**千缘MES系统解决方案**

**上海子杰软件有限公司**

**2018-6-11**

**目录**

[一、 子杰软件开发平台功能简介 1](#_Toc516533316)

[1 支持单点登录（SSO） 1](#_Toc516533317)

[2 基础数据采用数据词典方式配置 1](#_Toc516533318)

[3 支持多级授权体系 2](#_Toc516533319)

[4 信息表单字段灵活配置，数据传输接口及时生成 3](#_Toc516533320)

[5 多表数据自动关联，避免信息孤岛产生 6](#_Toc516533321)

[6 复杂关联数据及时生成缓存信息，提升并行性能 7](#_Toc516533322)

[7 威客信息（修改前信息）自动保存，对程序员透明 7](#_Toc516533323)

[8 工作流在线配置，从此没有难配置流程 8](#_Toc516533324)

[9 前端和后台都可配置规则引擎，复杂功能随时修改，及时生效 9](#_Toc516533325)

[10 强大排程功能，影响因素可灵活配置和选择 9](#_Toc516533326)

[11 多数据库灵活选择，当前可支持Oralce、Mysql、sqlServer等数据库 16](#_Toc516533327)

[12 系统具有强大伸缩扩展性，未尽功能可用SSM框架扩展 16](#_Toc516533328)

[二、 子杰软件开发平台技术方案 17](#_Toc516533329)

[1 开发语言使用Java8 17](#_Toc516533330)

[2 整体框架使用SSM 17](#_Toc516533331)

[3 项目jar包管理使用maven 17](#_Toc516533332)

[4 分布式部署使用Dubbo框架 18](#_Toc516533333)

[5 主流数据使用关系型数据库保存 18](#_Toc516533334)

[6 缓存数据使用Redis 18](#_Toc516533335)

[7 日志使用Log4J 18](#_Toc516533336)

[8 后端规则引擎使用Drools 18](#_Toc516533337)

[9 工作流引擎使用Activiti 19](#_Toc516533338)

[10 定时器使用Quartz Schedule 19](#_Toc516533339)

[11 反向代理使用Nginx1.12 19](#_Toc516533340)

[12 消息队列使用TIBCO RV 20](#_Toc516533341)

[13 模板引擎整合使用freemark 20](#_Toc516533342)

[14 搜索引擎使用Lucene 20](#_Toc516533343)

[三、 技术解决方案 21](#_Toc516533344)

[1 清理了一下，千缘MES包含以下几种功能： 21](#_Toc516533345)

[2 说明： 22](#_Toc516533346)

[四、 技术难点突破 23](#_Toc516533347)

[1 现有的流程正确性检测 23](#_Toc516533348)

[2 排程运算 23](#_Toc516533349)

[3 大量的流程配置 23](#_Toc516533350)

[4 大量规则引擎的定义与解析 23](#_Toc516533351)

[5 大量统计数据的产生 23](#_Toc516533352)

# 

# 子杰软件开发平台功能简介

子杰软件开发平台是以遵循软件工程的相关规范为基础，以CMM相关质量控制方法为核心思想，在软件开发的系统分析、设计、开发、测试等环节提出的一套独有的能够量化的定制软件开发管理工具和开发规范，实现对软件定制开发的全过程控制与管理。子杰开发平台不仅能为企业提供一套快速开发的工具，同时也提供了一套智慧的管控一体化的信息支撑平台。

