Design and Implementation of Process Rollback Mechanism Based On Activiti For Power

Jie Zhao\textsuperscript{1, *}, Yitian Liu\textsuperscript{1}, and Jiantao Peng\textsuperscript{1}

\textsuperscript{1} Product research and Development Department II
Nanjing NARI Information and Communication Technology Co.

* 15171460@qq.com

Abstract. With the development of power system business, the complexity of business processes and the frequency of interaction between processes are getting higher and higher. The complexity of interaction between processes is not only limited to the functions of starting, sending and returning to the main process after the end of the sub-process, but also needs the support of the rollback mechanism of the main sub-process. Because the main sub-process can be nested in multiple layers, the rollback mechanism of the main sub-process is relatively complex.

Keywords: Workflow; Rollback; Activiti.

1. Introduction

A workflow system is a software system that uses computer technology to define, execute, and manage workflows. It can coordinate the automatic transfer of documents, information, or tasks between multiple participants according to predetermined rules. Workflow systems include: workflow model designer, workflow engine, workflow client, workflow monitoring, and other components\cite{1}.

As an important function of the workflow engine, rollback refers to an operation that a workflow participant performs on their own "to-do tasks" (actually work items), that is, the participant actively rolls back tasks in the to-do task list to manual nodes that have already been executed. The current rollback function can already well solve the functions of level-by-level rollback, cross-level rollback, branch rollback, and aggregation rollback between unified processes, but it does not have good support for cross-process rollback functions. This article researches and implements the requirements of cross-process rollback.

In order to enhance the flexibility and agility of the rollback function across processes, this article proposes a design and implementation of a cross-process-level rollback rule and algorithm based on the open source workflow engine activiti, which provides better support for the process rollback mechanism between processes.

2. Cross-process rollback process analysis

2.1 Process definition

Main process: mainly showing the main steps and nodes
Sub-process: Other processes initiated to complete the main process
Process template: Each process template represents a business process, consisting of several steps, such as the leave form process, power outage maintenance process, etc.;
Process instance: A published process version generates a process instance during runtime. Each time a process version is launched, it generates a process instance. The relationship between process versions and process instances is one-to-many.
Current task: Also known as pending task. Refers to the task queue that the user has permission to process, waiting for the current user to process.
Historical tasks: Also known as completed tasks, these refer to tasks that have been processed and completed by the current user.
Rollback is a commonly used function in workflow systems. The conventional rollback mode is to arbitrarily roll back to a processed task node within the process. With the development of business, a new rollback mode has emerged, which is cross-process rollback. Any internal rollback within the process includes: sequential rollback, branching back to the main branch, and main branch back to the branch. Next, we will introduce the cross-process rollback[2] [3].

2.2 Rollback process analysis

![Fig. 1 Two process picture](image)

As shown in the Fig. 1, sub-process activity 1 in process 1 can be bound to a remaining flowchart process 2. Complex processes can be nested with multiple levels of sub-processes.

1. Subtask 1 of process 2 is rolled back to task 2 of process 1
2. Terminate or delete the current process 2, obtain the relevant configuration of the sub-process activity 1, and perform a rollback operation. If it is sent to the sub-process activity 1 again, it will be activated or restarted again.
3. The task 3 of process 1 is rolled back to the subtask 1 of process 2
4. Terminate task 3, restart process 2, and send it to subtask 1

3. Cross-process rollback process rules and algorithms

3.1 Rollback rule

The cross-process rollback implemented in this article needs to meet the following rules

Rule 1: The process instance must be a running process instance, the activity that can currently use the rollback function must be the current task, and the activity that is rolled back to must be a completed historical task.

Rule 2: The main process can be rolled back to any node of the executed sub-process. The sub-process can be rolled back to the node before the sub-process activity primitive in the main process. The sub-process can be rolled back to an internal node of another sub-process in the main process.

When reverting to other process nodes unexpected by the current process, you can choose to delete the current process or suspend it. When choosing to suspend the current process, the workflow service will directly resume the current sub-process when the parent process returns to the current node.

3.2 Rollback algorithms

In order to solve the problem of complex business process rollback in the power system, this article designs and implements a rollback algorithm between the main and sub-processes. The algorithm first provides the set of activities that can be rollbacked at the current task node of the process and the dependent sub-process graph element information. If encountering sub-process activity graph element information, it uses a recursive algorithm to trace back upwards. Next, it calls the rollback execution algorithm, which designs a unified rollback target node expression, uses a directory structure to represent the relationship between processes, uses relative paths to represent
target nodes, and uses a recursive algorithm to trace back upwards when encountering nested processes. Next, we will focus on introducing these two algorithms.

