BPS统一流程平台改造技术方案

1. **变更说明：**

|  |
| --- |
| 新需求：   1. 为字段和记录增加操作标记的功能，区分是新增、修改还是删除；对于修改数据的流程，要区分哪些记录做过修改，并为修改过的字段提供前后值比较功能；在向业务系统同步的时候对于没有修改过的记录就不同步。 2. 组织机构模型改造及存量数据处理 3. 在每支流程结束的时候，调用业务系统通知结束相关的参数维护任务(收单、跨行、文件) 4. 查询复核历史功能，支持按照业务流程实例查询，并区分是测试复核还是生产复核 5. 为流程的强制终止和任务的强制完成记录审计日志，并提供审计日志查询功能 6. 流程图业务目录划分，并根据业务目录过滤任务查询功能； 7. 为UOSP系统开发卡bin准入的4支流程图 8. 安全漏洞修复 9. 流程主题拼装逻辑的修改 10. 修改记录引擎和外部交互的接口，以前只在失败的时候记报文，修改为成功的时候也记录（bpsconstant.properties 中修改CUPBPS\_LOG\_SUCCESS为1） 11. 自动活动代理类的实现（在EOS\_DICT\_ENTRY表中配置每个系统的服务地址） 12. 为机构准入系统提供启动流程的服务 13. 流程相关功能的代码梳理 14. 将强制生效按钮放到最后（考虑红色标注） 15. 将bpsmonitorApp优化拆分，并对脚本进行改造； 16. 将流程图的发布过程和应用的发布过程分开，使得每次上线流程图的发布操作只进行一次 17. 统一用户管理平台增加了查询用户全部权限的接口 18. 跨行管理、收单管理、统一文件增加了参数任务关闭的接口 |

1. **技术实现方案**

# 为字段增加操作标记功能

## 需求描述

|  |
| --- |
| 为字段和记录增加操作标记的功能，区分是新增、修改还是删除；对于修改数据的流程，要区分哪些记录做过修改，并为修改过的字段提供前后值比较功能；在向业务系统同步的时候对于没有修改过的记录就不同步。 |

## 实现方案

### 表结构变更

1. 在表配置表中增加配置字段“主表标识ID（pri\_tbl\_id）、是否显示（show\_in）”；
2. 在字段值表中增加“原字段值(orig\_fld\_val)、操作标记（oper\_in）”
3. 增加表行记录状态表（tbl\_glbps\_para\_rec\_st）（FLOW\_BUSS\_ID、TBL\_ID、PRI\_KEY、REC\_NO、PARENT\_REC\_NO、ROW\_OPER\_IN、FINISH\_IN、CHK\_ST、VFY\_ST）

### 逻辑实现

1. 将以前的是否在界面显示的配置迁移到is\_show字段，以前是根据是否是子表进行的判断（目前只有NAP对上、对下线路有父子表关系）；
2. 将PRI\_TBL\_ID字段按照业务系统的主题域配置进行配置；
3. 在从业务系统同步数据到BPS的时候，同时将字段的值存储到orig\_fld\_val,fld\_val字段中，同时置oper\_in为A；
4. 在第三步保存数据的同时要插入一条记录到tbl\_glbps\_para\_rec\_st表中，其中pri\_key是根据参数表的主键用“-“串联得到的，主键字段的顺序在初始化缓存的时候保证（配置信息中要保证主表和子表的相同字段的顺序一致）；oper\_in字段为”A”，PARENT\_REC\_NO、FINISH\_IN、CHK\_ST、VFY\_ST字段目前不使用；
5. 在BPS系统对同步过来的数据进行修改的时候，界面根据每个字段值的变化（orig\_fld\_val,fld\_val相互比较）情况修改oper\_in，并根据整条记录的情况修改tbl\_glbps\_para\_rec\_st中对应记录的状态，同时还要修改关联表的字段状态和对应记录状态（根据关联字段表判断）；
6. 对于BPS新增的记录，则字段值表的oper\_in,和tbl\_glbps\_para\_rec\_st表的row\_oper\_in全程为I；
7. 在BPS系统对本支流程新增的数据进行删除的时候，同时删除tbl\_glbps\_para\_rec\_st表中对应的记录，对于有父子表关系的父表还要同时删除子表对应的记录（目前只有NAP对上、对下有父子表关系）；
8. BPS系统目前不支持对业务系统数据删除操作，保留D标记为了后续扩展；
9. 在同步数据的时候，只同步tbl\_glbps\_para\_rec\_st表中row\_oper\_in为I、U的记录，但是对于在业务系统为主表的记录要进行特殊的计算：该表自己的记录标记为I、U或者该记录的对应子表的记录为I、U，这个关系到同步到业务系统之后的创建任务操作，主子表的配置在表配置表中的pri\_tbl\_id字段中体现，主子表之间的记录关联通过pri\_key字段进行，这里有一个基本条件为：子表的联合主键一定包括主表的联合主键而且顺序一致，这样我们就可以根据主表的pri\_key去搜索子表记录、即假设主表的pri\_key长度为length，则条件为子表的substr（pri\_key，length）=主表的pri\_key；
10. 修改判断是否需要给子系统发送同步、生效消息的逻辑，以前是只判断本流程是否维护了对应子系统的参数，现在要增加判断维护的参数操作标记有不为“A”的；
11. 修改和页面的数据交互模型，增加字段原值、字段操作标记、行操作标记，如果字段修改过，则在该字段后增加一个字段值比较的控件，对于多条记录的参数表在记录列表前面增加操作标记显示是临时、新增、修改、删除；

