政府绩效产品设计

# 工作思路和过程



# 业务分析

## 业务范围



## 业务目标



## 业务流程



### 制定考核方案



### 指标绩效考核



### 年底绩效汇总



## 部门及岗位



## 领域模型



### 指标考核



### 单元年度成绩



# 产品定义

## 愿景

解决政府单位在实施绩效工作中缺少XXX、YYY的问题；

相对于a（类似产品），我们的优势是：

1. 有系统化分解指标的业务能力；
2. ……

## 产品分类

1. 针对不同类型的客户问题归集产品特性，形成不同的产品，形成不同的价格；
2. 对客户问题的分类可以来源于客户绩效业务发展的阶段，不同的阶段有不同的问题；

|  |  |  |
| --- | --- | --- |
| 绩效业务发展的阶段 | 遇到的主要问题 | 需要的产品特性 |
| 初始阶段（半手工） | 指标不体系化、缺少周期性考核指标的设计 |  |
| X阶段 | 却少具体指标的考核办法或没有合适的软件系统处理 |  |
| Y阶段 | 不能够利用既往考核周期的数据、不能够对下一考核周期的考核工作作出量化指导 |  |
| Z阶段 | 却少绩效工作与日常工作的有效衔接 |  |

## 产品特性（全集）

1、内置政府单位中组织绩效、个人绩效两块业务，并能够实现其双向互动；

2、支持政府单位内部多级组织结构（如：省、市、县区）下的绩效业务；

3、通过内置应用和外部API的形式实现了年底绩效成绩的多种来源（指标考核、测评、公众评议、领导评价、重点项目执行情况、加减分等）；

4、实现了考核指标的多级灵活设置；

5、支持对考核指标不同频次的考核（年、半年、季、月等）；

6、支持在不同考核频次类型上的进一步定制（删除指定考核频次、设置频次权重及计分规则等）；

7、内置针对不同考核指标的多种计算模型（直接扣分法、累计计分法、数量递减法等）；

8、实现了考核指标向被考核对象上进行计分时的不同规则（加权平均、XXX）；

9、支持不同角色共建指标模型，并提供灵活的审批流程；

10、支持填报和打分过程中灵活的审批流程；

11、支持实时查看不同对象（方案、被考核对象、被考核对象上的指标、指标上的不同频次）上的考核分数情况；

12、能够根据考核结果为被考评对象生成改进建议；

13、支持在创建考核方案时根据之前考核周期生成的数据来提供参考考核指标、目标值等；

14、能够对比多个考核周期不同方案、不同被考核对象的考核结果；

15、提供外部API的方式实现对一个被考核对象上考核指标计分规则中原始分数的采集；

## 系统用例

### 参与者



### 用例

#### 组织绩效-指标考核



##### 创建年度考核方案



##### 制定共性指标



##### 关联被考核单元



##### 查看考核方案执行情况



# 系统分析

## 概念结构



### 领域层主要概念

#### 指标考核



##### 考核方案状态图



##### 计划考核执行记录状态图



## 用例分析

### 关联被考核单元



### 考核打分



### 查看考核方案执行情况



### 查看被考评单元实时分数



# 系统设计

## 逻辑结构



### 类图

#### 指标考核



#### 控制器



## 用例实现

### 关联被考核单元



### 考核打分



### 查看考核方案执行情况



### 启动方案

## 可变性设计



# 系统实现

## 主要决策

1、采用eclipse\jdk8\maven\springboot构建；

2、将应用划分为platform、domain、application、listener、portal、mobile、rest、test八个工程；

3、包结构以com.rofine.gp开始；

4、view采用springboot推荐的thymeleaf构建；

5、持久化采用springdata；

6、主键生成采用uuid；

7、核心对象关系采用manytoone和onetomany构建（采用lazyload策略）；

8、数据库采用mysql；

9、采用Repository模式，通过在IdEntity中调用平台中的ApplicationContextUtil获取Repo；

10、采用可捕获异常来定义业务异常，并建立异常树来分类业务异常；

11、通过实体关系获取实体将不再通过Repo来获取，只通过实体上的引用获取；

12、当通过定制条件获取实体时使用Repo；

13、除了创建实体外，其他操作尽量在DomainService后通过Repo.fin实例化实体；

14、领域逻辑尽量附着在实体上，其次是DomainService；

15、非领域逻辑不允许被DomainService和Entity依赖；

16、领域事件由DomainService抛出；

17、通过@Async可实现异步监听；

18、领域模型中错误处理代码应为正常代码量的n倍；

19、领域模型需配备n倍场景的单元测试代码；

## 已模拟的场景

### 创建方案

/scheme/create GET/POST

### 创建被考核对象

/scheme/{schemeId}/object/create GET/POST

### 创建指标类型

/scheme/{schemeId}/target/type/create GET/POST

/scheme/ {schemeId}/target/type/{parentId}/target/type/create GET/POST

### 创建指标

/scheme/{schemeId}/target/create GET/POST

### 关联指标和被考核对象

/scheme/{schemeId}/target/{targetId}/2object GET/POST

### 启动方案

/scheme/{schemeId}/start POST

### 填报

/ scheme/{schemeId}/fill GET/POST

### 打分

/ scheme/{schemeId}/evaluate GET/POST

### 查看自己的操作记录

/ scheme/{schemeId}/operated/list GET

### 删除指定考核执行

/ scheme/{schemeId}/execute/{ executeId}/delete GET/POST

### 监控考核状态

/ scheme/{schemeId}/monitor GET

### 关闭方案

/ scheme/{schemeId}/close GET

### 登陆

/login/user/{userId}/org/{orgId}/dept/{deptId}/role/{roleIds} GET/POST

### 生成部门分数

/ unit/score/year/{year}/create POST

### 获取部门分数

/ unit/score/year/{year}/get GET

## 代码风格举例

## 可变性举例

# 目前的问题

## 缺少对业务系统性的理解，尤其是对客户问题的总结；

# 附录

## 问题备忘录

1. 产品范围是组织绩效、个人绩效，还是包含两者的政府绩效？
   1. 组织绩效和个人绩效的共同点和差异点有哪些？二者是否有做到一个产品的可能？业务和商业理由充分？
      1. 个人绩效受领导的管理思路影响较大，我们是否有影响领导管理思路的能力呢？
   2. 最终的产品输出物和内部的产品划分是两件事
2. 重点项目和领导驾驶舱的管理模式与组织绩效有较大差异，启动流程是否也由绩效办启动？
3. 一个政府单位的绩效业务，单位以外的人谁关心？参与到流程中吗？
4. 指标分解的职责是否可以固定？（共性指标由绩效办做，个性指标一部分由绩效办设计大的指标项，由被考核部门细化分解，另一部分完全由被考核部门设计）
5. 填报和打分是否一定配置审批流程是否应该成为一个可变点？
6. 年度部门成绩比较接近，谁不愿看到？
7. 对单位的指标考核是在一个上级单位系统上完成还是跨系统完成呢？
8. 对多组被考核单元的考核工作是通过创建多个考核方案来完成还是在一个考核方案下建立多个被考核单元分组来完成考核工作呢？