工作流系统异常处理框架的研究与实现

The Research and Implement Of the Exception Handling Framework in Workflow System

王　辉　（合肥工业大学计算机与信息学院 安徽合肥 230009）
吴小志　（南京航空航天大学信息科学与技术学院 江苏南京 210016）

摘要：工作流管理系统负责业务过程的建模和执行，这些业务过程往往涉及到多个参与者，需要使用分布的资源，调用多个软件系统，而且时间跨度很长，因此在工作流执行时可能存在多种潜在的工作流异常。本文设计并实现了 WSEHF（Workflow System Exception Handling Framework）核心系统模块的大部分功能。异常处理交由异常管理器集中控制，异常管理器在执行监控器、恢复管理器和人工处理器的协助下能够提供比较完善的可预测异常处理机制，并为处理不可预测异常提供了良好的支持。

关键词：工作流 异常处理 ECA 规则 工作流系统异常处理框架

1 引言

工作流管理系统中不可缺少的一个重要部分是异常处理能力，异常是指任何对原协同处理过程的变化偏移。随着工作流技术的飞速发展，工作流在企业信息化过程中也起着越来越重要的作用[1]。可是，由于大规模、分布式信息系统的复杂性增强、业务过程的动态变化以及工作流执行时各种不确定因素的存在，使得工作流在执行时经常出现各种错误和异常。对于制造企业这样一个复杂的应用系统，出现异常和错误是非常正常的情况[2]。因此，工作流异常处理是工作流系统设计和工作流执行中要解决的关键问题之一。本文提出了较为灵活的工作流系统异常处理框架 WSEHF（Workflow System Exception Handling Framework），详细设计并实现了 WSEHF 异常处理框架的核心功能模块，给出了可预测异常处理基本流程的具体实现。

2 异常的分析及分类

Claus Hagen 认为引起工作流异常的原因有技术原因和用户原因，技术原因包括通讯问题、计算机断电、程序失败等；用户原因包括工作流模型错误、系统变化、缺少雇员等。G. Alonso 认为工作流异常包括系统失败和语义失败，系统失败是指工作流管理系统自身的失败，语义失败是指工作流中某个活动的执行所遇到的失败，如发送没有库存的物品或者从空的银行帐号上提款等[3]。

工作流运行过程中，会经常发生异常，一方面，工作流管理系统要具有动态自适应能力，适应实际应用中的动态变化；另一方面，系统本身在运行过程中，因为内部本身操作时的数据以及底层支持的网络、数据库的异常操作等都可能导致流程运行中出现异常。

实际上，产生工作流异常的原因很多，只要与工作流的执行相关的因素，包括硬件、软件、通讯、工作流模型、工作流系统（主要是工作流引擎和应用程序接口）、工作流执行者、相关应用程序、过程逻辑约束、工作流相关数据约束、时间约束以及执行算法（资源分配算法、活动执行调度算法）等都可以引起不同种类的工作流异常。

人们从不同的角度对工作流中的异常进行了分类，最常用的分类方法是 Eder. J 等人提出的，将异常分为 4 类：基本故障类，应用故障类，可预料的异常和不可预料的异常[4]。基本故障指工作流系统执行时出现的故障或执行环境的故障，应用故障指的是工作流调用的应用程序出现故障。可预料的异常指工作流中可预见的正常流程的偏离，不可预料的异常指一个业

务流程的执行结果与它的原始定义不匹配。

3 建模工作流异常

工作流异常是事件驱动的，这些事件可能源自于数据更新、对象状态改变、时间事件、资源冲突等。典型的工作流过程可能存在大量异常情况需要特殊处理，这些处理不是正常执行工作流过程的组成部分。

在 WSEHF 框架中，一个给定的工作流过程中的异常可以被描述为：

$(O,E)\rightarrow H$

表示如果工作流可执行对象 O 出现了异常情况 E，则执行异常处理(子)程序 H。允许多个异常情况 E 能够使用相同的异常处理(子)程序 H，因此异常处理(子)程序 H 对于指定的异常情况 E 不具备唯一性。

异常处理程序可以被建模为子活动，异常管理器具备用来将控制权转移给指定的异常处理程序、改变发生异常的任务或活动实例所包含的子活动或任务节点状态、最后依照它的类型恢复该实例的执行的能力。图 1 运行时异常处理流程图。