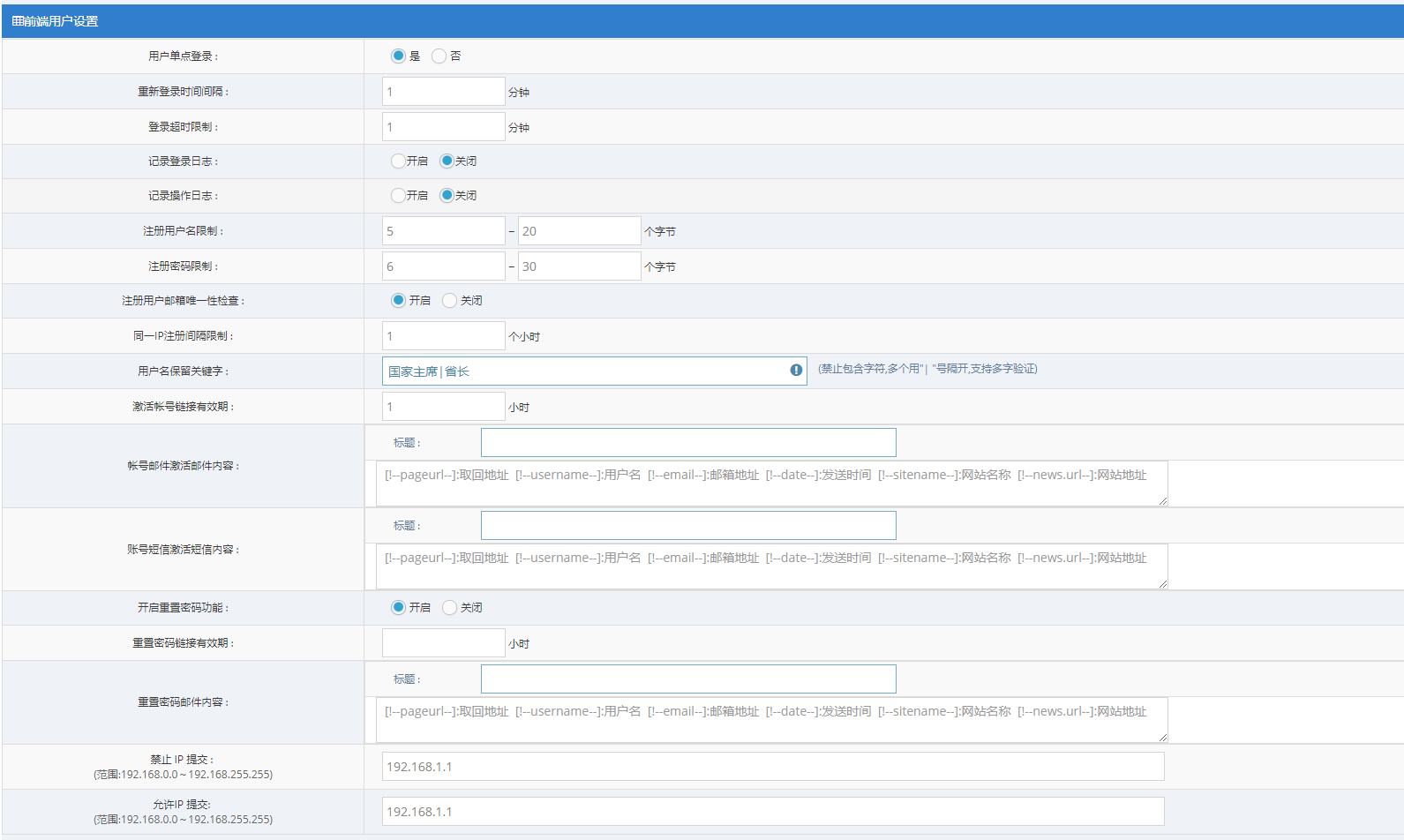
## **支持单点登录（SSO）**

平台可以直接对于前端用户的行为进行配置和记录

### 可直接设置用户是否只能单点登录

### 对于用户登录的信息进行记录

### 设置关键字禁用以及特定IP禁用等



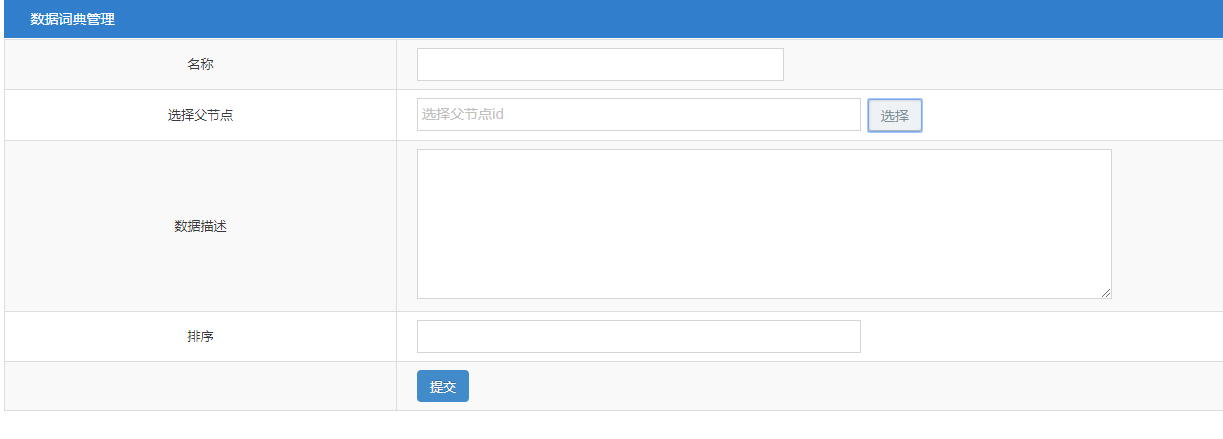
## **基础数据采用数据词典方式配置**

对于不经常进行修改的数据，可通过数据词典进行配置

### 数据词典可根据需要动态添加

### 数据词典各个值之间的父子关系可以在添加时进行选择。

比如：学期可作为上学期、下学期的父级。





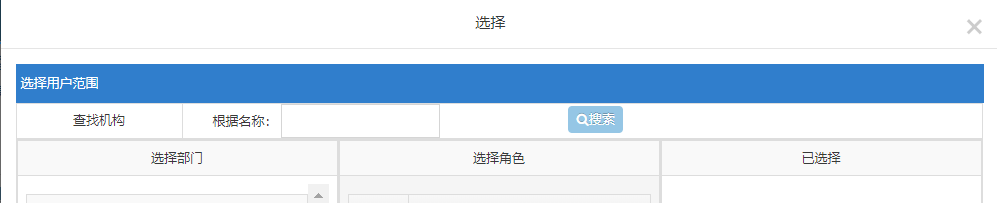
## **支持多级授权体系**

### 可设置不同管理员可操作后台的模块甚至按钮

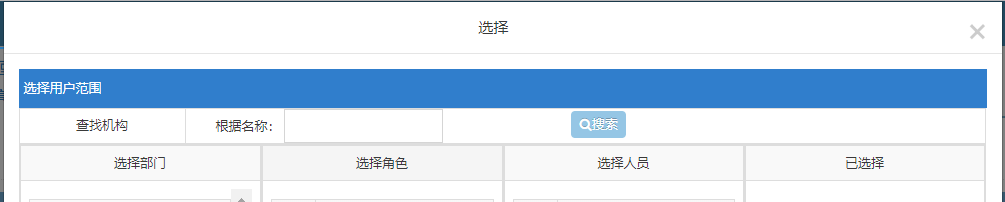


### 针对于不同信息的前端用户的可浏览、可修改权限进行设置；

#### 可直接针对整个部门或者部门角色进行浏览、修改权限设置



#### 可以直接对于某些人进行白名单和黑名单的设置



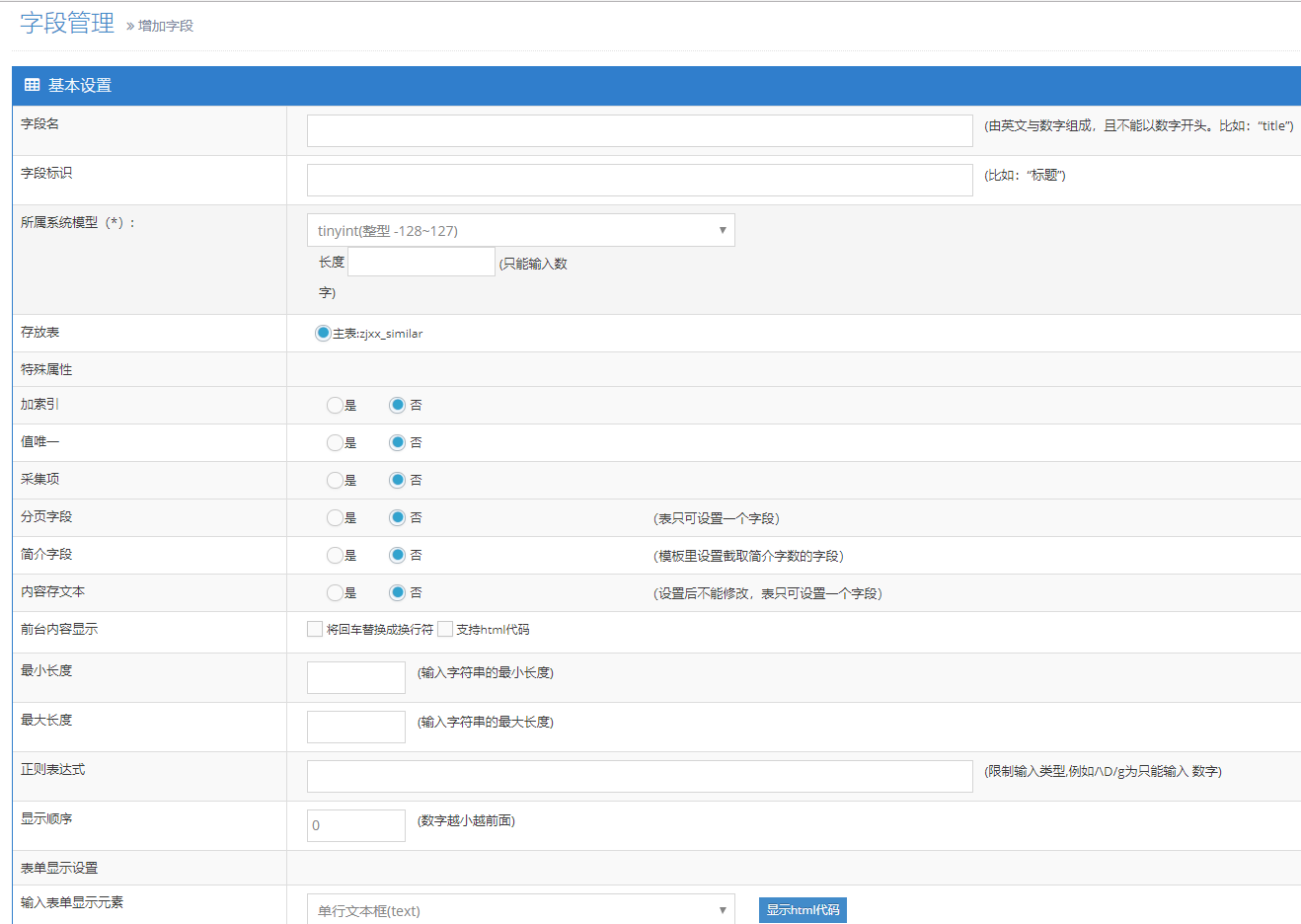
#### 入口如下图



## **信息表单字段灵活配置，数据传输接口及时生成**

可直接设置并生成表的字段，字段可根据需要配置

