

# BPMS 中一种基于流程异常库的异常处理方法

盛步云<sup>1,2</sup>, 万哲<sup>1</sup>, 丁毓峰<sup>1</sup>

(1. 武汉理工大学机电学院, 湖北 武汉 430070;

2. 贵州大学 CAD/CIMS 工程技术研究中心, 贵州 贵阳 550003)

**摘要:**针对传统的 BPMS 处理异常时所采用的失败补偿法、ECA 法、案例推理法等方案在智能化方面存在的不足,着重研究了业务流程管理系统中异常处理的问题,提出了一种 ECA 法与案例推理法相结合的、基于流程异常库的处理方式,来提高系统对于异常的预测能力以及在异常发生时能够尽早采取有效的解决措施,减小异常的负面影响,最大程度上实现系统处理异常的自动化及智能化,增加了 BPMS 的柔性。

**关键词:**BPMS 异常;ECA 规则;案例推理法;流程异常库;相似度

**中图分类号:**TP18

**文献标识码:**A

**文章编号:**1673-629X(2008)12-0084-04

## A Method Based on Process Abnormity Set for Resolving Abnormity in BPMS

SHENG Bu-yun<sup>1,2</sup>, WAN Zhe<sup>1</sup>, DING Yu-feng<sup>1</sup>

(1. College of Mechanical Electronic Engineering, Wuhan University of Technology, Wuhan 430070, China;

2. Research Center of CAD/CIMS Engineering, Guizhou University, Guiyang 550003, China)

**Abstract:** In allusion to the weakness of traditional methods BPMS used to deal with abnormity, including compensation of failure, ECA and case based reasoning, researches on the abnormity resolving in BPMS and brings forth a method based on process abnormity set, which combines ECA and CBR. It improves the forecasting capability of BPMS and makes sure to solve abnormity effectively as soon as possible, descend abnormity's negative affection, realize solving abnormity automatically and intelligently and then make BPMS more flexible.

**Key words:** BPM abnormity; ECA rule; case based reasoning; process abnormity set; similar degree

### 0 引言

信息技术的飞速发展和市场全球化趋势的日益加深,使企业的组织形式、生产流程发生巨大变化,对相应的业务流程管理提出了敏捷、实时和动态需求。这些需求归结起来就是对 BPMS 的柔性要求,也就是系统对于变更和异常的处理能力<sup>[1]</sup>。而在今天的分布、异构环境与复杂的企业经营过程下,大部分 BPMS 面对并发操作与操作失败等情况缺乏保证执行正确性与可靠性的能力<sup>[2]</sup>,所具有的事务处理能力远远不能满足企业的要求,这也是业务流程管理系统在实际应用推广中所遇到的主要障碍之一,同时也影响到业务流

程管理系统的可扩展性。

BPMS 对于异常处理的方法,目前主要有失败补偿法、ECA 法、案例推理法等,这些方法能对某些可预见的异常进行非人工干预的处理(系统自动处理),或者对非可预见的异常进行人工干预的处理(系统管理员处理),由于 BPMS 中异常的发生大多不可预知或至少不可完全预知,多数情况下都需要系统管理员的参与才能完成对异常的处理,系统很难实现自动处理异常的机制,而理想的情况应是系统能自我学习,从而减少失败发生的次数并能自动正确处理非可预见性异常,故这三种方法都没有从真正意义上实现系统的智能化。

文中针对以上异常处理方法在智能化方面存在的不足,着重研究了业务流程管理系统中异常处理的问题,提出了一种 ECA 法与案例推理法相结合的、基于流程异常库的处理方式,来提高系统对于异常的预测能力以及在异常发生时能够尽早采取有效的解决措施,最大程度上实现系统处理的自动化及智能化。

收稿日期:2008-03-26

基金项目:国家自然科学基金重大国际合作项目(50620130441);武汉市科技攻关项目(20061002082);武汉市重点科技攻关项目(20055102021)

作者简介:盛步云(1964-),男,湖北武汉人,博士,教授,博导,研究方向为计算机集成制造。

## 1 BPMS 中的异常

### 1.1 异常的定义与特点

任何业务过程在运行过程中,都可能因受到来自系统内、外各种因素的影响而与原来定义的过程发生偏差。BPMS 的异常是指,在流程运行中偏离正常情况并可能导致障碍的流程变化,一般不可预知或不可完全预知,需要人工活动干预<sup>[3]</sup>。