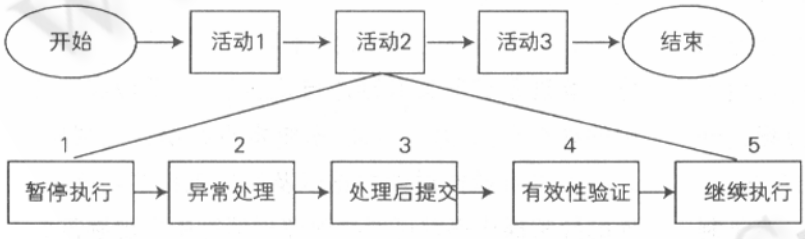

图 1　活动级异常处理基本流程

异常处理程序可以被建模为工作流过程，用来动态地标示出引发异常的活动或任务实例的后续执行路径。这种方法相当于整合一个实例到一个其它类型的工作流中，也就是引发异常的过程被整合到异常处理程序。WSEHF 异常处理框架在过程级对异常处理的支持体现在建立阶段明确对可预测流控异常的描述和在运行阶段动态重定向不确定性因素。

4 WSEHF 的核心系统结构

在如图 2 所示的 WSEHF 核心系统结构中，异常处理交由异常管理器模块集中控制，异常管理器模块在执行监控器子模块、恢复管理器子模块和人工处理器子模块的协助下能够提供比较完善的可预测异常处理机制，并为处理不可预测异常提供了良好的支持。在正常的处理模式下，如果执行监控器检测到异常事件的发生，系统将授权异常管理器执行处理。异常管理器通过诊断异常的类型进而确定适当的异常处理程序，由异常处理程序实际执行处理工作流失败恢复中的补偿逻辑操作。如果异常处理程序成功地解决了问题，则异常管理器返回控制权；如果异常处理程序在执行过程中需要撤销异常已经产生的负面影响，则异常管理器调用恢复管理器重返一致性状态；如果自动控制模式下异常无法得到有效解决，则可以通过人工处理器请求系统辅助和参与者干预。执行监控器在异常任务的执行过程中进行全程跟踪，并将相关信息记录到历史数据库，为执行异常处理提供参照。

4.1 恢复管理器

恢复管理器通过执行撤销操作使受到异常影响的作用域在异常管理器执行相应的异常处理程序之前重返一致性状态，具体的撤销行为受到异常对象的实现类型以及当前状态的制约，因此在执行撤销之前必须进行必要的判断。

恢复管理器的主要操作包括：

(1) 计算同异常对象数据依赖的其它对象。

(2) 计算同异常对象控制依赖的其它对象。

(3) 计算受到异常影响的作用域。

(4) 在指定作用域内屏蔽异常。

(5) 撤销指定作用域内已经执行的操作。

(6) 计算指定作用域内对象的普通消费者。

(7) 触发普通消费者的不一致性异常。

4.2 执行监控器

在整个工作流过程实例的执行过程中，执行监控器全程跟踪实例的执行序列、状态转变、数据传递以及同该实例存在依赖关系的其它对象，并将相关信息包括控制流和数据流记录到历史数据库，为异常管理器执行异常处理和恢复管理器执行撤销操作提供必要参照。如果执行监控器监测到实例的执行行为不在系统预定义的正常行为之列，则立即发出异常消息通知异常管理器进行相应的处理。执行控制器必须为异常管

理器诊断异常类型并确定适当的异常处理程序提供足够的信息项，这些信息项通常包括异常源、异常类型、实例发生异常前后的状态变迁以及任何在异常类型中定义的额外参数等。图 3 为执行监控器的基本概图：