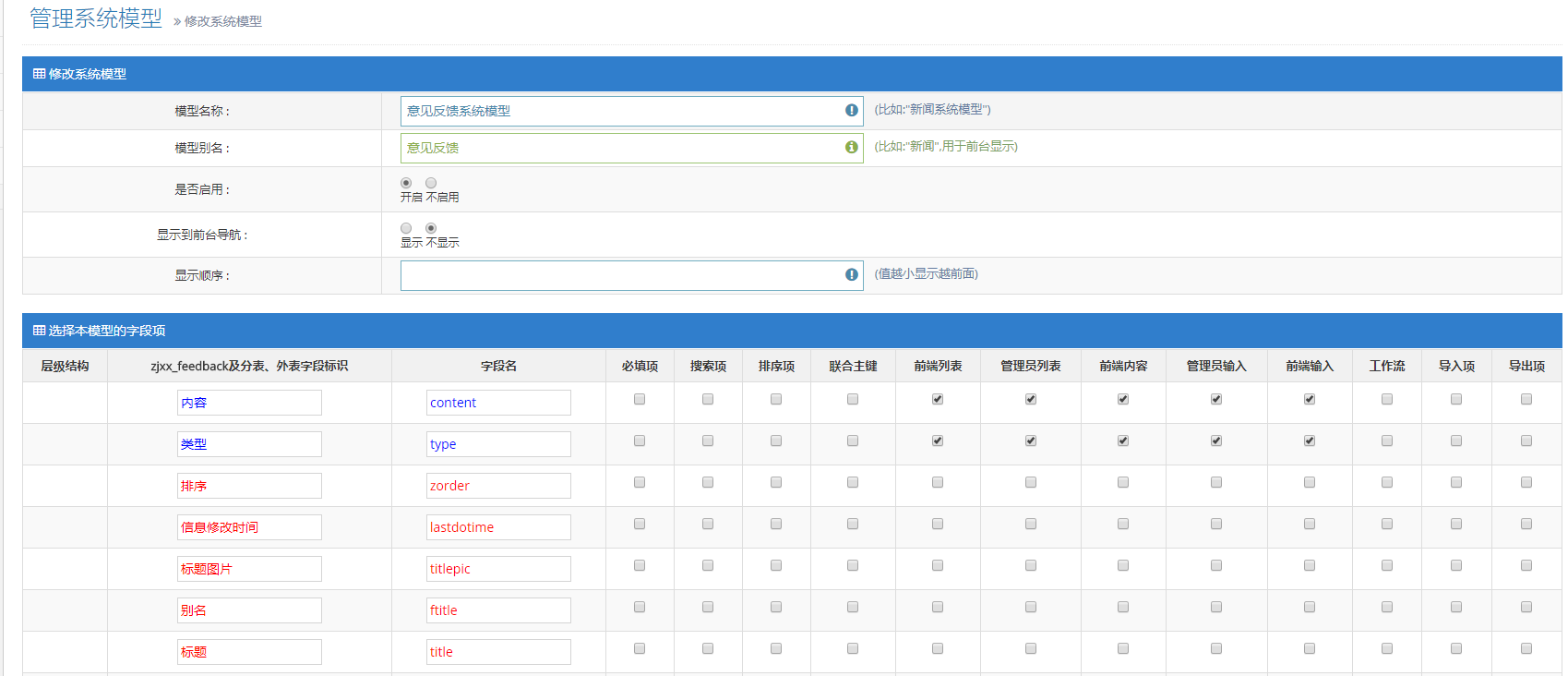
### 对字段的类型、长度等基本属性进行设置。同时可直接设置当前字段再页面上显示的样式（复选框、输入框、富文本等等）



### 通过字段组成数据表，此时，本平台可以通过表建立系统模型，并建立对应的栏目，同时基础的增删改查接口自动生成。

#### 系统模型：一张表可建立多个系统模型，系统模型中选中的字段，可以直接设置为是否前端、后端可查询以及修改等基本权限。

#### 基本配置页如下图：



#### 同时在对应栏目中，需要再次设置当前栏目中信息的查询权限，列表或者内容查询接口中，部分字段是否后台管理员可见或者前台用户可见；

#### 此处还可设置信息接口的查询数量、排序方式。

#### 配置如下图：



## **多表数据自动关联，避免信息孤岛产生**

不同表的数据，可以通过字段进行关联以及查询；比如，A表中，字段a，可以通过B表系统模型对应的栏目id关联到B表，此时，当查询A表中的信息时，可以通过a直接查询出B表中所有的信息。当然，B表中有哪些信息可以被查询、被修改也是可以配置的。

### 第一步，字段关联栏目，可在[1.4.1的模块](#_1.4.1__)中进行设置，如下图



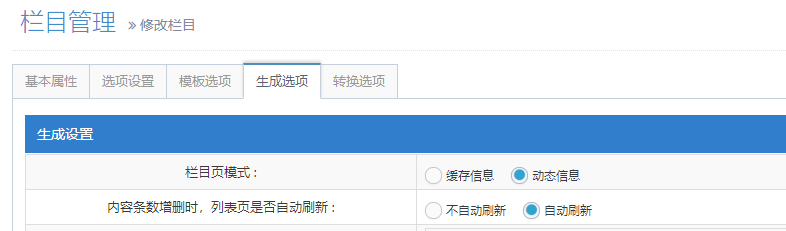
### 第二布，可设置被关联的表的查询信息。方式同[1.4.2](#_1.4..2__通过字段组成数据表，此时，我们可以通过表建立系统模型，)