基于不同的角度,异常有多种分类方式。如按异常的产生源,可分为系统异常和任务异常两类;按异常是否可预见可分为可预见性异常与非可预见性异常等。但无论哪种分类,异常均被公认为是流程运行中与原定义的偏移,具有如下特点<sup>[3]</sup>:

(1) 仅仅发生在过程的运行期间;

(2) 异常导致过程不能正常运行,但也有一些异常的负影响可以忽略不计;

(3) 异常在过程定义阶段是不可预见的,不能在模型中对其进行说明;

(4) 所有的异常处理都不同程度地需要人工干预,尤其对于非可预见性异常,人工干预是主要处理手段。

### 1.2 BPMS 中异常的处理

根据异常的特点,BPMS 对异常的处理主要包括以下三个步骤:

异常检测 → 异常报告提交 → 异常处理

异常检测由系统监控模块中的异常检测控件来完成,通过将目标模块的输入输出与期望值进行比较,判断是否产生异常。当发现并确认异常产生后,生成相应的异常事件报告,在系统内部进行传递,由可以处理该异常的责任人,采用相应的策略进行处理。

根据异常对系统流程的影响程度的不同,BPMS 所采取的异常处理策略主要包括:

\* 忽略:当出现异常的活动执行结果不影响整个流程目标时,忽略该异常,直接回到正常运行状态,继续流程运行。

\* 重试:停止异常活动,将该活动重新执行,直到任务正确完成或重试次数超过最大设定值。

\* 回滚:通过相应补偿行为,使实例状态恢复到可执行状态,再重新启动实例运行。

\* 修改模型:对由于模型与实际情况不符造成的异常,要通过修改过程模型来解决问题。这样的修改会影响当前运行的实例,需将其按旧模型继续运行或迁移到新模型,后续启动的新实例则按新模型运行。

\* 组合:对于复杂的异常,结合上述两种或两种以上的策略进行处理。

BPMS 如何正确选择以上这些异常处理的策略,从而快速有效地解决系统所出现的异常情况的能力对

系统的灵活性、动态性和自适应性有着至关重要的影响。

目前常见的异常处理方法,主要包括:

1) 失败补偿法:对 BPM 流程的每个活动都给出一个补偿活动,用于消除已执行活动产生的影响。当一个活动发生异常时,就启动相应的补偿活动消除其影响。补偿法是目前商业软件中处理异常的主流方式。但实施中难以做到准确界定有效恢复范围。

2) ECA(Event - Condition - Action)规则法:将异常处理条件和行为分别用 E、C、A 进行描述。其中 E 指异常事件、C 指异常对应条件、A 是异常修正或补偿活动。ECA 规则一般用于处理可预见异常,事先将异常处理程序写成 ECA 规则,当异常 E 发生并满足条件 C 时,调用规则处理程序 A<sup>[4]</sup>。

3) 案例推理法:将异常处理作为历史案例保存在案例库中;当新异常发生时,将新异常与库中案例进行相似度匹配,找到相似度最大的案例;若二者情况相同,则按案例的解决方案处理新异常,若不完全一样,则以案例解决方案为模板进行调整和修改,获得新异常的解决方案,并把该新异常及解决方案添加到案例库中。案例推理法具有较强的推理能力,适合处理不可预见的异常。

文中在结合 ECA 规则法与案例推理法的原理和特点的基础上,提出了一种基于流程异常库的异常处理方法,能更好地处理可预见性与非可预见性异常,并强调系统的自我学习能力及智能化设计。

## 2 基于流程异常库的异常处理

### 2.1 总体设计思想

在 1.2 节中提到的 ECA 规则法,对那些规则库中规定好的异常给出了明确的处理方案,具有较高的处理效率。但由于该方法本身不具备学习能力,对那些不符合 ECA 规则库中规则特征描述的异常,则只能将其抛出;案例推理法是将各种异常处理的案例保存在一个案例库中,按照异常案例的相似度来寻找可采用的解决措施,并不断地将新的解决案例加入案例库。这两种方法恰好能形成互补,故文中的基于流程异常库的异常处理方法是以前 ECA 规则法为基础,结合案例推理法原理,设计一个流程异常库,对 ECA 规则法进行扩展。利用以往流程的异常处理经验,将那些在 ECA 库中没有处理规则的异常,按照相似度匹配原理,找到与之最相似的案例,结合该异常的具体情况,调整其处理方式形成本次处理方案,并将该异常处理过程作为新规则添加到流程异常库中,使该 BPMS 具有“智能化”处理异常的能力,提高系统对异常处理的