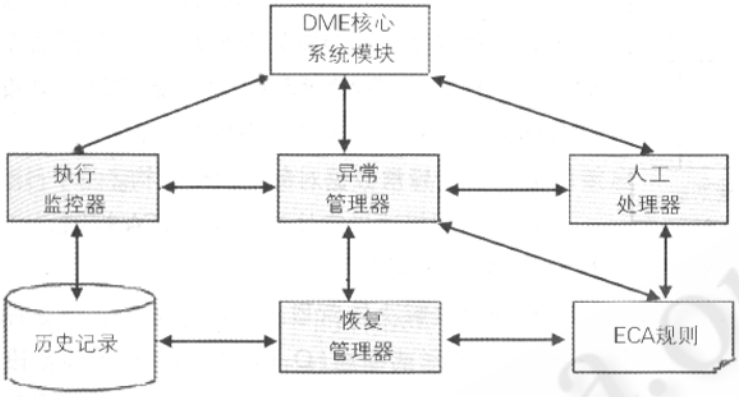

图 2　WSEHF 核心系统结构

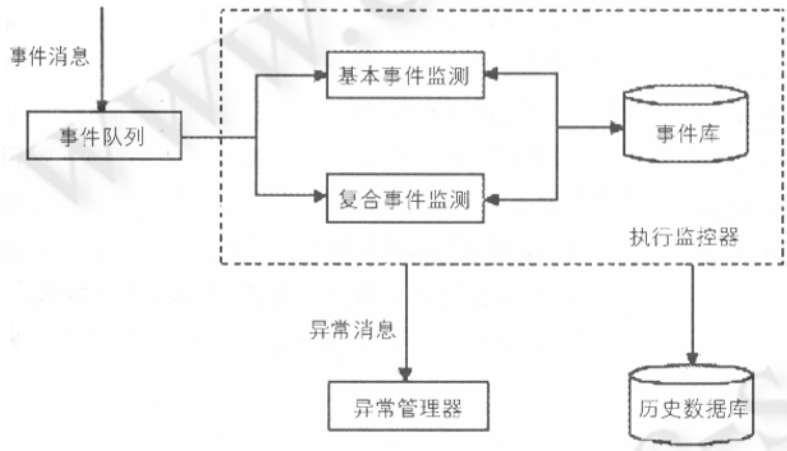

图 3　执行监控器基本概图

4.3 异常管理器

异常管理器根据检测到的异常事件诊断异常类型并确定适当的异常处理程序执行处理，异常管理器在完成异常处理后返回控制权。异常管理器可能需要调用恢复管理器撤销异常已经产生的可能的负面影响，或调用人工处理器请求系统辅助和参与者干预来帮助做决策。

如果可执行对象实例在执行过程中返回异常或没有在最终期限内响应，或者执行监控器检测到非正常行为的发生，则系统立即挂起运行中的实例并授权异常处理器进行处理。异常管理器接管控制权，并执行如图 4 所示的操作来处理异常。

4.4 人工处理器

人工处理器提供系统辅助和参与者干预的异常处理模式来协助处理致命性或突发性的异常事件。人工处理器调用客户端应用程序接口在 Web 页面中为指定的参与者添加协助处理异常的任务项，并以循环消息的方式请求参与者关注任务。Web 页面中会提供可重用的现有异常处理程序、解决方案和相关的数据对象列表来辅助人工决策。人工处理器支持用户在运行时修改所有声明和关联。需要注意的是，在执行外部指令之前人工处理器必须进行可行性验证来避免发生级联异常，如果同某些工作流约束发生冲突，异常管理器将进一步通知参与者来关注可能的（潜在的）问题。在整个人工处理的过程中必须保证工作流实例总体状态得以正确维护。

4.5 历史记录

工作流历史管理通过详细记录可执行对象执行过程中所有的控制流和数据流等相关数据信息，包括正在运行和已经运行完毕的活动实例或任务实例的状态转变、数据传递等，为工作流异常处理和工作流失败恢复提供支持，日志管理协同历史管理为工作流失败恢复提供动态的参照依据。

4.6 ECA 规则

ECA（Event – Condition – Action）规则法[5,6]：事件描述了一个潜在的异常情况，条件用来表示当事件发生时的处理前提，动作是指是对异常事件应做出的反应。在异常事件发生时，根据预先定义的规则判断条件状态，然后决定所要采取的动作[7]。该规则可以配合异常处理知识库实现。

WSEHF 异常处理框架中主要使用 ECA 规则来模拟异常处理，工作流约束和声明型异常处理程序都被定义为 ECA 规则，异常管理器通过查看是否为某异常事件定义任何 ECA 规则来确定异常处理程序。