## **复杂关联数据及时生成缓存信息，提升并行性能**

对于数据量大的信息，本平台可以选择生成缓存信息

### 可以设置当前栏目的信息是否需要缓存以及设置是否自动刷新



### 对于缓存，不仅仅是缓存了信息，还缓存了对应的SQL语句，这样可以更好的提高查询效率。

如果不需要自动刷新的缓存，但是有些时候又需要人工手动刷新，本平台可直接在栏目管理处对于SQL缓存和信息缓存进行手动刷新。



## **威客信息（修改前信息）自动保存，对程序员透明**

各个栏目可以根据需要，选择是否生成威客信息

### 如果选择生成了，那么只要有对当前栏目中信息进行操作，行为都会被记录，随时可以根据需要进行追溯。

### 只有一种情况，必须选择生成威客信息，就是工作流相关的栏目，工作流中的信息是要严格按照规定执行相应操作，本平台必须保证工作流的数据被记录，防止有人误操作导致流程不可追溯和逆转。

设置如图



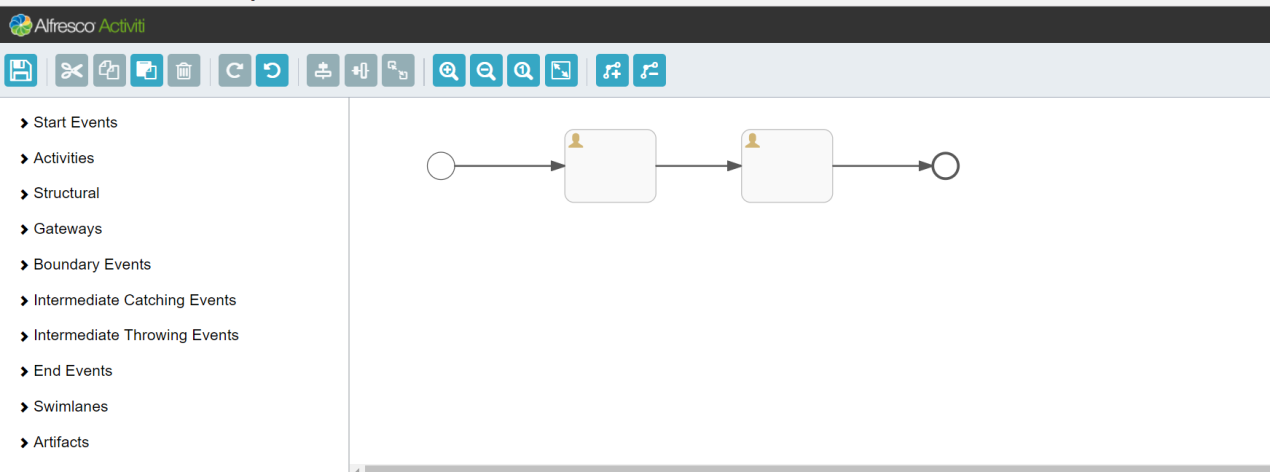
## **工作流在线配置，从此没有难配置流程**

在Java领域，JBPM和Activity是两个主流的工作流系统，本平台选择了结合Activity

### 通过Activity自带的功能，产生最基本的工作流，同时，本平台会根据项目的实际需要进行二次开发。

### Activity主要为本平台提供了基础的流程图配置和节点控制的功能。

### 可通过页面进行流程图的绘制



## **前端和后台都可配置规则引擎，复杂功能随时修改，及时生效**

规则引擎分两大类：前端规则引擎和后端规则引擎。前端规则引擎在前端被解析，逻辑判断部分使用js语言书写。后端规则引擎在后端被java解析，使用Drools语言编写。

每一类规则引擎由三个组成：输入参数定制、规则编写、数据输出规则定义。

## **强大排程功能，影响因素可灵活配置和选择**

排程目的：结合现场需求，系统需满足基于部分参数最大/最小化的排产方案。

例如：将某几日的产线排空，系统优化排产；以“人员”峰值最小化的优化排产等。



一般分两个步骤完成：数据模型化、处理模块化。

### 将排程数据模型化

将数据分解在几个输入表中，输入表大概由以下几个部分组成，示例如下：

#### 订单表（表中数据为假数据）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 订单代码 | 品目代码 | 最早开始时间 | 交货期 | 订单数量 | 优先度 |
| M01 | 0383281 |  | 2004/09/21 09:00 | 9000 |  |
| M02 | 0065205 |  | 2004/09/22 09:00 | 32000 |  |
| M03 | A |  | 2004/09/16 09:00 | 10000 |  |
| M04 | X |  | 2004/09/14 09:00 | 5000 |  |

说明：订单是指工厂里的制造订单。

列的说明：

订单代码 输入订单代码。不要重复，可以不是数字。

品目代码 此订单的产品代码（成品、半成品的物料号）。

最早开始时间 此订单能最早开始生产的时间。可以设定到分秒，可任意设定或不设定。

交货期 此订单的交货期。可以设定到分秒。

订单数量 此订单的数量。

优先度 哪个订单优先的设定。可用于分配顺序的考虑。优先度从0～100，也可以不设定。通常从大的数字开始分派。

根据数据的不同：

如果在90以上，正排的安排（尽可能从最早的工序开始）；

如果在50～90之间，倒排的安排（以交货期为基准从最后的工序开始）；

如果在50以下，可以做顺排的安排。

#### 制造BOM（表中数据为假数据）

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 品目 | 工序编号 | 工序代码 | 指令类别 | 指令代码 | 品种／资源 | 前设置 | 制造 | 后设置 | 接续方法 | 移动时间MIN | 工作批量MAX |
| 0383281 | 10 | 切断 | I | In1 | 软铜材 |  | 0.02 |  |  |  |  |
| 0383281 | 10 | 切断 | U | M | 切断机1 | 10M | 1MP |  |  |  |  |
| 0383281 | 20 | 加工 | U | M | 加工机1 |  | 2MP |  | ES | 10M |  |
| 0383281 | 20 | 加工 | U | M | 加工机2 |  | 2.5MP |  | ES | 10M |  |
| 0383281 | 30 | 热处理 | U | M | 热处理机1 |  | 4h |  | ES | 10M | 20 |
| 0383281 | 40 | 镀锌 | U | M | 外协1 |  | 1D |  | ES | H1D |  |
| 0383281 | 50 | 检查 | U | M | 检验员 |  | 2MP |  | ES | H1D |  |
| 0065205 | 10 | 注塑 | U | M | 注塑1 |  | 1MP |  |  |  |  |
| 0065205 | 10 | 注塑 | U | S1 | 模具备 |  | 0 |  |  |  |  |
| 0065205 | 20 | 去毛刺 | U | M | 去毛刺1 |  | 0.5MP |  | ES | 10M |  |
| 0065205 | 30 | 检查 | U | M | 检验员 |  | 1MP |  | ES | 10M |  |
| 0065205AX | 10 | 组装 | I | In1 | 0065205 |  | 1 |  |  |  |  |
| 0065205AX | 10 | 组装 | I | In2 | 0383281 |  | 1 |  |  |  |  |
| 0065205AX | 10 | 组装 | I | In3 | X |  | 2 |  |  |  |  |
| 0065205AX | 10 | 组装 | U | M | 自动组装机1 |  | 2MP |  |  |  |  |

说明：

主数据是设定各种成品、半成品的工序的流程以及在各工序中制造所需的资源(设备、人员、模具等)的能力值。

列的说明：

品目 成品或半成品的物料号。

工序编号 成品或半成品生产时的工序编号，从开始的工序开始编号逐渐增大。

工序代码 成品或半成品生产时的工序代码。

指令类别 分为：输入指令、使用指令和输出指令等。

使用指令：是指在此工序生产时所使用的机台设备和人力等资源，请输入“U”；

输入指令：是指投入的物料，请输入“I”。

指令代码 请设定为M或S+数字，In+数字.