Algorithm 1: Obtain the set of backable activities of the current task node

Provides a query that includes the backable target nodes of sub-processes: uses recursion to query all sub-processes initiated by the current process and the activity nodes traversed by the parent process, and combines all results as the query result.

Algorithm 2: Inter-process rollback execution algorithm

As shown in Fig. 2, Design a unified expression for the backtracking target node: Use a directory structure to represent the relationship between processes, and use relative paths as a representation of the current node: Backtracking from the human activity 2 node (act2) of sub-process 1 (act4) to the human activity 1 of the main process can be represented as:../act2. When the human activity 1 of the main process completes the activity and wants to return to the human activity 1 inside the sub-process, it can use act4/act2 to represent the backtracking target;

![Flowchart of rollback algorithm](image)

**Fig. 2** Flowchart of rollback algorithm

4. Realization and verification of power system cross-process rollback

4.1 Overall framework

In order to implement the rollback algorithm, this article expands and adds a process rollback manager to the open source activiti-based power system platform[4] [5]. As shown in the Fig. 3., the rollback manager is divided into the following parts: script event processor, rollback command processor, rollback processor, and remote dispatcher. The script event processor is a tool for processing business data. Through the script event processor, business data can be restored to the required state of business data needed by the target activity node; the rollback command processor is used to process the rollback request, obtain the cross-process sub-activity graph configuration information from the process database, and remotely or locally call and execute the data operation for rollback; the remote dispatcher is used if the call between processes is a cross-service call
between different urls for rollback. Then, it needs to use the remote dispatcher to remotely call the service interface for related processing operations.

![Implementation architecture diagram](image)

**Fig. 3 Implementation architecture diagram**

### 4.2 Realization based on activiti

The functions that need to be implemented for the rollback manager added to the power system infrastructure platform based on the open source engine activiti are:

1. Obtain all the associated process backout target activities for the currently active process, including all completed nodes for the main process and sub-processes;
2. Delete or suspend the current activity based on the parameters, determine whether multiple levels of main and sub-processes are nested, and process recursively.
3. Return to the target activity and complete the recovery of process data
4. Complete the recovery of business data through script events.

Therefore, in order to achieve the above functions, the power system infrastructure platform based on the activiti open source engine has been extended with functions, adding the command processing class GetAllRelatedFinishedActivitiesCmd to implement the function of obtaining all the target nodes that can be rolled back for the current task; the handlerRollbackSub of ProcessRollbackCmd and JumBetweenProcessInstancesUtil implement the functions of deleting or suspending the current activity based on parameters, determining whether multiple levels of main and sub-processes are nested, recursively processing back to the target activity, completing the recovery of process data, and implementing functions (2) and (3).

### 4.3 Verification

To verify the improved process execution effect, a flowchart of the interaction between the main process and sub-process was constructed using the Activiti model designer, as shown in the figure. When the process flows to sub-process 3 and a rollback is performed, the page retrieval method is first used to obtain a list of active nodes that can be retried for the current task, as shown in the table below:
Select a manual activity 2 node in the cross-process sub-process 2 from the table as the rollback target, and assemble the parameters for calling the rollback method, as follows:

seqAtvdId: ".../callact2/act2", when performing a rollback operation, the flowchart and data are also rolled back, achieving the correct rollback of the process. The operation results are as follows:

<table>
<thead>
<tr>
<th>No.</th>
<th>Activity</th>
<th>Responsible</th>
<th>Status</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Activity1</td>
<td>padm</td>
<td>处理</td>
<td>2024-03-14 10:46:27</td>
</tr>
<tr>
<td>2</td>
<td>Activity2</td>
<td>padm</td>
<td>已处理</td>
<td>2024-03-14 10:46:33</td>
</tr>
<tr>
<td>3</td>
<td>Activity3</td>
<td>padm</td>
<td>处理</td>
<td>2024-03-14 10:46:53</td>
</tr>
<tr>
<td>4</td>
<td>Activity4</td>
<td>padm</td>
<td>处理</td>
<td>2024-03-14 10:47:09</td>
</tr>
<tr>
<td>5</td>
<td>Activity5</td>
<td>padm</td>
<td>已处理</td>
<td>2024-03-14 10:58:01</td>
</tr>
</tbody>
</table>

5. Summery

This article studies the workflow rollback mechanism and implements a rollback application model based on Activiti, which solves the problem of cross-process rollback between the main process and sub-processes of the process. Future work will focus on improving the rollback algorithm to provide better rollback support for some special workflow patterns.

References