本文中 ECA 规则被表示为五元组：

（Object，Event，Condition）（Executor，Action）

表示如果可执行对象 Object 发生了 Event 异常事

件,如果满足条件 Condition 或条件 Condition 为真,则执行对象 Executor 就执行 Action 所声明的异常处理措

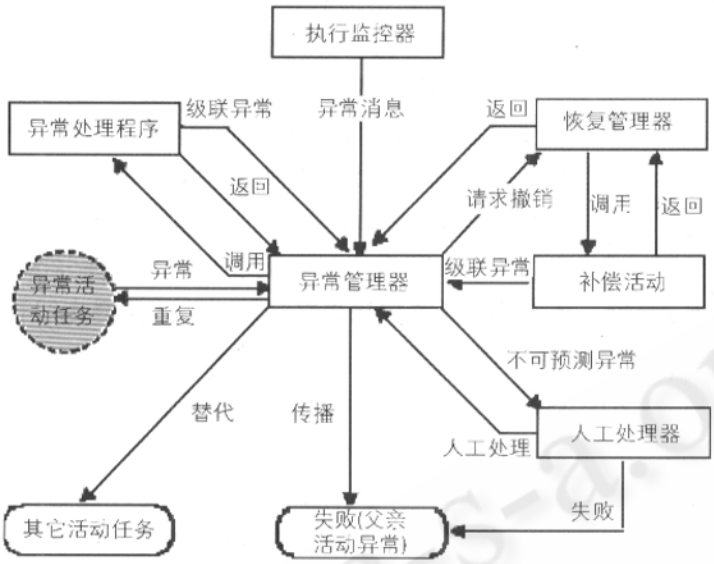

图 4　异常管理器执行逻辑

施。其中 Object 和 Executor 都是可选属性,五元组中的每个元素都具有若干相关子属性。为了在异常处理过程中实现 ECA 规则的重用,每个 ECA 规则都可以被组合用于复杂类型的异常事件,但是对复杂事件的诊断和异常处理程序的确认通常需要系统辅助和人工决策。

5　异常处理的基本流程

异常处理包括三个典型的阶段,分别是异常检测阶段、异常诊断阶段和异常处理阶段,每个阶段都在 WSEHF 框架的核心功能模块的控制下进行。首要的是对异常情况进行系统计算(当异常通过异常规则被诊断时)或者是为异常情况提供外部指定(当异常通过异常过程被诊断时)。一旦异常管理器确定了异常的类型,必须定位适当的异常处理程序进行有效的处理。

5.1　检测异常发生

异常检测阶段的目的是要使系统中异常事件的发生可以及时被发现,异常的形式描述(O,E) H 中的 E 是本阶段的关键元素,而且触发异常事件的对象必须能够提供足够的上下文环境信息来计算 E。

(1) 首先在工作流建模阶段确定可能导致工作流失败的途径。

(2) 通过实例返回值判断异常是否发生。

(3) 执行监控器运行时全程跟踪实例的执行过程。

5.2　诊断异常类型

一旦检测到异常的发生,必须对异常类型进行诊断以便确定适当的异常处理程序,异常的形式描述(O,E) H 中的 是本阶段的关键元素,ECA 规则和工作流约束为诊断异常类型提供依据。

(1) 分析异常数据对象的数据结构。

(2) 匹配描述异常处理过程的 ECA 规则。

5.3　处理异常问题

如何有效解决异常取决于异常处理方案是否完善,异常的形式描述(O,E) H 中的 H 是本阶段的关键元素。除了嵌入式异常处理之外,异常通常是通过调用异常处理程序,并(暂时)挂起引发异常的实例来解决。完成异常处理程序之后如何恢复被挂起的实例也是异常处理重点要解决的问题。分两种情况:

5.3.1　异常发生在就绪态

如果异常发生时异常实例处于就绪态,则有如下几种可能:

① 控制权可以不返回给该实例。

② 如果控制返回给该实例,则实例被重新就绪。

③ 控制返回给该实例后实例可能在不同点上恢复执行。

④ 异常处理程序也可以与实例平行执行。

其中在③这种特殊情况下,异常处理程序通常是作为另外的商业过程的试图来实现,用来描述一个指定的异常。

5.3.2　异常发生在执行态

如果异常发生时异常实例处于执行态,则有如下几种可能:

① 控制权可以不返回给该实例,实例被异常中断;

② 如果控制返回给该实例,则实例被重新激活;

③ 控制返回给该实例后实例可能在不同点上恢复执行;