### 存量数据处理

由于系统目前有很多存量数据，对于存量进行操作标记的维护没有价值，所以要对存量数据进行标记，并在操作标记处理的时候进行特殊处理；

* 1. 上线的时候统计目前已经做了参数录入的流程实例ID，标准是tbl\_glbps\_para\_fld\_val表中有值，则将该流程实例的业务流程实例ID插入到tbl\_glbps\_sys\_para表中去；
  2. 插入语句为：

|  |
| --- |
| INSERT INTO GL\_BPSDB.TBL\_GLBPS\_SYS\_PARA (PARA\_NM, USAGE\_TP, PARA\_VAL, PARA\_DESC, OPER\_IN, EVENT\_ID, REC\_ID, REC\_UPD\_USR\_ID, REC\_UPD\_TS, REC\_CRT\_TS ) (select distinct flow\_buss\_id,'2','','','I',0,0,'90000000',current timestamp,current timestamp from tbl\_glbps\_para\_fld\_val) |

* 1. 在查询给界面显示的时候如果是在此表中存在的流程实例在统一显示为修改的记录、以提醒用户要全部做复核；
  2. 在做数据同步的时候，如果是在此表中存在的流程实例则全部同步该流程实例关联的记录。

### 影响性分析

由于此功能的开发，整个核心服务模块都进行了修改，包括参数查询、修改、删除、同步，所以影响范围非常广泛，需要全功能的回归测试；

# 组织机构模型改造及存量数据处理

## 需求描述

|  |
| --- |
| 1）以前的组织机构模型是在根节点下挂BPS系统的所有角色组，每个角色组下挂具有该权限的人员；另外以前的用户之支持总公司的用户。  2）这个组织机构模型存在两个问题：1.不能使用多个平台的接入；2.不能实现对机构维度的权限控制，更不能实现角色组和机构的组合权限控制；  3）所以需要改造，改造后的组织机构模型应该为：根节点下挂机构和系统两个虚拟节点；机构下挂所有Ldap中的真实的机构（比如00010000），每个真实的机构下挂机构下的用户；系统下挂接入到BPS系统的所有业务系统（比如048：机构参数管理系统），每个系统下挂该系统所有的角色组，每个角色组下挂具有该权限的用户；  4）另外以前的组织机构上的人员的标识是用户ID，当组织机构模型的用户扩展到多个机构后，用户ID就不是全局唯一的了，所有要将用户ID拼装成：userId-insCode格式；  5）提供按照角色机构的交叉权限控制。 |

## 方案实现

1. 在配置表中配置组织机构模型要使用的机构代码、子系统代码；
2. 在系统初始化的时候从Ldap获取这些机构和子系统的权限信息；
3. 首先根据配置的子系统列表，逐个获取每个子系统具有的角色列表，将角色组列表前面拼装上子系统代码并放入到缓存中；
4. 根据配置的机构列表，逐个获取机构的信息放入缓存；
5. 查询每个机构下的所有人员，将用户ID拼装成userId-insCode格式，并放入缓存；
6. 对各人员进行权限信息查询，得到他具有的所有角色组，然后将人员增加到对应的角色组关联人员列表中；
7. 实现产品的接口构建组织机构树；
8. 为组织机构模型提供缓存刷新功能；
9. 修改每个和流程引擎交互的地方，将设置用户的时候将用户ID设置为拼装后的ID；
10. 修改每支流程图的参数者设置，比如以前的参与者这是201这个角色组，现在就要变成048201；
11. 修改workspace的登录验证方法，要求用户在输入用户ID的时候，输入格式为userId-insCode,然后在验证方法中在进行拆分，然后进行验证；

