基于 Petri 网的 RFID 中间件中复合事件检测研究

叶 蔚[1,3], 黄 雨[2,3], 赵 文[2,3], 张世琨[2,3], 王立福[2,3]
(1. 北京大学信息科学技术学院, 北京 100871; 2. 北京大学软件工程国家工程研究中心, 北京 100871;
3. 北京大学信息科学技术学院软件研究所高可信软件技术教育部重点实验室, 北京 100871)

摘 要: 探讨了一种利用复杂事件处理技术处理 RFID 高层业务逻辑的机制. 通过用复合事件来表达 RFID 应用系统中常见的高层业务逻辑, 将对业务逻辑的处理转化为 RFID 中间件对复合事件的检测. 从 RFID 中间件检测行为的角度探讨了在这一转化过程中事件定义和事件检测的若干关键问题. 基于有色网定义了 RFID 事件流检测网系统作为 RFID 事件检测模型的描述工具. 给出了事件检测模型的构造规则. 根据检测模型的网结构特征对具有复杂层次的复合事件的可检测性进行了分析.

关键词: 无线射频识别; 无线射频识别中间件; 复杂事件处理; 复合事件检测; 有色网
中图分类号: TP311 文献标识码: A 文章编号: 0372-2112 (2008) 12A-001-08

Research on Composite Event Detection in RFID Middleware Based on Colored Petri Net

YE Wei[1,3], HUANG Yu[2,3], ZHAO Wen[2,3], ZHANG Shi-kun[2,3], WANG Li-fu[2,3]
(1. *School of Electronics Engineering and Computer Science, Peking University, Beijing* 100871, *China*;
2. *National Engineering Research Center for Software Engineering, Peking University, Beijing* 100871, *China*;
3. *Key Laboratory of High Confidence Software Technologies (Ministry of Education), School of Electronics Engineering and Computer Science, Peking University, Beijing* 100871, *China*)

Abstract: This paper presents a mechanism of using complex event processing technology to process high level business logic of RFID application. By devising common high level business logic into hierarchal composite event, we automate business logic processing as event detection in RFID middleware. We first address several key issues in event definition and event detection from the perspective of RFID middleware's detection behavior. Then based on colored Petri net, we define Event Flow Detection net System as a formal modeling tool for RFID event detection behavior. We propose constructing rules for event detection model and analyze detectability of hierarchical composite event based on characteristics of the corresponding net structure.

Key words: radio frequency identification(RFID); RFID Middleware; complex event processing; composite event detection; colored petri net

1 引言

随着无线射频识别(RFID)技术的逐渐成熟与发展, RFID 正越来越广泛地应用在供应链、交通、医疗、国防等各个行业. RFID 是一种利用射频方式进行非接触式双向通信的自动识别技术, 与传统的条形码相比, 具有快速、实时、可重复使用、穿透性强、环境适应性强、数据容量大等诸多优点. 更重要的是, RFID 技术给我们带来了"物联网"的概念, 使其不仅仅局限于工具型的应用, 而形成更广泛的价值链体系.

RFID 系统包括四个部分: 标签, 读写器, 中间件和应用程序. RFID 中间件通常也叫做 RFID 边缘服务器, 作为连接硬件系统和企业应用的基础设施, 被称为是 RFID 应用的神经中枢. 当前大多数 RFID 中间件平台, 如 BEA 公司的 Weblogic RFID Edge Server 和 Sun 公司的 Java System RFID Software, 以及开源领域的 Singularity 和 Logicalloy 等, 着重关注低层的数据过滤与搜集. 从某种意义上说, 这些中间件平台是 ALE[1] 标准的一种具体实现.

ALE 是 EPC Global 组织提出的 RFID 中间件标准.

收稿日期: 2008-07-05; 修回日期: 2008-10-10
基金项目: 国家高技术研究发展计划(No. 2006AA04A119, No. 2006AA04A121); 国家自然科学基金(No. 60803014)

ALE 关注低层 EPC[2] (Electronic Product Code) 数据的过滤和搜集, 不增加 EPC 数据的语义. 在 ALE 的上层, EPC Global 提出了一套关于 RFID 高层业务信息的标准 EPCIS[3], 其中定义了四种类型的 RFID 业务事件 (IS Event). 我们发现在 EPC Global 最新的协议栈中, ALE 和 EPCIS 之间新定义了一个 IS Capture Application 模块, 在特定的业务过程中, 这个模块与将底层 EPC 数据与其他信息源相关联, 从而为 EPC 数据提供业务上下文. 为了更快速地响应 RFID 应用需求的变化, 以及进一步简化第三方 RFID 应用程序的开发, RFID 中间件应为 IS Capture Application 模块或者具有类似功能的模块提供更大的支持. 这就要求 RFID 中间件关注更高层的信息处理, 为 EPC 数据转化为业务信息提供更加智能的自动化定制功能.