有效性。该异常处理方式的基本流程图如图 1 所示。

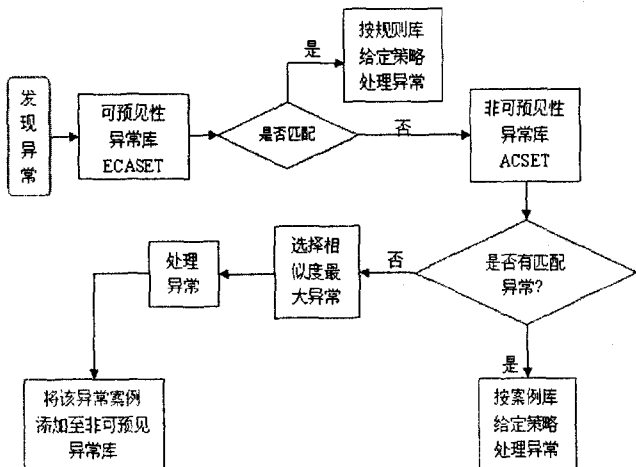


图 1 基于流程异常库的异常处理流程图

## 2.2 详细设计

### 2.2.1 构建流程异常库

流程异常库 (Process Abnormity Set, PASET) 由两部分组成,一部分是基于 ECA 规则的可预见性异常库 (ECASET),另一部分是由异常案例 (Abnormity Case) 组成的非可预见性异常库 (ACSET)。当进行异常匹配时,首先在 ECASET 中搜索,如无符合条件的异常,则转入 ACSET 继续搜索,若均无搜索结果,则在 PASET 中进行相似度的计算以确定最相似的异常案例来作为处理的参考与借鉴。关于相似度的计算会在后文中介绍,现给出流程异常库中相关数据的定义:

$$PASET = ECASET \cup ACSET$$

$$ECASET = \{ECA_1, ECA_2, \dots, ECA_n\}.$$

ECASET 中的每一个 ECA 规则均定义成一个三元组:

$$ECA_i = (Ev, Co, Ac) \quad 1 \leq i \leq n$$

其中,Ev 为异常事件,是一个三元组:  $Ev = (Time, Locate, Type)$ , Time 表示异常发生时间,Locate 表示异常发生位置,可以是活动、子过程或一个系统模块, Type 为枚举型数据,表示该异常事件的类型,在 ECASET 中,先将异常分为系统异常 (System.ab) 和任务异常 (Task.ab),其中任务异常又再分时间异常 (Task.ab.t)、数据异常 (Task.ab.d) 和资源异常 (Task.ab.r) 三种。

Co 是异常处理的前提条件,是一个谓词语句。

Ac 是该 ECA 规则包含的处理措施,它是一个二元组 (Op, Pr),其中,Op 是对该异常所进行的操作,包括 5 种类型 {Ignore, Retry, Rollback, Modify, Comp}, 分别对应 1.2 节中提到的忽略、重试、回滚、修改模型、组合等 5 种异常处理策略;Pr 代表具体的处理过程,可以是一个活动、子函数、应用程序等。

$$ACSET = \{AC_1, AC_2, \dots, AC_n\}.$$

ACSET 中的每一个异常案例均做如下定义:

$$AC_i = (Ev, Sys. Status, Proc. Status, Activity, Participant, Character, Ac)$$

其中,Ev 是异常事件,与 ECASET 中的定义一致,也是一个三元组 (Time, Locate, Type); Sys. Status 和 Proc. Status 分别记录了异常发生时系统和流程的当前状态,包括初始化、激活、运行、暂停、中断、完成等; Activity 表示引发异常的活动; Participant 表示引发异常的活动执行者; Character 为特征数据,它随流程的不同应用领域而变; Ac 同 ECA 三元组中的 Ac,为该案例包含的处理措施。