(M是指生产时所能使用的机台设备；

S是副资源，辅助作用，如工装或人员，如有多个可设定为S1、S2、S3……；

In为输入，如有多个，可设定为In1、In2、In3……)。

品目/资源 在此工序中生产时所使用的资源(设备、机器、人员等)的设定。

前设置 在此资源下生产时所需的前准备时间，以分钟为单位进行设定时可不输入单位。

制造 进行此资源的能力设定(标准时间)，能力值和单位连动着进行设定（输入指令行为投入物料的数量）。

单位如下：

SP 生产1个所需的时间(秒)

MP 生产1个所需的时间(分)

HP 生产1个所需的时间(小时)

PS 1秒所能生产的个数

PM 1分所能生产的个数

PH 1小时所能生产的个数

S 和生产数量无关,必须所花的时间(秒)

M 和生产数量无关,必须所花的时间(分)

H 和生产数量无关,必须所花的时间(小时)

后设置 在此资源生产时所需的后准备时间，以分钟为单位进行设定时不需要输入单位，后设置不常用。

接续方法 进行此工序和前工序之间关系的设定，能进行以下任一种的设定：

ES 前工序结束，后工序才开始生产；

SS 前工序开始，后工序也开始；

SSEE 前后工序同时开始同时结束；

EES 前工序生产任意数量的一部分，后工序就可以开始生产了。

ESE 为了能适应前工序的分割作业的数量和后工序的任意时间相关联

移动时间MIIN 使用接续的方法相关联后的接续时间的最小值的设定，比如工序之间的搬运时间。

如果只输入数字将按分计算，基本的单位如下：

S 秒

M 分

H 小时

作批量MAX 在此资源生产时一个批次数量最大值的设定，比如热处理一次最多只能同时做20个产品。

#### 出勤表（表中数据为假数据）

|  |  |
| --- | --- |
| 出勤模式代码 | 模式 |
| 正常班 | 8:00-12:00;13:00-17:00 |
| 白班 | 8:00-12:00;13:00-17:00;18:00-20:00 |
| 白班B | 8:00-12:00;13:00-17:00;18:00-22:00 |
| 全天 | 0:00-24:00 |
| 休息 |  |

说明：出勤是指记入1天能工作的时间段。

列的说明:

勤模式代码 请记入出勤代码

模式 请记入工作时间段

时间和时间之间用"-"连接，连接的部分为能工作的时间

能工作的时间全部罗列时，用";"连接。

输入例:

正常班的出勤模式，从8:00到12:00是工作时间，12:00到13:00是休息时间，从13:00到17:00是工作时间，

则模式设定为：8:00-12:00;13:00-17:00

休息表示没有工作时间

#### 生产日历

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 资源 | 日期/星期 | 出勤模式代码 | 优先级 | 资源量 | 备注 |
| \* | 星期一-星期五 | 正常班 | 10 | 1 |  |
| 切断机1 | 星期一-星期五 | 白班 | 20 | 1 |  |
| 热处理机1 | 星期一-星期五 | 全天 | 20 | 1 |  |
| 外协1 | 星期一-星期天 | 全天 | 20 | 1 |  |
| 检验员 | 星期一-星期五 | 正常班 | 20 | 5 |  |

说明：生产日历计入资源(设备、人员等)的出勤、休息的设定。

列的说明:

资源 记入资源(设备、人员等)

日期 记入日期或星期

日期处记入星期几或日期(2016/01/01等)

星期几或日期用"-"连接表示范围

星期几或日期用";"连接表示罗列

出勤模式代码 记入出勤表所记载的出勤代码

优先级 当日历有重复时，选用优先级高的日历

资源量 如果使用人员，请记入此出勤时间段的人数

输入例:

第一行 所有的资源（\*代表所有资源），从星期一到星期五按正常班出勤

第二行 "切断机1"这一资源，从星期一到星期五按白班出勤

第三行 "热处理机1"这一资源，从星期一到星期五按全天出勤

第四行 "外协1"这一资源从星期一到星期天按全天出勤

第五行 "检验员"这一资源，从星期一到星期五按正常班出勤，且有5个人

### 数据处理模块化

将会通过调度优化引擎来帮助做决策：每个试制资源需要执行哪些试制任务。

该引擎通过采集试制需求计划之后，引擎将会启动，自动生成最小化试制时间，最佳资源利用率的调度结果来满足需求。

#### 算法描述

算法一般有两种方式，第一在比较有名引擎基础上进行第二次开发，比如采用CPLEX CPO约束规划引擎，第二完全由公司编写算法。两种方式给有优缺点。以第一种方法为例来解释，CPO是约束规划算法库，它可以用来对调度问题建模，CPO的优势是约束传播 (Constraint Propagation)，在调度结果生成的过程中，它非常有效地进行Domain Reduction, 避免不必要的搜索，它能尽可能减少回溯，找出满足约束规则的调度计划。而且，CPO是一个非常成熟的产品，使用它开发和维护模型非常简单有效。CPO默认包括如noOverlap函数，它可以用来表示在同一个时间点，某一个资源上的任务不能重叠等。使用平台的建模师可以把重点放在业务分析和建模上。

算法非常高效而且稳定，是优化算法领域的黄金标杆，能处理百万级以上的决策变量的模型。多次成功地将约束规划技术应用在调度解决方案中。

#### 从数据模型中获取决策变量

决策变量：

各试制工序的计划起始时间和结束时间

各试制工序所使用的试制资源

等等

#### 从数据模型中获取约束条件

约束条件（具体约束条件根据需求调研最终确定）:

计划的起始、终止时间约束

试制作业所包含试制工序的结构性约束

试制工序的时间约束

试制工序对多资源类、多资源的需求约束

试制工序的前后继关系约束

试制工序间的排斥、耦合、保护关系约束

试制资源能力约束

试制资源的工作时间约束

等等

#### 设计模型目标函数

该模型是个多目标模型（实际项目中根据需求可以选择考虑下面一项或者多项）:

试制任务时间最短

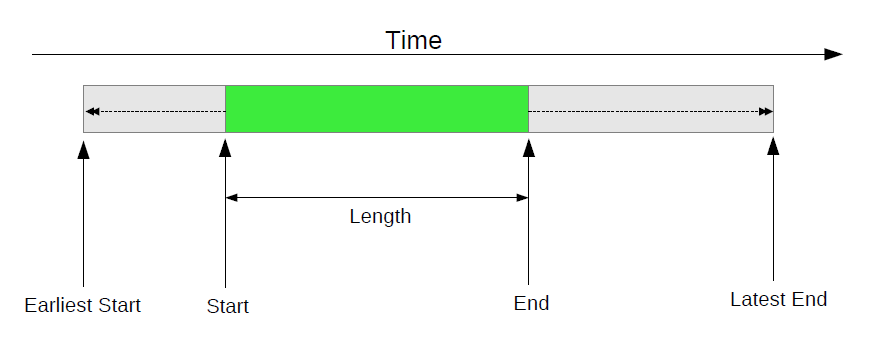
试制资源利用率最高

#### 模型设计并求解

引擎将会管理多个目标，每个目标根据业务需求设置不同的权重进行优化。

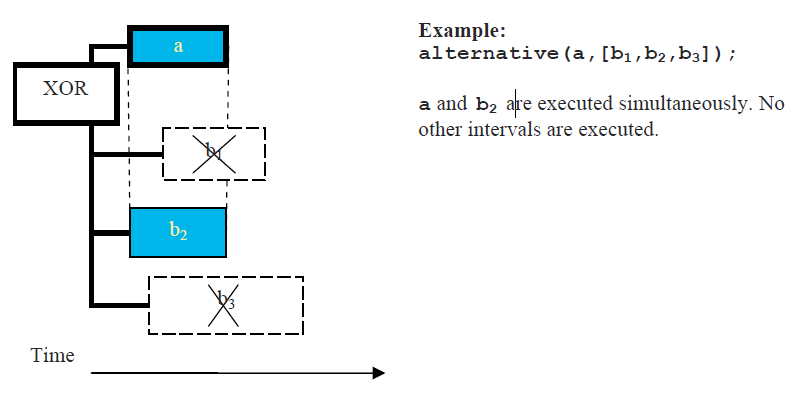
决策变量，约束建模

本调度问题的决策变量如下图所示，每个任务的开始和结束时间都是变量，同时根据实际情况，工作时间的长度也是变量。



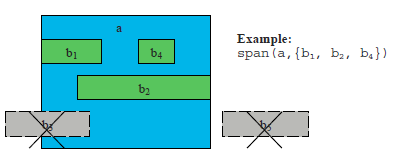
决策变量

具体每个任务选择哪个资源负责，如下图所示通过alternative函数进行约束。



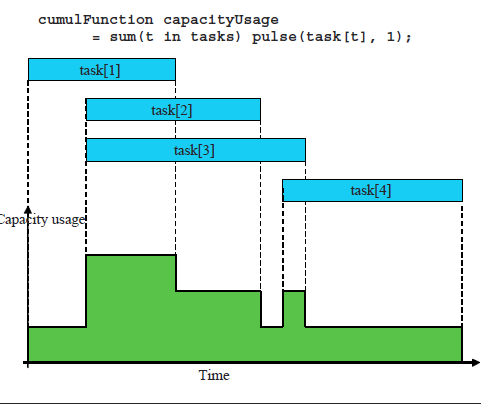
多资源选择约束

整个试制时间的时间区间约束由span函数来进行约束。



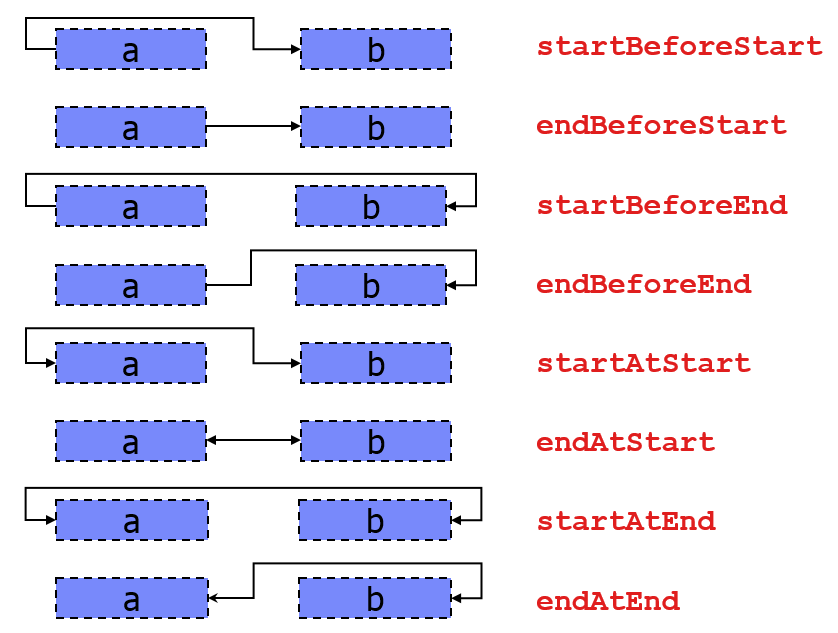
约束

资源的数量由cumulFunction来进行约束。



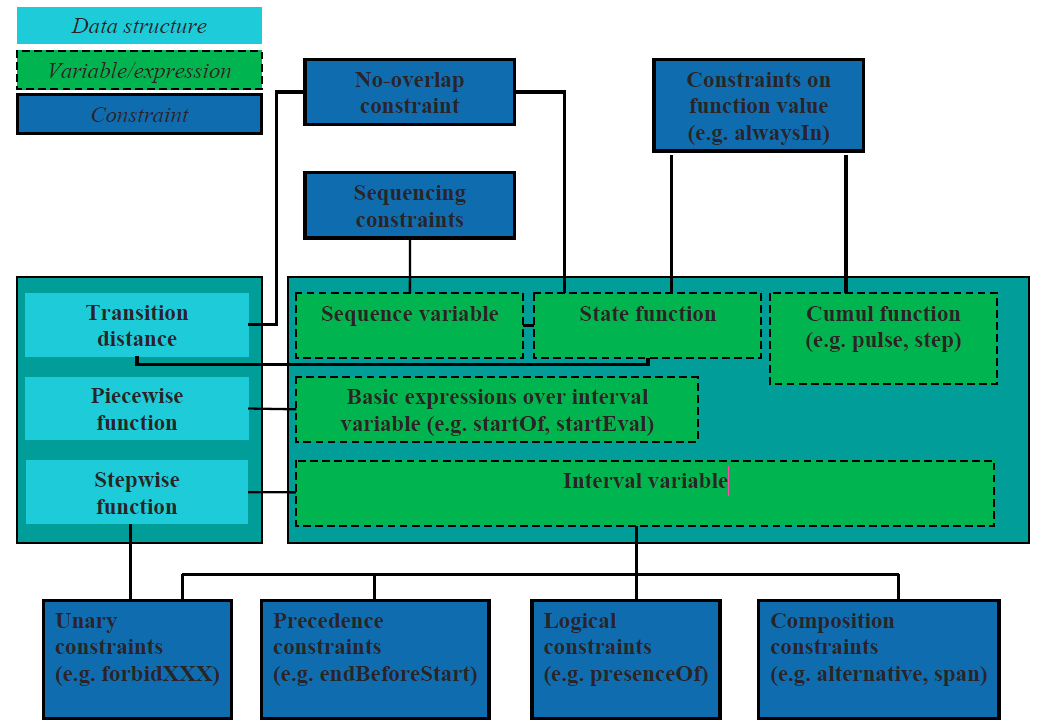
资源数量约束

时间顺序约束由以下函数来进行约束



时序约束

其他约束由以下图中的其他约束函数进行编程满足：



其他约束，如互斥约束、包含约束等

## **多数据库灵活选择，当前可支持Oralce、Mysql、sqlServer等数据库**

当前系统支持Oralce、Mysql、sqlServer三种大型关系型数据库，可以在这三种数据库中任选一种数据库。

## **系统具有强大伸缩扩展性，未尽功能可用SSM框架扩展**

整体技术框架使用SSM，即Spring+Spring MVC+MyBatis。在解决问题过程中，如遇到新的问题，本平台不能直接使用配置方法解决时，可以直接使用SSM框架来编写新的controller来解决。

