

基于 ECA 规则的 BPEL 异常流程定义方法

闵丽娟, 卢捍华, 王亚石
(南京邮电大学, 江苏 南京 210003)

摘要:当前进行 Web 服务编排的首选标准流程语言是 BPEL4WS。在流程编排过程中,除了关注正常情况下的流程定义场景,如何清晰、快速地描述 BPEL 流程中出现的异常也同等重要。文中把数据库领域的 ECA 技术用于 BPEL 异常流程定义中,通过对实际业务流程进行分析,把流程中出现的异常分支都映射成 ECA 规则;此外,提出了 BPEL 规范中的异常定义元素与 ECA 元素的对应关系;在上述工作的基础上,实现 BPEL 异常流程的快速定义。通过对电信企业的施工过程进行分析,验证了该方法的可行性。

关键词:ECA 规则;BPEL;异常

中图分类号:TN914

文献标识码:A

文章编号:1673-629X(2011)01-0058-04

A Method Based on ECA Rules for Defining BPEL Exception Handling

MIN Li-juan, LU Han-hua, WANG Ya-shi

(Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: BPEL4WS was the preferred language for Web Service Orchestration. In addition to defining normal processes, it was important how to describe exceptions during the execution of BPEL processes more clearly and quickly. ECA rules of database fields was used for defining BPEL exception handling in this paper. All the exceptions during the execution of business processes were analyzed and mapped according to ECA rules. It was proposed about mapping between activities about exception handling and ECA elements. BPEL exception handling were defined based on the above work. An example about installing process of telecom industry was given to prove the feasibility.

Key words: ECA rules; BPEL; exception

0 引言

将相对简单的 Web 服务按一定的业务流程逻辑组合起来,构成具有复杂功能的组合服务,可以为用户提供更强大、更完整的业务功能。当前进行 Web 服务编排的首选标准流程语言是 BPEL4WS^[1] (简写成 BPEL)。通常情况下,定义正常的业务流程只是流程总体设计工作的一小部分。由于业务过程的复杂和多变,流程执行过程中出现的异常问题显得尤为突出^[2]。因此,如何快速、正确的定义异常流程,也成了 BPEL 应用过程中的一大难点。

ECA 规则指的是事件—条件—动作<Event-Condition-Action>规则,表示在某一引发事件发生时,如果条件满足,则执行用户预先设定的动作。ECA 规则

主要用于主动数据库领域,具有表达能力强、实现相对简单且技术成熟的特点^[3],因此,很多研究人员将 ECA 规则用到了流程建模等方面^[3-5]。受上述研究工作的启发,文中重点研究了 BPEL 的异常处理机制,在将 BPEL 规范中的异常定义元素与 ECA 元素相对应的基础上,提出了一种 BPEL 异常流程定义方法;最后根据实例建模,进行该方法的验证。

1 BPEL4WS 规范

BPEL4WS 是由 IBM、Microsoft 和 BEA 公司联合制定和发布的基于 Web 服务的业务流程表示方法^[6]。BPEL4WS 位于几个 XML 规范之上:WSDL、XML Schema 和 XPath。WSDL 消息和 XML Schema 类型定义提供了 BPEL4WS 流程所用的数据模型。XPath 为数据处理提供支持。所有的外部资源和伙伴被表示成 WSDL 服务。

BPEL4WS 不但可以定义企业内部的流程之间,以及企业与合作伙伴之间使用 Web 服务进行交互的情

收稿日期:2010-04-29;修回日期:2010-07-17

基金项目:国家科技支撑计划基金资助项目(2007BAH17B04);南京邮电大学创新计划(NY207069)

作者简介:闵丽娟(1975-),女,湖北襄樊人,讲师,研究方向为电信运营支撑系统。

况,也提供了这些组合服务的执行过程中出现异常情况时所采取的异常处理手段。

BPEL 规范中定义了“故障处理程序”来实现对异常的处理。故障处理是指发生异常后,BPEL 可以采取的补救措施,例如重新执行当前服务、对已完成的服务进行撤销操作,或者调用其它的服务等。故障处理机制的关键元素是 FaultHandlers。FaultHandlers 类似于 try-catch 结构,它包含 catch 元素和 catchAll 元素,每个都提供活动为特定类型的错误条件进行异常处理。BPEL 规范定义的故障处理程序,使得异常处理与正常的流程逻辑分开。与 Java 等语言类似,用户可以在业务流程中添加 FaultHandlers 元素来捕获并处理相应的异常。当异常发生时,BPEL 正常执行流结束,控制流转入 FaultHandlers 内执行。