## 存量数据

组织机构模型的改造导致出现了两个存量数据的问题：

### 变更说明

原组织机构非常简单，根节点下为角色组，角色组下为用户。现由于接入其他系统，提供流程引擎给其他系统调用。组织机构树需要加载此系统的角色组和机构，原组织机构树中用户ID作为用户唯一标识，在不同机构下用户ID可以重复，所有新的组织机构树要改变用户ID作为唯一标示，如：用户唯一标示【用户ID-机构号】，拼接用户ID和机构号作为用户唯一标识。组织机构中存在多个系统，系统下的角色组ID可能存在重发，所有角色组ID改用系统号与角色组ID拼接。

### 产生问题

修改用户ID产生问题：

在组织机构树中修改用户唯一标识后原来产生的任务无法执行，因为在新组织机构树上已经找不到原用户标识，原任务无法执行。

修改角色组ID产生问题：

老的流程模板中角色组ID在组织机构树上不存在，会导致无法启动任务。

### 改造内容

组织机构用户ID改变涉及修改：

涉及任务参与者ID的表为（WFWORKITEM，WFWIPARTICIPANT）两张表，所以只需修改这两张表中字段即可。

WFWORKITEM表

PARTICIPANT字段 用于任务查询及workspace中任务界面显示用

ASSISTANT字段 任务执行者，最终执行任务的人员ID

WFWIPARTICIPANT表

PARTICIPANTID字段 执行任务时从此去取参与者。如果这里的参与者在组织机构中找 不到，则执行任务报错。

GLOBALID字段 待办任务查询从此去读取参与者，不修改则无法查到任务。

ASSISTANT 字段 执行者id

组织机构角色组ID改变涉及修改：

WFWORKITEM表

PARTICIPANT字段 用于任务查询及workspace中任务界面显示用

WFWIPARTICIPANT表

PARTICIPANTID字段 执行任务时从此去取参与者。如果这里的参与者在组织机构中找 不到，则执行任务报错。

GLOBALID字段 待办任务查询从此去读取参与者，不修改则无法查到任务。

PARTICIPANT字段 用于任务查询及workspace中任务界面显示用

新的流程模板中参与者全面选取新的角色组ID。

老的流程模板中参与者全部为未修改前的角色组ID，需要从数据库中导出所有流程模 板，通过程序修改所有版本模板中配置了参与者的角色组ID，然后导入数据库修改好 的数据。

### 数据修改前验证

目的：验证修改表字段是否全为人参与者或角色参与者

如：执行validate.del文件中validate wfwiparticipant语句，验证wfwiparticipart表中PARTICIPANTID，GLOBALID，PARTICIPANT，ASSISTANT字段是否全为人参与者。

执行validate wfworkitem部分语句，验证wfworkitem表中PARTICIPANT字段是否全为人参与者。

### 修改前数据统计

目的：验证数据修改前的任务和修改后的目标任务是否一致。

统计用户（如90000000）已经领取，正在运行中的任务

select count(\*) from GL\_BPSDB.WFWIPARTICIPANT where PARTICIPANTTYPE = ‘person’ and globalid = ‘P{90000000}’and currentstate = 10;

记录任务数量，并记录菜单 “任务管理-我的任务”下的我执行中的任务中记录总数。

统计用户（如90000000）用户已经完成的任务

Select count(\*) from GL\_BPSDB.WFWORK where ASSISTANT = ‘90000000’ AND CURRENTSTATE = 12;

记录任务数量

### 表备份

执行backup.sql,备份两张表数据，backup.sql中备份路径可修改。

### 更新数据

执行update.sql,更新两张表记录。若角色组ID也需修改，则需要执行注释部分的sql语句。

### 修改后数据验证

数据总数验证

更新完成后可登录应用，查看应用“insmgm”应用菜单 “任务管理-我的任务”下的我执行中的任务中记录增加总数是否与修改前数据统计的数据一致。

同样验证“任务管理-我的任务”菜单下我已完成的任务。

功能验证（选取5支流程，记录下流程实例ID）

任务查询功能：

在workspace中找到流程当前运行的任务ID，并查看状态，若为待领取的任务，则到应用中找到任务管理菜单下的BPS任务菜单，在待执行的任务界面根据任务号查询任务。若查不出此任务，则说明更新WFWIPARTICIPANT表的 GLOBALID 字段有误。