从原始 EPC 数据到 EPC 业务信息的转化关键在于提取 EPC 数据的隐含语义. 复杂事件处理 (Complex Event Processing, CEP) 是一种可以从任何分布式的基于消息的系统中提取和分析信息的技术[4], 适合于 RFID 应用的高层业务信息处理. 目前已经有一些 RFID 复杂事件处理方面的工作, 主要研究适合 RFID 应用特征的事件定义语言和事件检测算法. 但是这些工作更多关注于原子事件和相对简单的复合事件, 其重点仍然在低层数据处理. 为了处理日益复杂的 RFID 应用逻辑, 需要对具有复杂层次的 RFID 复合事件提供更大程度的支持.

表达 RFID 业务逻辑的高层事件通常具有多层次的内部结构, 如果使用事件操作符进行任意的事件复合, 常常导致事件描述本身的不一致性和不可检测性. 因此本文首先从中间件事件检测行为的角度, 探讨复合事件定义的若干问题, 从而为复合事件定义提供一定程度的指导. 事件定义和事件检测是复杂事件处理的两个关键问题, 然而这两者之间却有一道天然的鸿沟. 到现在为止事件处理被关注的更多的是处理效率问题, 而事件检测正确性验证的问题至今没有相关研究. 作为对该问题研究的初步尝试, 本文为复合事件建立了形式化的可执行检测模型. 检测模型是事件检测正确性验证的一个重要基础, 同时也是 RFID 中间件实现的直接指导. 我们在有色网的基础上进行扩展, 提出了 RFID 事件流检测网系统, 同时以事件操作符为基础给出了网的基本构造块, 提出了层次网结构的构造规则. 针对含有否定事件操作符的一类复合事件, 我们通过定义时间事件来表示系统的查询计划, 根据事件表达式特征给出了时间事件的产生规则, 并对不可检测事件的网结构特征进行了简要分析.

本文的工作基于 RFID 中间件原型系统 PKU RFID Edge Server (PRES). PRES 是兼容 EPC Global 标准的纯 JAVA RFID 中间件, 采用复杂事件处理技术来提取 EPC 数据的业务信息并产生高层业务事件.

2 相关研究

RFID 复杂事件处理已有不少研究, 但是更多关注低层数据过滤和搜集. 文献[8] 提出一种基于查询计划的事件处理方法, 同时提供一组性能优化策略, 但是没有关注复杂层次的复合事件检测, 事件操作符在复合事件表达式中只允许使用一次, 限制了表达复杂业务逻辑的能力. 文献[9] 给出了一种基于规则的 RFID 事件处理机制, 提供一种基于图的事件检测算法, 但是只关注 EPC 事件, 而且对多层次的复合事件定义缺乏指导. 文献[10] 是基于 Petri 网来进行事件检测的相关工作, 但是只给出了网定义, 没有给出事件网的构造规则, 而且基于事件表达式所产生的网结构比较复杂. Esper[11] 是一个开源的复杂事件处理引擎, 用类 SQL 语言来表达复合事件, 使用有穷状态自动机进行复合事件检测, PRES 的复杂事件处理框架参考了 Esper 的架构.

3 RFID 事件模型

定义 **1** RFID 事件是在 RFID 系统中一个有意义的发生的规约.

从事件的构造层次角度来看, RFID 事件分为原子事件和复合事件两种类型. 在上下文清楚的语境下, 我们简称事件类型为事件, 事件类型的具体实例则称为事件实例. RFID 中间件的一个主要功能就是根据应用需求, 将系统中原始的原子事件流转化为有语义的复合事件流. RFID 事件具有以下讨论的几种形式之一.

3.1 原子事件

定义 **2** 原子事件是瞬时发生, 即发生时间为一个时间点的事件.