### 2.2.2 非可预见性异常案例的相似度匹配

如果 BPMS 捕捉到的异常事件  $Ev_i$  无法从 ECASET 和 ACSET 中找到与之完全匹配的案例  $Ev_m$ , 则转而在 PASET 中根据异常案例相似度匹配原则搜索与  $Ev_i$  最相似的案例。BPMS 选择异常的若干主要属性,如异常类型、引起异常的活动、异常发生的时间、异常影响的范围等,根据其在流程中的重要性进行排列,确定出每个属性的概念层次<sup>[5]</sup> (Concept Hierarchy),再分别求出两个案例在各个属性上的相似度值并加权求和,所得的结果即为两案例之间的相似度  $dist(Ev_1, Ev_2)$ 。

概念层次是指概念域中从一般到特殊的层次结构。如在制造企业组织中,可以将其组织的层次从大到小,从一般到特殊依次划分为如图 2 所示情况。再根据此层次结构给每个概念层次赋予一个层次值 (level value),通常将末级节点的层次值 (图 2 中的叶节点) 为 0,由下向上层次值逐渐加大,如图中各概念旁所标注的数值。

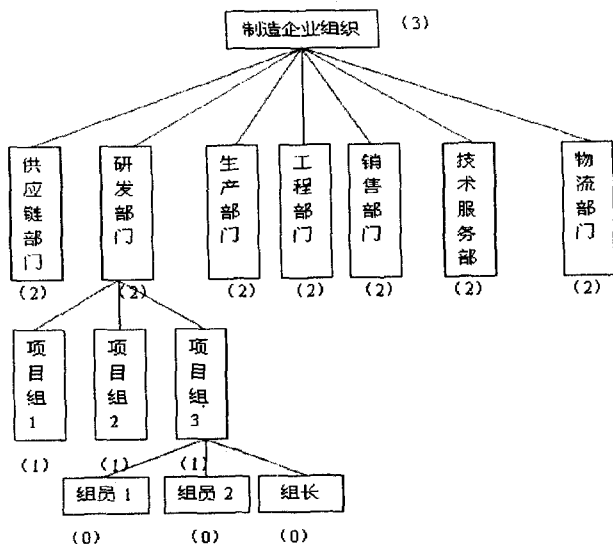


图 2 制造企业概念层次结构图

设 BPMS 捕捉到的异常事件为  $Ev_1, Ev_2$  为 PASET 中一个异常案例,  $a_j$  为异常的某个属性(异常类型、引起异常的活动、异常发生的时间、异常影响的范围等), 则  $a_{j_1}$  和  $a_{j_2}$  分别对应  $Ev_1$  和  $Ev_2$  的  $a_j$  属性。定义属性相似度为  $\text{dist}(a_{j_1}, a_{j_2})$ , 则:

$$\text{dist}(a_{j_1}, a_{j_2}) = |a_{j_1} - a_{j_2}|$$

当  $a_{j_1}$  和  $a_{j_2}$  都是数值型时, 其相似值就等于二者相减的绝对值; 如不是数值型, 则以  $a_{j_1}$  和  $a_{j_2}$  对应概念层次中的最小相同节点层次水平为相似度。

求得案例之间各属性的相似度后, 将其按规定因子加权求和, 即可得到异常事件  $Ev_1$  与 PASET 中当前案例  $Ev_2$  之间的相似度值  $\text{dist}(Ev_1, Ev_2)$ , 即

$$\text{dist}(Ev_1, Ev_2) = \sum_{j=1}^m w_j \cdot \text{dist}(a_{j_1}, a_{j_2})$$

其中,  $w_j$  为选择属性  $a_j$  的权重。由公式可知, 相似度越小, 两个案例之间越相似, 其处理方案对当前异常事件处理的参考借鉴价值也就越大。

### 2.3 基于流程异常库的 BPMS 异常处理详细步骤

根据上述基本定义, 所设计的基于流程异常库的 BPMS 中异常处理的详细步骤描述如下:

Step1 当异常发生时, 系统自动捕捉异常事件  $Ev_1$ , 首先在可预见性异常库 ECASET 中选择一条 ECA 规则  $ECA_i$ ;