若为正在运行的任务，则到应用的任务管理菜单下的我的任务菜单，在我执行中的任务中查询此任务。若查不出此任务，则说明更新WFWIPARTICIPANT表的 GLOBALID 字段有误。

任务领取功能：

在任务管理菜单下的BPS任务菜单的待执行任务界面选中一条任务，点执行（点执行按钮即可，不需要复核），再到任务管理菜单下的我的任务菜单的我执行中的任务中查看任务ID，

能够找到此任务，则此任务领取成功。

任务领取后释放：

任务管理菜单下的我的任务菜单的我执行中的任务页面找一条已经领取的任务，点击释放按钮，然后到任务管理菜单下的BPS任务菜单的待执行任务界面查找此任务，任务能够查出，说明释放成功。

任务执行：

正常录入复核任务，检查任务是否执行成功，并到我的任务业务查看结果。

### 测试后数据恢复

备份测试数据：

将backup - update.sql文件in操作符后的括号内填写实际测试时操作的流程实例id，并执行sql。

数据恢复：

将recover.sql文件中的delete sql语句的根据时间删除记录替换为数据统计步骤后查询出的时间点并执行sql，删除时间点前的所有数据，导入更新前的数据。

同样将recover - update.sql文件中sql语句中in操作符后括号内的值替换为测试使用的流程，并执行sql，使操作的流程数据为最新数据。

## 测试要点

# 每支流程结束的时候增加任务关闭功能

## 需求描述

|  |
| --- |
| 在每支流程结束的时候，调用业务系统通知结束相关的参数维护任务(收单、跨行、文件) |

## 方案实现

### 现状分析：

根据实际调研情况分析，目前BPS牵涉的业务系统包括商户系统、跨行管理系统、收单管理系统、统一文件管理系统、资金管理平台、多渠道管理系统；这些系统只有跨行管理平台、收单管理平台、统一文件管理平台存在任务管理的概念。

### 逻辑实现：

1. 在目前已经实施的5支流程中根据实际的维护情况分别对应每个子系统增加一个自动活动来调用业务系统接口关闭任务；
2. 业务系统收到关闭任务的请求以后，根据关联表得到该流程实例关联的任务列表，然后逐个检查是否符合关闭条件，将符合条件的任务关闭；将不符合关闭条件和关闭不成功的任务id返回给BPS系统；
3. BPS系统记录每个系统的任务关闭情况；
4. BPS系统提供一个查询界面为用户提供查询功能。

# 查询复核历史功能，支持按照业务流程实例查询，并区分是测试复核还是生产复核

## 需求描述

|  |
| --- |
| 以前的复核历史查看功能只能查看到当前子流程的复核历史，不能按照统一流程业务编号来查询，这样的问题就是用户无法得到这支流程实例流转过程中的全部复核记录；这次优化为按照统一流程业务编号查询，并在查询列表中对复核阶段进行区分，分为测试方案、复核方案。 |

## 方案实现

* + - 1. 修改扩展API的查询复核历史的接口，传入的参数从流程实例ID变成统一业务流程编号；
      2. 修改界面调用的逻辑，传入参数修改统一业务流程编号；
      3. 修改界面展示增加复核阶段字段；

# 审计日志改造

## 需求描述

|  |
| --- |
| 有两个方面需求：   1. 流程的强制终止和任务的强制完成记录审计日志，并提供审计日志查询功能； 2. 将BPS系统的审计日志进行一些二次开发，提供对一些重要操作的查询功能   BPS产品提供的审计日志是比较细粒度的，目的是帮助技术人员定位问题的引擎操作日志。对于一个操作，审计日志前后会记录开始和结束两次，对于用户而言可读性不强。用户通过应用强制终止流程和强制终止任务，与引擎的终止无法区分。  引擎日志记录格式：[日志唯一编号][B/E/EX][功能类型][功能名称][用户名][时间][IP][操作级别][流程实例ID][活动实例ID][工作项ID][用户自定义信息]。 |