大多数 RFID 事件处理关注相对低层的事件处理, 系统只考虑 EPC 标签读取事件. 在当前的某些 RFID 应用中, 过滤搜集层的事件和 EPCIS 层的事件比较类似, 这种情况下往往不需要复杂的处理. 但是随着 RFID 应用的发展, 业务逻辑变得更加复杂, EPC 的隐含语义也趋于丰富. 语义一方面来自系统中存储的与 EPC 关联的业务信息, 另外一个重要方面则来自与其他系统的交互. 这就要求 RFID 中间件要关注更多类型的原子事件, 为此我们将 RFID 原子事件分为以下三类: EPC 事件, 交互事件和时间事件.

EPC 事件

定义 **3** 当 RFID 标签通过 RFID 读写器读写范围时, 产生的 EPC 读取事件称为 EPC 事件.

EPC 事件是流入 RFID 中间件的原子事件流的主要组成部分. 对于 EPC 事件我们做出如下假设: (1) RFID

标签中只有 EPC 数据; (2) 读取 RFID 标签的读写器可以是一个实际的物理读写器, 也可以是一个包含了一个或者多个物理读写器的逻辑读写器; (3) EPC 事件的时间点是离散的, 时间点与 RFID 中间件的时钟同步, 这一假设也针对其他所有原子事件. 基于以上假设, 一个 EPC 事件表示为(epc, reader, timestamp), 其中 epc 是被读取标签的 EPC 码, reader 是进行读取操作的读写器标识, timestamp 是读取发生的时间点. 举例来说, 在 2006-09-20T06: 36: 17 这一时间点 reader1 扫描到标签 urn: epc: id: sgtin: 11852001. 12345. 000000000019 这一 EPC 事件实例, 可以用(urn: epc: id: sgtin: 11852001. 12345. 000000000019, reader1, 2006-09-20 T06: 36: 17) 来表示.

交互事件

定义 4　交互事件是外部业务系统传入 RFID 中间件的信号.

交互事件的发生时间可能是一个时间间隔, 而不是一个时间点. 但是通常我们只关注交互事件发生的结束, 因此我们假设交互事件的结束时间点为它的时间点, 仍然认为交互事件是原子事件. 交互事件通常带有业务信息, 以事件属性的形式表示. 交互事件表示为 $(\{p_1, p_2, \cdots, p_n\}, \text{timestamp})$, 其中 $\{p_1, p_2, \cdots, p_n\}$ 代表该交互事件的业务相关的属性集合, timestamp 是交互事件发生的时间点. 作为交互事件的一个例子, 我们考虑一个供应链中场景: 在供应链的业务过程中, 拖盘需要经常装载贴有 RFID 标签的货物, 每一次扫描到货物, 可能是货物加入托盘, 离开托盘或者只是被扫描, 这个信息涉及到货物实时追踪中的位置. 我们可以定义交互事件({action, location}, timestamp) 来表示外部系统请求扫描这一消息, 其中 action 表示对货物进行的操作, location 表示操作地点, timestamp 表示请求发生时间. 例如在进行装货操作之前, 在应用系统中点击一个"开始装货"的按钮, 这一事件经过 RFID 中间件的适配器后, 成为一个交互事件实例({action= add, location= warehouse1}, 2008-01-20 T08: 37: 27), RFID 中间件将该事件实例和之后发生的 EPC 事件实例进行关联, 生成高层的业务逻辑事件.

时间事件

定义 5　时间事件是表示 RFID 中间件中一段时间推移或者周期性时间推移的事件.

时间事件是由 RFID 中间件自身产生的一种特殊的原子事件, 用来表明到达某一时间点, 而在该时间点通常中间件需要产生某些动作. 用户自己可以定义时间事件, 并与其他事件进行组合. 中间件自身在检测过程中也可能为了检测需要而产生时间事件. 时间事件表示为 $(t, offset, n)$, 表示 $t + offset \cdot i\ (i = 0, \cdots, n)$ 时间点到达, 其中 t 是基准时间点, $offset$ 是推移时间间隔, n 是推移次数.

3.2　复合事件

定义 6　复合事件是一种聚合事件, 通过使用一个事件操作符集合对原子事件和其他复合事件进行复合而产生[5].

我们把在一个复合事件定义中那些被用来进行复合的事件称为复合事件的子事件, 把用来定义一个复合事件的由事件操作符的表示符号和子事件的表示符号组成的表达式称为事件表达式. 利用复合事件可以表达 RFID 应用系统中常见的高层业务逻辑, 将其用复合事件中的子事件以及子事件之间的关系来表达, 而子事件之间的关系则是由事件操作符所规约. 这样, 实现业务逻辑的过程也就可以相应地自动化为中间件进行事件检测的过程.