④ 异常处理程序也可以与实例平行执行。

其中在③这种特殊情况下,异常处理程序通常是

作为另外的商业过程的试图来实现，用来描述一个指定的异常。

5.4 进化工作流

对于频繁出现的极具相似性的异常情况需要在工作流执行期间动态修改工作流实例和工作流定义，这是通过消除存在异常隐患的活动、添加可供选择的路径等方式来实现的。WSEHF 框架中工作流进化是通过人工干预实现的，在执行工作流进化时必须首先终止运行中的整个工作流实例，在应用进化结果之前重启工作流实例。

6 工作流失败恢复中的补偿逻辑

在出现异常的情况下补偿逻辑是非常重要的，对于已经完成的活动，撤销操作可以通过反向执行补偿活动来实现。将 T 分解成的实例序列集合 T1，T2，……，Tn，集合 C 是包含了集合 T 中的每一个活动所对应的补偿活动 C1 C2 C3 ……Cn，系统有两种执行路线，一是顺序执行了 T1 T2 T3 ……Tn ，这时 T 提交；二是顺序执行 T1 T2 T3 ……Ti Ci……C2 C1 ，（0 < i < n），这时 T 取消。图 5 是一个只有三个实例序列恢复模型。实际上这和许多 Saga 的扩展事务模型，例如并行 Saga 事务模型等的基本原理都大同小异[8]。

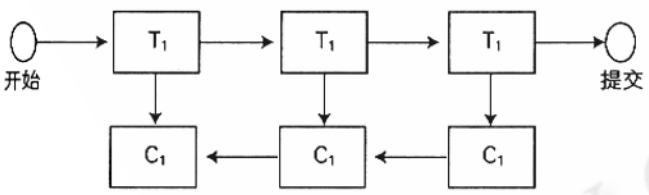

图 5　异常恢复简单流程图

工作流失败恢复处理主要是通过异常管理器向恢复管理器子模块发出恢复请求，恢复管理器首先使用历史记录回退并逆转指定作用域内的每一操作，反向生成补偿序列，然后调用相应的补偿活动反向执行对作用域内已经成功完成的操作的补偿，使作用域在发生异常的活动被重新执行之前恢复到初始状态。如果补偿活动在执行过程中发生异常则反复重新执行能够补偿活动直到成功为止。

ECA 规则可以用来关联补偿活动到受影响的可执行对象，通过计算条件表达式和相应的变量来实现补偿活动的重用。

7 总结

异常处理成为提升工作流管理系统容错性和自适应性的关键和热点，本文通过分析工作流异常处理的相关问题，使用了一种多层面的解决方案来整合并推进可实现的异常处理方法。工作流异常解决方案的每个实施阶段都在 WSEHF 异常处理框架的核心功能模块的控制下进行。

WSEHF 异常处理框架提出了一个以异常为中心的工作流模型，有效地实现了工作流异常处理逻辑同工作流执行控制逻辑的分离；WSEHF 框架中异常处理交由异常管理器模块集中控制，异常管理器模块在执行监控器模块、恢复管理器模块和人工处理器模块的协助下能够提供比较完善的可预测异常处理机制，并为不可预测异常处理提供支持。

参考文献

1 丁正国、许炜、李冰，作流异常处理技术与方法[J]，算机与数字工程，2005，33(11)：22－25.

2 范玉顺、罗海滨、林慧苹，等. 工作流管理技术基础[M]，北京：华大学出版社，2001. 52－53.

3 李伟平、范玉顺，工作流系统的异常处理[J]，高技术通讯，2004，14(12)：50－54.

4 孙瑞志、史美林，基于内在事务的工作流异常处理方法[J]，计算机工程，2005，31(12)：33－35.

5 Hagen C，Alonso G. Exception Handling in Workflow Management Systems[J]，IEEE Transactions on Software Engineering，2000，26(10)：943－958.

6 Grefen P. Pernici B. Schez G. Database Support for Workflow Management：the WIDE project[M]，Boston：Kluwer Academic Publishers，1999.

7 张志君、范玉顺，工作流系统异常处理实现方法[J]，高技术通讯，2004，14(8)：62－66.

8 Kuo D，Lawley M，Liu Chengfei et al. A Model for Transactional workflows[J]，Australian Computer Science Communications，1996，18(2)：139－146.