Step2 判断  $Ev_1$  和  $ECA_i$  在时间 Time、地点 Locate 和类型 Type 上是否一致。

(1) 当一致时。

a. 在匹配项  $ECA_i$  中继续判断  $ECA_i$  . Co 是否满足, 若不满足, 则继续判断直至条件满足。

b. 选择  $ECA_i$  . Ac. Op 中的相应处理策略(包括忽略 Ignore、重试 Retry、回滚 Rollback、修改模型 Modify 和组合 Comp 等 5 种处理操作, 后 3 种操作需要调用  $ECA_i$  . Ac. Pr 进行具体处理)。

(2) 当不一致时。

a. 继续在 ECASET 中选择另一条规则, 重复 Step2。

b. 当遍历完所有 ECA 规则仍未找到匹配项时, 则认为该异常事件  $Ev_1$  为非可预见性异常, 转入非可预见性异常库 ACSET 中搜索现有异常案例。

c. 在 ACSET 中选择一个异常案例  $AC_i$ , 判断  $Ev_1$  与  $AC_i$  在时间 Time、地点 Locate 和类型 Type 上是否一致。

1) 当一致时。

i. 在匹配项  $AC_i$  中继续判断  $AC_i$  . Sys. Status、 $AC_i$  . Proc. Status、 $AC_i$  . Activity、 $AC_i$  . Participant 和  $AC_i$  . Character 是否满足, 如果不满足, 则继续判断直至条

件满足。

ii. 选择  $AC_i$  . Ac. Op 中的相应处理策略(包括忽略 Ignore、重试 Retry、回滚 Rollback、修改模型 Modify 和组合 Comp 等 5 种处理操作, 后 3 种操作需要调用  $AC_i$  . Ac. Pr 进行具体处理)。

2) 当不一致时。

i. 继续在 ACSET 中选择另一个异常案例, 重复 Step2。

ii. 当遍历完所有 AC 案例仍未找到匹配项时, 则认为异常事件  $Ev_1$  不能按现有 ECA 规则及 AC 案例处理, 需报告系统管理员, 同时启动相似度匹配搜索, 查找相似异常。

iii. 从 PASET 中选择一条案例  $Ev_2$ , 依次分别计算  $Ev_2$  与当前异常  $Ev_1$  之间在各选择属性的相似度值  $\text{dist}(a_{j_1}, a_{j_2})$ 。

iv. 完成所有属性相似度计算后, 将其加权求和, 得到两案例之间的相似度  $\text{dist}(Ev_1, Ev_2)$  并将值赋给变量  $t$ , 用 Case 记录所选案例。

v. 判断 TASET 是否搜索完毕, 如果没有, 继续选择另一条案例, 重复步骤 iii 和 iv, 计算相似度。如果新求得值小于  $t$ , 将该值赋给  $t$ , 并用当前案例覆盖 Case 中的案例。如果已经完毕, 则 Case 中的案例即为所求与当前异常  $Ev_1$  最相似的案例。

vi. 将该案例的基本信息和相关信息提交系统管理员, 由管理员根据情况做出方案判断和调整, 采用相关措施进行异常处理。

vii. 上述异常处理完毕后, 将其相关处理内容及基本信息, 按前述 AC 案例构成形式, 形成新的 AC 案例, 加入 ACSET 库。

### 3 结束语

综上所述, 基于流程异常库的 BPMS 异常处理机制是: 对于可预见性异常, 根据 ECA 规则库中已有的处理策略和方式, 自动执行相应的处理操作; 对于非可预见性的异常, 则转入异常案例库 ACSET 搜索相同异常案例并由系统根据异常案例采用相应处理策略; 若均无匹配的异常, 则按相似度最小的定则在库中找到与本次异常最相似的案例提交给系统管理员, 再根据实际情况对案例进行相应调整后, 将其作为本次异常的处理方案, 并在处理完毕后, 将这一过程写成新的 AC 案例, 以备后续使用。

由于实际应用中大部分异常都具有相似性, 又非完全相同, 故这种 ECA 规则法与案例推理法相结合的处理机制, 能使系统的异常处理能力得到较大的提升,

(下转第 91 页)

法和基于样本的方法。表 1 给出了几种方法处理结构子图的运算时间。由表可见,使用 BSCB 模型和基于样本的方法需要几分钟甚至几十分钟才能完成的修复,文中方法几秒内就能完成,在运算时间上的优势是显而易见的。

