空间的要求

无

#### 预计数据/日志对磁盘I/O的要求

无

#### 是否对ISP及上行带宽有特殊要求

无

### 应用所支持的业务数据重要性要求

#### 是否需要对应用数据进行备份

* 将过期政策数据迁移至录入历史数据表
* 将一年前的政策录入历史数据迁移进行清理

#### 是否需要对应用本身进行备份

无

#### 是否需要特殊的数据保留期限

无

#### 是否需要跨服务器交换数据

无

### 业务对应用可能发生中断的容忍程度（包括日常维护,意外中断)

#### 是否需要特殊的数据备份时间窗口

无

#### 是否需要特殊的数据恢复时间要求

无