## 实现方案

### 记录审计日志

在用户强制终止流程实例、或者强制结束任务的时候增加审计日志记录功能，使用我们传统的日志记录方式，操作对象分别为流程、任务，操作类型为强制结束。

### 提供用户查询界面

这里的查询包括两个：

1. 提供对业务系统审计日志的查询功能，即上一节提到的对流程实例的强制结束和对任务的强制结束；
2. 对产品审计日志的查询，我们重点值关注对流程实例、活动、工作项的完成操作和对工作项的领取操作。

# 流程图业务目录划分，并根据业务目录过滤任务查询功能

## 需求描述

|  |
| --- |
| 由于BPS统一流程管理平台定位于公司级流程引擎，随着其他系统的接入，需要对流程图的业务资源目录做一下规划；目前的规划是每个业务系统拥有一个顶级的业务资源目录，可以在该目录下再分子目录；在业务系统做任务查询的时候应该按照业务资源目录进行过滤，只查出属于自己系统业务资源目录下的任务； |

## 改造方案

根据分析发现在产品的WFWIWorkitem表中有一个字段是该任务所属的业务资源目录，另外有一张表中存有业务资源目录的层级关系；根据以上情况制定了一下方案：

* 1. 先根据每个系统拥有的顶级业务目录ID（此目录是预先定好的）得到它下面的所有业务资源目录；
  2. 然后在查询任务的时候，根据这些业务资源目录拼装查询语句，例如查询语句为“ CataLogID in (?,?,?,?)”，查询值为“ 11111，22222，33333”；此处由于产品的限制只能这样拼，具体的原因产品那里也没给出解释，后续咨询是否能改进一下；

## 存量数据

由于以前没有进行严格的业务目录划分，所以存在存量数据兼容的问题，为了查询过滤时不过滤掉这些存量数据，需要在上线的时候对这些任务进行处理，处理方案为：

更新WFWIWorkitem表的所属业务资源目录字段为本业务系统拥有的顶级业务资源目录ID；

## 测试注意事项

# 将bpsmonitorApp拆分

## 需求描述

|  |
| --- |
| bpsmonitor应用之前是个大杂烩，包括管理和监控应用、流程引擎的接口扩展实现、流程引擎的一次性修改内容、流程引擎的patch、流程引擎提供给外部的服务实现、组织机构模型实现、流程图源码；这样管理非常混乱，也容易出问题；打包脚本也存在问题，很多内容存在重复打包的问题，一些一次性操作也放在了脚本中变成了每次都进行的操作。 |

## 改造方案

将bpsmonitor应用拆成4个部分：

1. 对流程引擎的一次性操作，包括安装产品后的修改操作，产品方出具的patch，并编写自动化脚本对一次性操作实现自动化，供以后流程引擎环境的重建使用；
2. 将流程引擎的接口扩展实现、流程引擎提供给外部的服务实现、组织机构模型实现以及流程引擎公共代码做一个单独的工程(defaultApp)，修改对应的打包脚本；
3. 将流程图源码和流程图用到的特殊java代码单独做为一个工程(insmgmflowApp),这个也会影响打包脚本的修改，以及发布脚本和生产环境的清理（需要考虑是否存同样的类在两次打包的时候打包到不同的地方）
4. 将流程模拟启动功能单独拆分成一个web工程，命名为simulateApp，后续考虑将他持续完善
5. 将剩余的东西作为bpsmonitor，这个就是纯粹的管理与监控应用了。

## 注意事项

这个改造过程中牵涉到大量代码的归属问题，有些可能还存在共用；遵循一个原则，defaultApp致力于为流程图应用提供基础服务，insmgmflowApp中流程图用到的特殊代码先不重构、保持现状；

改造后牵涉到打包脚本和发布脚本的大变动，要做好前后两个脚本的结果比对，做好生产环境的清理工作。

# 修改数据的流程优化

## 方案：

# 自动活动代理类的实现

## 实现方案

1. 修改bpsconstant增加ESB\_URL的配置，只配置csb的地址，去掉配置UOSP\_ESB\_URL、INSMGM\_ESB\_NAMESPACE、INSMGM\_ESB\_METHODNAME、UOSP\_ESB\_URL、UOSP\_ESB\_NAMESPACE、UOSP\_ESB\_METHODNAME；
2. 在EOS\_DICT\_ENTRY表中配置每个系统的服务URL，增加type为CUPBPS\_ESB\_SERVICES的配置项；
3. 在自动活动调用中根据type取到服务地址；
4. 组装流程实例信息和该流程实例的相关数据信息为报文
5. 调用外部服务
6. 将返回报文中的相关数据设置到流程实例中