故障处理的定义格式如下:

```
<faultHandlers>
  <catchfaultName="QName1"
    Activity1
  </catch>
  <catch faultName="QName2"
    Activity2
  </catch>
  <catchAll>
    Activity3
  </catchAll>
</faultHandlers>
```

其中,faultname 表示异常的名称,ActivitX (X=1, 2,3...)表示不同异常发生时采取的相关补救措施。在定义 ActivitX 时,既可以用基本活动来表示,也可以采用结构化活动来对基本活动进行组合。其中基本活动包括:调用某个 Web 服务的操作 (invoke);等待一条消息来响应由某人从外部进行调用服务接口的操作 (receive);生成输入/输出操作的响应 (reply);等待一段时间 (wait);把数据从一个地方复制到另一个地方 (assign);指明某个地方出错 (throw);终止整个服务实例 (terminate);或者什么也不做 (empty)。结构化活动包括:定义一组步骤有序的序列 (sequence);使用“case-statement”方法来产生分支 (switch);定义一个循环 (while);执行几条可选路径中的一条 (pick);指明一组应该并行执行的步骤 (flow);在并行执行的一组活动中,可通过使用链接 (link) 来指明执行顺序的约束,并允许递归地组合结构化活动,以表达任意复杂的流程定义。

2 BPEL 异常流程定义方法

按照异常的产生原因进行分类,流程执行过程中发生的异常通常包括两种:系统异常和任务异常^[7]。

系统异常指在流程执行过程中,由应用系统直接激发的异常。例如被调用的服务部署成功后没有正常启动或者被人为关闭,此时应用系统无法调用而产生的异常。任务异常主要是指由任务的执行者或执行程序激发的异常,这类异常主要是由于任务输入参数错误、资源不足等原因而导致被执行的任务无法正确完成而产生的异常。

一般来说,系统异常应该在程序实现时得到解决。因此文中所讨论的 BPEL 中的异常主要指任务异常。

在 BPEL 规范的故障处理机制中,针对不同的异常原因,系统可以采取不同的措施。因此,文中把 BPEL 规范中定义的任务执行映射为 ECA 规则中的 Event;规范中定义的故障名称映射成 ECA 规则中的 Condition;规范中定义的基本活动映射成 ECA 规则中的 Action。具体见表 1。

表1 ECA 规则与 BPEL 异常定义
元素之间的对应关系

ECA 元素	BPEL 元素	元素解释
事件 Event	<invoke>	任务执行
条件 Condition	<faultname>	异常原因
动作 Action	<empty>	忽略
	<invoke>	调用、重试、替代
	<compensate>	补偿
	<exit>	中止
	<wait>	等待

其中,Action 支持的作类型解释如下^[8]:

1) 忽略。当任务是否成功执行对后续任务不产生影响时,可以忽略该任务出现的异常。这种情况下系统可以由异常处理状态直接回到正常运行状态。

2) 重试。任务执行失败后,可通过重新指定执行条件来重试该任务,直到任务运行正确完成。

3) 替代。如果任务失败且有必要,可以选择备用路径来替代当前任务。

4) 补偿。如果任务执行失败,可以对部分已经提交的任务进行回退处理,使流程实例恢复到一个可以重新执行的正常状态。

5) 中止。若在工作流执行过程中,出现的异常使得整个流程不能继续执行下去,且已执行的行为无须进行恢复或没有办法恢复,只能中止整个流程。

6) 人工恢复。如果在故障定义程序中没有找到发生的异常原因,则系统可以在缺省异常定义中抛出“等待人工恢复”异常,等待人工的后续处理。

7) 复杂组合情况:上面各种处理方式的组合。对于复杂的异常处理过程,需要把 BPEL 规范中定义的各种结构化活动和基本活动联合起来,以达到异常处理的目的。

在上述分析基础上,文中提出基于 ECA 规则的 BPEL 异常流程定义的步骤:

(1)分析业务流程,把正常业务流程与异常处理剥离;

(2)按照事件、条件、动作三者的关系,把流程中的异常处理进行归类 and 对应;

(3)根据 ECA 规则与 BPEL 规范中异常机制元素的对应关系,在 BPEL 正常流程的基础上补充异常处理部分。

3 实例验证

3.1 流程实例

文中以电信企业的施工过程为例来说明异常流程定义的步骤。通常情况下,电信企业的 CRM 系统受理完用户的业务申请后,通知资源系统配置相关资源,资源配置完成后,由资源系统通知施工系统开始施工。具体施工过程如图 1 所示。