# 子杰软件开发平台技术方案

## **开发语言使用Java8**

Lambda表达式（也称为闭包）它允许将函数当成参数传递给某个方法，或者把代码本身当作数据处理

Java8使用两个新概念扩展了接口的含义：默认方法和静态方法。

方法引用使得开发者可以直接引用现存的方法、Java类的构造方法或者实例对象。方法引用和Lambda表达式配合使用，使得java类的构造方法看起来紧凑而简洁，没有很多复杂的模板代码。

重复注解，Java8中使用@Repeatable注解定义重复注解

Java8编译器在类型推断方面有很大的提升，在很多场景下编译器可以推导出某个参数的数据类型，从而使得代码更为简洁。

Java8拓宽了注解的应用场景。

Java8增加了很多新的工具类（date/time类），并扩展了现存的工具类，以支持现代的并发编程、函数式编程等。

## **整体框架使用SSM**

SSM框架，是Spring+Spring MVC+MyBatis的缩写，这个是继SSH之后，目前比较主流的JavaEE企业级框架，适用于搭建各种大型的企业级应用系统。

Spring是一个开源框架。Spring是于2003年兴起的一个轻量级的Java开发框架，由Rod Johnson在其著作Expert One-On-One J2EE Development and Design中阐述的部分理念和原型衍生而来，它是为了解决企业应用开发的复杂性而创建的。Spring使用基本的JavaBean来完成以前只可能由EJB完成的事情。然而，Spring的用途不仅限于服务器端的开发。从简单性、可测试性和松耦合的角度而言，任何Java应用都可以从Spring中受益。简单来说，Spring是一个轻量级的控制反转（IoC）和面向切面（AOP）的容器框架。

Spring MVC属于SpringFrameWork的后续产品，已经融合在Spring Web Flow里面。Spring MVC分离了控制器、模型对象、分派器以及处理程序对象的角色，这种分离让它们更容易进行定制。

MyBatis本是apache的一个开源项目iBatis,2010年这个项目由apachesoftware foundation迁移到了google code，并且改名为MyBatis。MyBatis是一个基于Java的持久层框架。iBATIS提供的持久层框架包括SQL Maps和Data Access Objects（DAO）MyBatis消除了几乎所有的JDBC代码和参数的手工设置以及结果集的检索。MyBatis使用简单的XML或注解用于配置和原始映射，将接口和Java的POJOs（Plain Old Java Objects，普通的Java对象）映射成数据库中的记录。

## **项目jar包管理使用maven**

maven是一个使用java编写的开源的项目管理工具。实际上所有的项目都被定义到了pom.xml文件中。通过使用maven工具可以管理项目的生命周期，包括清除、编译、测试、报告、打包、部署等操作。

maven可以方便灵活的控制项目，重要的是方便开发人员不必浪费时间去在不同的环境中配置依赖的jar包，而专心于业务逻辑。

## **分布式部署使用Dubbo框架**

Dubbo是Alibaba开源的分布式服务框架，可以非常容易地通过Dubbo来构建分布式服务，并根据自己实际业务应用场景来选择合适的集群容错模式，这个对于很多应用都是迫切希望的，只需要通过简单的配置就能够实现分布式服务调用，也就是说服务提供方（Provider）发布的服务可以天然就是集群服务，比如，在实时性要求很高的应用场景下，可能希望来自消费方（Consumer）的调用响应时间最短，只需要选择Dubbo的Forking Cluster模式配置，就可以对一个调用请求并行发送到多台对等的提供方（Provider）服务所在的节点上，只选择最快一个返回响应的，然后将调用结果返回给服务消费方（Consumer），显然这种方式是以冗余服务为基础的，需要消耗更多的资源，但是能够满足高实时应用的需求。

## **主流数据使用关系型数据库保存**

目前支持三种数据库：MySql数据库、SQL Server数据库、Oracle数据库

## **缓存数据使用Redis**

redis是Nosql数据库中使用较为广泛的非关系型内存数据库，redis内部是一个key-value存储系统。它支持存储的value类型相对更多，包括string(字符串)、list(链表)、set(集合)、zset(sorted set–有序集合)和hash（哈希类型，类似于Java中的map）。

## **日志使用Log4J**

Log4j是帮助开发人员进行日志输出管理的API类库。它最重要的特点就可以配置文件灵活的设置日志信息的优先级、日志信息的输出目的地以及日志信息的输出格式。Log4j除了可以记录程序运行日志信息外还有一重要的功能就是用来显示调试信息。很多程序员经常会使用System.out.println语句输出某个变量值的方法进行调试。这样会带来一个非常麻烦的问题：一旦哪天程序员决定不要显示这些System.out.println的东西了就只能一行行的把这些语句注释掉。若哪天又需调试变量值，则只能再一行行去掉这些注释恢复System.out.println语句。使用log4j可以很好的处理类似情况：使用"DEBUG"等级输出会类似System.out.println，开发后期不需要输出调试时可以把等级根据需要调高，如调到"ERROR"等级，只有出错错误时才输出。

## **后端规则引擎使用Drools**

Drools（JBoss Rules ）具有一个易于访问企业策略、易于调整以及易于管理的开源业务规则引擎，符合业内标准，速度快、效率高。业务分析师或审核人员可以利用它轻松查看业务规则，从而检验是否已编码的规则执行了所需的业务规则。   
Drools是一个基于java的规则引擎，开源的，可以将复杂多变的规则从硬编码中解放出来，以规则脚本的形式存放在文件中，使得规则的变更不需要修正代码重启机器就可以立即在线上环境生效。

## **工作流引擎使用Activiti**

Activiti其核心是BPMN2.0的流程引擎。BPMN是目前被各BPM厂商广泛接受的BPM标准，全称为Business Process Model and Notation，由OMG组织进行维护，2011年1月份发布了其2.0的正式版。BPMN2.0对比于第一个版本，其最重要的变化在于其定义了流程的元模型和执行语义，即它自己解决了存储、交换和执行的问题。这代表着BPMN2.0流程定义模型不仅仅可以在任何兼容BPMN2.0的引擎中执行，而且也可以在图形编辑器间交换。作为一个标准，BPMN2.0统一了工作流社区。

Activiti是由jBPM的创建者Tom Baeyens离开JBoss之后建立的项目，构建在开发jBPM版本1到4时积累的多年经验的基础之上，旨在创建下一代的BPM解决方案。同时Activiti选择了Apache许可，一方面是希望Activiti能有更长久的生命力，因为它不受任何个人或是公司的控制而是属于整个社区，另一方面更是希望这个宽松的许可能够让Activiti BPM引擎和BPMN2.0被更广泛的采纳、使用和商业化。

对于Java开发者来说，Activiti的首席架构师Tom Baeyens曾提到，Activiti的首个目标就是要获得开发者的青睐。首先它在使用时极为方便，只是个jar文件，使用时仅需要将其放在类路径中，当然，Activiti也可以作为独立服务器的方式使用；同时Activiti提供了很多BPM高级工具，其中还包括开发了协作工具，使得开发人员、业务人员和运维人员能够更好的协同工作