复合事件时间

RFID 中间件中的复合事件, 从某种意义上说和主动数据库中的复合事件[7~10]类似, 主要关注于事件检测. 在主动数据库领域, 复合事件的时间通常是当作一个时间点, 而非时间区间, 其原因就是因为更关注事件检测的时间点, 而非整个复合事件发生的过程. 文献[6]指出, 把复合事件的时间当作时间点会带来一系列的逻辑问题. 此外, 复合事件内部子事件之间的时间关系在 RFID 业务逻辑中至关重要, 把复合事件的时间当作时间点会降低复合事件表达业务逻辑的能力. 因此, 本文中复合事件的时间定义为一个时间区间, 而原子事件的时间点可以认为是长度为 0 的特殊时间区间. 对于一个任何事件实例 e, 我们用 $[\text{begin}_e, \text{end}_e]$ 表示其时间区间.

事件操作符

事件操作符表达事件之间的关系, 体现在时间和逻辑方面的约束上. 在事件处理的相关研究领域, 没有一个标准语言用以描述所有可能的复合事件, 所以不同的系统往往根据自身的特点采用不同的事件操作符集合. 从供应链、零售、监控等常见的 RFID 应用场景出发, 结合考虑 EPCIS 中的四种业务事件, 我们提出 5 个事件操作符, 包括 Negation、Conjunction、Sequence、Disjunction 和 Within. 他们的运算优先级依次递减. 用 A 和 B 表示事件, a 和 b 相应的表示其事件实例, 5 个操作符的定义如下:

(1) Negation: 记为 $\neg A$, 表示 A 没有发生, $\neg A$ 的事件实例表示为 $(\neg a)$;

(2) Conjunction: 记为 $A\&B$, 表示 A 发生而且 B 发生, $A\&B$ 的事件实例表示为 $(a\&b)$;

(3) Sequence: 记为 $A \overset{\rightarrow}{} B$, 表示 A 和 B 先后发生, 且 $end_a \leqslant begin_b$, $A \overset{\rightarrow}{} B$ 的事件实例表示为$(a \overset{\rightarrow}{} b)$; 对表达式 $A \overset{\rightarrow}{} A \cdots \overset{\rightarrow}{} A$, 我们记为 A^n, 其中 n 是 A 在该表达式中出现的次数, A^n 的事件实例表示为(a^n);

(4) Disjunction: 记为 $A \mid B$, 表示 A 发生或者 B 发生, $A \mid B$ 的事件实例表示为$(a \mid b)$;

(5) Within: 记为 $w(A, n)$, 表示 A 发生, 且 $end_a - begin_a \leqslant n$, $w(A, n)$的事件实例表示为 $w(a, n)$.

事件操作符变元

复合事件的子事件之间除了时间和逻辑方面的约束, 还存在其他的属性相关约束, 这种约束以加在事件表达式后面的布尔表达式存在. 例如 $A\&B(\text{getProperty}X(A) == B.\text{property}X)$ 中的事件操作符就是 Conjunction 的一个变元, 其中 $\text{getProperty}X(A)$ 可以认为是一个系统中的全局函数, 通过查询系统元数据得到 A 事件实例所关联的 X 属性值, X 同时是 B 事件实例中的一个属性. 这个 Conjunction 变元不仅要求 A 发生且 B 发生, 还会将事件实例绑定到后面的布尔表达式中, 只有后面的布尔表达式成真才表示复合事件发生.

事件操作符对其所支配的事件实例的使用可以有多种策略. 实例使用策略不同的事件操作符所定义的事件关系是不同的. 关于事件实例的使用, 文献[7]提出了四种策略, 本文假定所有事件实例以 Chronicle 策略使用, 即对于任何检测过程中产生的事件实例, 相应的事件操作符会把该实例与它支配的其他事件的实例进行复合, 而选取其他实例的原则是符合约束条件且发生时间最早. 例如, 对于 $A\&B$ 这个复合事件, 用 a_i 表示 A 的第 i 个实例, b_i 表示 B 的第 i 个实例, 则对于$\{a_1, a_2, b_1, b_2\}$ 这样一个事件实例序列, 将产生$(a_1\&b_1)$和$(a_2\&b_2)$两个 $A\&B$ 复合事件的实例.

被动事件

基于事件操作符进行任意组合的多层次复合事件可能是不可检测的, 为了说明这一问题, 我们首先给出被动事件的定义.

定义 **7** 含有 Negation 事件操作符的事件表达式所描述的事件, 称为被动事件.