施工系统接收到相关信息后,生成工单,工单分为自动工单和人工处理工单两种。对于自动工单,由机器自动处理,如果处理失败,则等待人工提取;对于人工工单,分为三类:外线工单、测量台工单、程控工单,需要分别施工。外线工单在施工的过程中,如果处理

成功,则正常回单;如果发现用户地址信息有误,则需要通知 CRM 系统做相应处理;如果发现配线错等线路异常,需要通知资源系统做相应处理。类似的,在测量工单和程控工单施工的过程中,如果处理成功,可以正常回单;如果发现号码错、横列端子错等问题,需要通知资源系统。当自动工单和三种类型的人工工单都成功处理后,表示施工过程结束。

从上述分析可以看出,每个活动的执行过程中都可能发生错误,除了正常回单以外,其他路径都可以归结为异常。

3.2 规则描述

根据对上述流程的分析,总结出在异常发生时,需要执行不同的动作,见表 2。

表 2 异常流程的规则描述

	事件 E	条件 C	动作 A
异常路径 1	自动工单施工	首次失败	自动工单施工
异常路径 2	自动工单施工	连续失败	等待人工提取
异常路径 3	外线施工	用户地址错	通知 CRM 系统
异常路径 4	外线施工	配线坏/重	通知资源系统
异常路径 5	测量台施工	横列端子错	通知资源系统
异常路径 6	程控机房施工	号码重	通知资源系统

其中,条件对应的故障名称需要在 WSDL 文件中

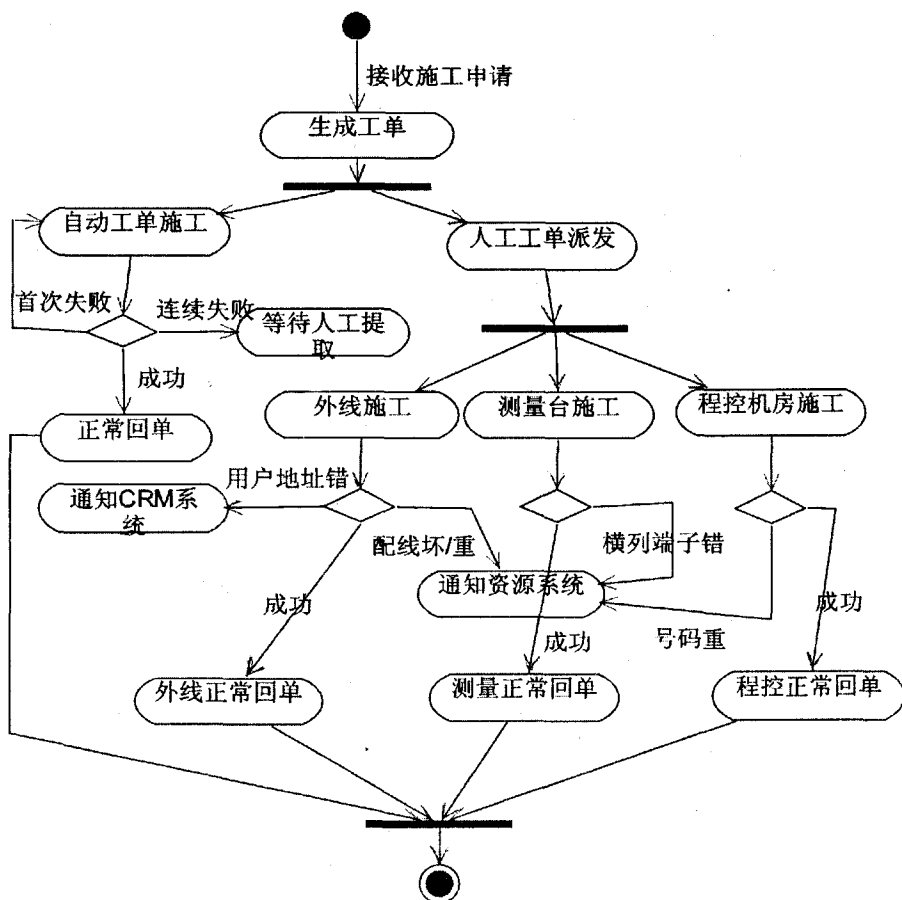


图 1 电信业务施工过程图

定义出来,响应异常时执行的动作需要在 WSDL 文件中以服务的方式进行定义。出于篇幅的考虑,此处没有给出完整的 WSDL 文件,仅给出说明。

故障定义:自动工单施工异常 AutoWorkSheetRunningException、外线施工异常 LineException、测量台施工异常 DeviceException、程控机房施工异常 NumberException。变量名:ExceptionName,其中:元素 ErrorNum 表示失败次数;ErrorCode 表示失败原因。