## **定时器使用Quartz Schedule**

Quartz是一个任务调度框架。比如你遇到这样的问题：

想每月25号，信用卡自动还款

想每年4月1日自己给当年暗恋女神发一封匿名贺卡

想每隔1小时，备份一下自己的爱情动作片学习笔记到云盘

这些问题总结起来就是：在某一个有规律的时间点干某件事。并且时间的触发的条件可以非常复杂（比如每月最后一个工作日的17:50），复杂到需要一个专门的框架来干这个事。Quartz就是来干这样的事，你给它一个触发条件的定义，它负责到了时间点，触发相应的Job起来干活。

## **反向代理使用Nginx1.12**

什么是Nginx？Nginx是一个高性能的HTTP和反向代理服务器，也是一个IMAP/POP3/SMTP代理服务器。Nginx是一款轻量级的Web服务器/反向代理服务器以及电子邮件代理服务器，并在一个BSD-like协议下发行。由俄罗斯的程序设计师lgor Sysoev所开发。

什么是反向代理？反向代理是指以代理服务器来接受internet上的连接请求，然后将请求转发给内部网络上的服务器，并将从服务器上得到的结果返回给internet上请求连接的客户端，此时代理服务器对外就表现为一个反向代理服务器。

反向代理使用Nginx1.12的好处：占有内存少，并发能力强。

## **消息队列使用TIBCO RV**

TIBCO Rendezvous（或称为TIBCO RV）产品是一种中间件，它具有发布/订阅(Publish/Subscribe)、基于主题寻址(Subject-Based Addressing)和自定义数据信息(Self-Describing Data Messages)等专利技术功能，使不同应用平台上的信息在一个共享的虚拟总线Information Bus(TIB)上进行传输交换。这些技术能有效地帮助企业从传统的请求/应答(Request/Reply)模式转到自动数据接受的事件驱动模式(Event-Driven，或称之为Push)。TIBCO RV有助于在各种应用系统中获取信息和数据，能将异构平台有机地联结起来,通过以即插即用(Plug &Play)、位置无关（Location-Independent）和分布式服务（Distributed Services）的方式在WAN和LAN间配置系统。并且TIBCO RV具有认证消息传递(Certified Message Delivery)、容错(Fault Tolerance)和分布式队列（Distributed Queue）功能。因为使用TIBCO RV不用考虑网络的技术细节，而只需专注于企业应用的开发，所以能快速建立和配置一个可伸缩的分布式应用系统。

TIBCO Rendezvous的优点：

加快应用的开发，减少维护费用；

唯一独立于硬件、操作系统、网络和协议平台供应商；

动态组件替换:进程可以随时加载、退出、替换，而不影响系统运行；

屏蔽网络细节；

应用伸缩性高；

地址无关，简化增加/改变组件；

提高分布系统的生命期；

## **模板引擎整合使用freemark**

FreeMarker是一个用Java语言编写的模板引擎，它基于模板来生成文本输出。FreeMarker与Web容器无关，即在Web运行时，它并不知道Servlet或HTTP。它不仅可以用作表现层的实现技术，而且还可以用于生成XML，JSP或Java文等。

FreeMarker的优点：1、可以彻底的分离表现层和业务逻辑；2、可以提高开发效率；3、使得开发过程中的人员分工更加明确；

## **搜索引擎使用Lucene**

索引文件格式独立于应用平台。Lucene定义了一套以8位字节为基础的索引文件格式,使得兼容系统或者不同平台的应用能够共享建立的索引文件

在传统全文检索引擎的倒排索引的基础上,实现了分块索引,能够针对新的文件建立小文件索引,提升索引|速度。然后通过与原有索引的合并,达到优化的目的。

优秀的面向对象的系统架构,使得对于Lucene扩展的学习难度降低,方便扩充新功能。

设计了独立于语言和文件格式的文本分析接口,索引器通过接受Token流完成索引文件的创立,用户扩展新的语言和文件格式,只需要实现文本分析的接口。

已经默认实现了一套强大的査询引擎,用户无需自己编写代码即使系统可获得强大的查询能力,Lucene的査询实现中默认实现了布尔操作、模糊查询(Fuzzy Search)、分组查询等等。

# 技术解决方案

## **清理了一下，千缘MES包含以下几种功能：**

### 人力资源管理

### 工时管理

### 财务管理

### 合同管理

### 项目管理

### 加工中心（工艺）管理

### BOM：（Bill of Material）物料清单

### 供应链管理

### 设备管理

### APS排程管理（生产计划、生产排产、生产调度）

### 生产管理（具体产线管理）

### 任务管理

### 外协管理（把供应商都当做部门管理）

### 仓库管理

### 售后管理

### 风险预警

### 异常管理

### 生产绩效：设备利用率、设备负荷分析、人员绩效

### 成本分析

## **说明：**

### “人力资源管理、工时管理、财务管理、合同管理、项目管理、加工中心（工艺）管理、BOM：（Bill of Material）物料清单、供应链管理、设备管理、仓库管理、售后管理”这些功能比较简单，是一般的内容增删改查功能，当然也会涉及到内容分类等功能，在系统中直接配置就可产生对应功能。

### 排程非常复杂，数据模型设计好后也要由专门的人员按照第一大点第10点来解决这部分功能。

### 一旦排程结束后，就会产生生产数据，同时系统进入生产管理流程；如果把外部资源统一管理，特别把外部的公司当成自己内部部门管理，“外协管理”就成了生产管理的一部分。

### 如果生产数据和人员相关，就会产生任务数据，同时系统就会进入任务管理流程。两者遇到问题，就进入风险报警流程，严重的进入异常管理流程。“风险报警流程”和“异常管理流程”由java定时器定期检测预警规则引擎和异常规则引擎产生。

### 等上述数据产生后，可以用系统统计功能统计“设备利用率、设备负荷分析、人员绩效、生产成本”等多种数据供领导参考。这一部分可由数据库存储过程配置来解决。

### 在需求中出现了“数控部”、“钳工部”，实际上这些概念应该分解在人力资源管理和工艺管理两个部分，只是同一工艺流程的不同阶段，就没有专门来讨论。

# 技术难点突破

## **现有的流程正确性检测**

整个项目流程很多，有些流程从现在感觉有些问题，要评审后确定正确性。

## **排程运算**

排程的解决先要理清本平台和排程引擎是两个不同部分，可能部署在同一台电脑上也有可能部署在不同的电脑上。

本平台和排程引擎关系如下：1、平台对数据进行初步整理.3、平台将整理的数据发给排程引擎，排程引擎对平台发过来的数据进行缓存、分类、规整、约束、运算等一系列过程处理，产生排程数据，并将排程数据传给平台。4、平台对排程引擎传过来数据保存在平台中。5、由平台对排程数据进行修改、汇总、显示。

## **大量的流程配置**

在需求阶段整理出所有的流程，利用平台功能配置出所有流程。本功能难点在需求分析，不是流程配置。

## **大量规则引擎的定义与解析**

同理，本平台支持规则引擎配置，难点在需求，不在最后的解决技术。

## **大量统计数据的产生**

一般使用数据库存储过程书写来解决。