对被动事件的检测, RFID 中间件不仅仅需要等待事件实例通知, 还需要自身对当前系统做出某些查询, 因此, 就需要进一步给出 RFID 中间件进行系统查询的行为规约, 否则 RFID 中间件就无法识别这种事件的实例. RFID 中间件进行系统查询的行为规约并非显式给出的, 而是体现在 Negation 与其他事件操作符结合使用的规则上. 如果无法从被动事件的事件表达式中解析出进行系统查询的行为规约, 则称这类被动事件为不完整事件.

定义 **8** 不完整事件为:

(1) 如果 A 是原子事件, $\neg A$ 是不完整事件;

(2) 如果 A 和 B 是不完整事件, $(A\&B)$和$(A \overset{\rightarrow}{} B)$是不完整事件;

(3) 如果 A 是不完整事件, 对于任意事件 B, $(A \mid B)$和$(B \overset{\rightarrow}{} A)$是不完整事件.

不完整事件是一种被动事件, 其事件定义中没有足够的信息来指导 RFID 中间件的查询行为. 例如对 $\neg A$ 事件, 中间件需要去查询系统中是否没有 A 的实例产生, 必须告诉 RFID 中间件在符合某些条件的时间点去查询, 中间件才能识别出事件实例. 所以不完整事件是不可检测的, 在事件定义的过程中, 应该避免产生不完整事件. 指导 RFID 进行系统查询的信息可以来自 Within, Conjunction 和 Sequence 操作符, 即 Negation 操作符需要与这三个操作符以一定的规则一起使用. 检测过程中查询动作可以由时间事件来触发, 我们将在 RFID 事件检测模型一节中探讨基于事件表达式特征的时间事件产生规则.

复合事件示例

本节给出两个 RFID 应用场景中常见的复合事件示例.

供应链中追踪货物的实时位置需要对货品装卸操作进行记录. 为了表达"一批货品被装载到某个托盘"这一业务逻辑, 可以定义复合事件 $\text{AddAction} \overset{\rightarrow}{} (\text{Item}^n \&\text{Pallet})$, 其中 AddAction 是一个交互事件类型, 该事件发生表明即将装货, Item 和 Pallet 是两个 EPC 事件类型, 分别表示 RFID 读写器发现货品和托盘标签, 一个复合事件的实例被检测到则说明 n 个货品被装载到托盘上.

另外一个例子是 PRES 被用于某军区枪支管理的场景. 为了表达预防枪支被盗这一业务逻辑, 可以定义复合事件 $w(\text{Gun}\& \neg \text{Solider}, 2)$, 其中 Gun 和 Soldier 是 EPC 事件类型, 分别表示在枪支仓库的出口处发现枪支和士兵, 复合事件发生, 说明在枪支被拿出且并非由军区战士拿出, 应该报警.

复合事件表达了业务逻辑, 而在系统中执行业务逻辑则需要进行复合事件检测. 基于 RFID 事件模型, 下面我们将讨论 RFID 事件检测模型.

4 RFID 事件检测模型

事件定义描述了复合事件类型, 而事件检测是基于系统运行时的事件实例对复合事件类型进行匹配的过程. 事件检测模型则是对这一匹配过程的描述. 这一过程涉及到原子事件实例的关联, 复合事件属性值的计算, 无用事件实例的删除等操作.

形式化的事件检测模型是事件检测正确性验证的一个重要基础, 同时也是 RFID 中间件实现的直接指

导. Petri 网图形化和可执行的特点, 使其可以非常直观的用于指导中间件的实现. 利用 Petri 网并发性, 可以处理多个并发的事件实例. Petri 网的层次性也使得其表示复杂层次的复合事件具有天然的优势, 用库所表示事件, 变迁表示事件操作符, 可以很直观的反应复合事件的复杂层次. 用传统的有色网表达事件的时间关系以及事件使用策略等方面会比较复杂, 因此我们对有色网进行了扩展, 提出了事件流检测网系统, 作为事件检测模型的描述工具. 基于事件流检测网系统所构造出来的网结构是对复合事件类型的匹配过程的形式化描述, 即 RFID 事件检测模型.