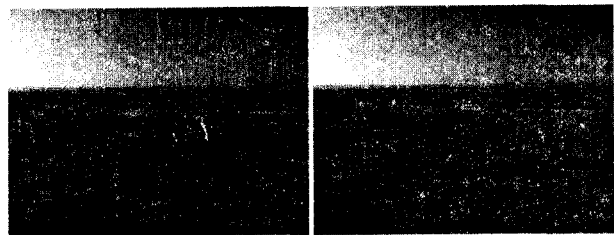


图 5 目标移除效果



图 6 文字去除效果

表 1 对结构图像修复的几种方法时间上的比较

实验图像	图像大小	破损区大小	修复时间		
			BSCB 模型	基于样本方法	文中方法
图 4	256×256	4080(6.23%)	3.0min	3.2min	0.66s
图 5	302×399×3	3595(3%)	38min	8.1min	4.84s
图 6	279×438×3	20737(15.94%)	21min	76min	5.43s

5 结束语

目前对纹理图像的修复主要采用基于样本的方法,对结构图像的修复采用基于 PDE 的方法。文献 [10]的方法把图像分解为结构子图和纹理子图;然后分别用不同的方法处理分解后的子图;最后合成来完成修复。对其中的结构子图的修复部分作了些改进:根据结构子图具有高度平滑、边缘信息不显著等特征,使用一种新的修复方法。该方法仅利用邻域信息填充

破损区,取得较好的效果,且大大缩短了修复时间。然而,该方法也具有一定的局限性,对破损区较小(如划痕、文字等)或边缘不显著的图像具有较好的修复效果;但对于有明显边界的大区域破损图像,修复效果不太理想。所以,今后可以在这方面做些改进工作,使其能有更大的适用范围。

参考文献:

[1] Bertalmio M, Sapiro G, Caselles V, et al. Image inpainting [C]//In Proc. ACM Conf. Comp. Graphics (SIGGRAPH 2000). New Orleans, LU:[s. n.],2000,417-424.

[2] Chan T,Shen J. Mathematical models for local non-texture inpaintings[J]. SIAM J App. Math., 2001, 62(3): 1019-1043.

[3] Rudin L,Osher S,Fatemi E. Nonlinear total variation based noise removal algorithms[J]. Physica D, 1992, 60(1-4): 259-268.

[4] Chan T F,Shen J H. Non texture inpainting by curvature driven diffusion (CDD) [D]. Los Angeles: Univ. of California, 2000.

[5] Chan T F,Kang S H,Shen J. Euler's Elastica and Curvature based inpaintings[J]. SIAM Journal on Applod Mathematics, 2002,63:564-592.

[6] Esedglu S,Shen J. Digital inpainting bassed on the Mumford-Shah-Euler image model [J]. European J. Appl. Math., 2002,13:353-370.

[7] Criminisi A,Perez P,Toyama K. Region filling and object removal by exemplar-based image inpainting[J]. IEEE Trans. Image Processing,2004,13:1200-1212.

[8] Bertalmio M, Vese L, Sapiro G, et al. Simultaneous structure and texture image inpainting[J]. IEEE Trans. Image Process, 2003,12(8):882-889.

[9] Meyer Y. Oscillating patterns in image processing and nonlinear evolution equations[M]. Priston: American Mathematical Society Press,2002.

[10] Vese L A, Osher S J. Modeling textures with total variation minimization and oscillating patterns in image processing[J]. Journal of Sci Comput,2003,19(3):553-572.

(上接第 87 页)

新案例的不断写入,也使系统有一定的学习能力,从而实现了一定的智能化。

参考文献:

[1] Harmon P. IBM's BPM Strategy, Products and Architecture [J]. Business Process Trends,2004,2(11):23-25.

[2] 罗海滨,范玉顺,吴澄. workflow数据的一致性保护框架 [J]. 计算机集成制造系统,2002,8(4):320-325.

[3] Li Weiping,Xue Jinsong,Zhu Yunlong. The Dynamic Modeling Method of Workflow Management System Supporting BPR[J]. China Mechanical Engineering, 2002, 13(15):1314-1317.

[4] 孙瑞志,史美林. workflow异常处理的形式描述[J]. 计算机研究与发展,2003,40(3):393-397.

[5] 陶亚雄,王坚. 基于流程知识的BPM系统异常处理研究 [J]. 计算机工程,2007,33(9):186-188.