服务定义:自动工单施工 AutoWorkSheetRunning、通知 CRM 系统 InformCRMSystem、通知资源系统 InformResourceSystem

3.3 BPEL 异常流程定义

在前期工作完成后,进行异常流程的定义:

```
<faultHandlers>
  <catch faultName = "AutoWorkSheetRunningException" faultVariable = "ExceptionName">
    <if>
      <condition> $ExceptionName.ErrorNum = 1 </condition>
      <!--失败次数为1-->
      <invoke name = "AutoWorkSheetRunning" />
    <else>
      <wait><for>'P2'</for></wait><!--等待时间根据业务规定定义-->
    </else>
  </if>
  </catch>
  <catch faultName = "LineException" faultVariable = "ExceptionName">
    <if>
      <condition> $ExceptionName.ErrorCode = 0 </condition>
      <!--0代表用户地址错>
      <invoke name = "InformCRMSystem" />
    <elseif>
      <condition> $ExceptionName.ErrorCode = 1 </condition>
      <!--1代表配线环/重>
      <invoke name = "InformResourceSystem" />
    </elseif>
  </if>
  </catch>
  <catch faultName = "DeviceException" faultVariable = "ExceptionName">
    <if>
      <condition> $ExceptionName.ErrorCode = 2 </condition>
      <!--2代表横列端子错>
      <invoke name = "InformResourceSystem" />
    </if>
  </catch>
  <catch faultName = "NumberException" faultVariable = "ExceptionName">
```

```
<if>
  <condition> $ExceptionName.ErrorCode = 3 </condition>
  <!--3代表号码重>
  <invoke name = "InformResourceSystem" />
</if>
</catch>
</faultHandlers>
```

4 结束语

给出了基于ECA规则的BPEL异常流程定义的方法,并针对电信企业的实际流程案例进行了流程分析,给出了异常流程的分析过程和定义步骤,验证了该方法的可行性。下一步的工作是对异常处理的理论和实践问题^[9~12]做进一步的研究,例如如何对BPEL异常建模进行自动检测。

参考文献:

- [1] Liu H, Li Q, Xiao M J. A declarative approach to enhancing the reliability of BPEL processes[C]// In Proc. IEEE Int. Conf. Web Services. Salt Lake City, USA: IEEE Computer Society, 2007:272-279.
- [2] Claus H, Gustavo A. Exception handling in workflow management systems[J]. IEEE Transactions on Software Engineering, 2000, 26(10):943-958.
- [3] 陈翔,刘军丽. ECA规则在工作流管理系统中的应用[J]. 计算机工程, 2007, 33(13):65-67.
- [4] 黄龙达, 姜渊胜. 基于WF-net的工作流恢复网模型的研究[J]. 计算机应用研究, 2007, 24(12):68-71.
- [5] Zhang Ychuan, Liu Ying, Wang Deshuai, et al. An Exception Analyzing and Handling Approach of Composite Service Based on ECA Rule[C]//World Congress on Software Engineering. Xiamen, China: IEEE Computer Society, 2009:387-391.
- [6] Web Services Business Process Execution Language Version 2.0[EB/OL]. 2007-06-11. <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>.
- [7] 盛步云, 万哲, 丁毓峰. BPMS中一种基于流程异常库的异常处理方法[J]. 计算机技术与发展, 2008, 18(12):84-87.
- [8] 曾慧琼, 李建华, 王美云. 组合服务事务的异常处理机制研究[J]. 计算机工程与设计, 2008, 29(8):1886-1889.
- [9] 张猛, 侯秀萍, 王烨, 等. 流失败恢复的研究[J]. 计算机技术与发展, 2009, 19(4):65-68.
- [10] 申屠晓丽, 姚明海, 汤幸江. 工作流管理系统的柔性技术研究与应用[J]. 计算机技术与发展, 2010, 20(1):120-123.
- [11] 高山. 基于BPEL的工作流系统研究与实现[D]. 成都: 电子科技大学, 2008.
- [12] 田静. 工作流异常处理方法与实现[D]. 南京: 东南大学, 2006.