4.1　事件流检测网系统

定义 9　事件检测网 $ED\text{-net}=(P, T; A, U, C, N, E, G, B, \text{Input}, \text{Output})$ 是 11 元组, 满足以下条件:

(1) P, T 分别表示库所和变迁, C 表示颜色集合, 描述事件类型, 可以是复合事件类型或者以上描述的三种原子事件类型;

(2) $A=NA\cup IA$, A 表示弧集合;

(3) $U: P\rightarrow N^+$, N^+ 表示正整数, 描述库所中的托肯失效时间;

(4) $NA\subseteq P\times T\cup T\times P$, 表示了一般的弧; 一般弧连接的库所和变迁其前集和后集分别定义为: $\bullet p=\{t\mid\langle t,p\rangle\in NA\}$, $p\bullet=\{t\mid\langle p,t\rangle\in NA\}$, $\bullet t=\{p\mid\langle p,t\rangle\in NA\}$, $t\bullet=\{p\mid\langle t,p\rangle\in NA\}$;

(5) $IA\subseteq P\times T$, 表示了约束弧; 约束弧连接的变迁其前集定义为: $^\circ t=\{p\mid\langle p,t\rangle\in IA\}$; 约束弧连接的库所其后集定义为 $p^\circ=\{t\mid\langle t,p\rangle\in IA\}$;

(6) $N: P\rightarrow C$, N 描述了库所颜色;

(7) $E: A\rightarrow N^+\times C$, E 描述了弧上消耗或者产生的托肯数和托肯颜色;

(8) $\text{Input}\subseteq P$, $\text{Output}\subseteq P$, 且满足 $\text{Input}\neq\varnothing\wedge\text{Output}\neq\varnothing\wedge\forall p\in\text{Input}: \bullet p=\varnothing\wedge\forall p\in\text{Output}: (p\bullet=\varnothing\wedge p^\circ=\varnothing)$; Input 和 Output 为网系统的输入库所和输出库所, 其颜色分别表示原子事件类型和复合事件类型.

(9) G 表示哨函数, 描述了事件的条件约束, 是与输入库所颜色相关的布尔表达式;

(10) B 表示变迁操作表达式, 描述了如何计算输出托肯的属性值, 这里假设我们已知表达式的语法.

定义 10　事件检测网系统 $ED\text{-system}=(P, T; A, U, C, N, E, G, B, \text{Input}, \text{Output}, M)$ 是 12 元组, 其中 $(P, T; A, U, C, N, E, G, B, \text{Input}, \text{Output})$ 是事件检测网, M 是事件检测网的标识. M 为 $P\rightarrow$ TOKEN 的函数, 在网系统中, 托肯表示事件实例, 它是具有某种颜色即具有一定属性值的对象, 设 TOKEN 表示所有托肯对象集合.

定义 11　事件流 σ 表示原子事件实例序列, 表示为 $\sigma=E_1E_2E_3\cdots$, 其中 E_i 是 TOKEN 的子集, E_n 表示第 n 个关注的时间点发生的原子事件实例集合, σ_n 表示当前从第 n 个关注的时间点开始的事件实例序列. 事件流中的事件实例包括上文中定义的三种原子事件实例, 即 EPC 事件实例, 交互事件实例和时间事件实例.

定义 12　事件流检测网系统由事件流 σ 与检测网系统 EFD-*System* 融合而成, 它是如下 14 元组 $EFD\text{-}System=(P', T'; A', U', C, N', E', G, B, Input, Output, M, p_0, \sigma)$, 其中 p_0 表示自产生托肯的库所, 它按照事件流 σ 产生表示事件实例的托肯, 融合后的网结构如图 1 所示:

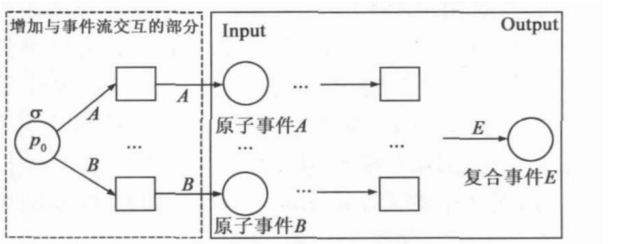


图1　事件流检测网系统

定义 13　事件流检测网系统触发规则

设事件流检测网系统 $EFD\text{-System}=(P', T'; A', U', C, N', E', G, B, \text{Input}, \text{Output}, M, p_0, \sigma)$, 由于检测网系统是与时间相关, 用 M_n 表示在第 n 个关注时间点下的检测网系统的稳定标识, 此时没有任何变迁发生, 简称实时稳态. 用 $M_n\rightarrow M_{n+1}$ 表示实时稳态从一个第 n 个关注时间点进入到第 $n+1$ 个关注时间点, 并设初始稳态为 $M_0=\varnothing$.

下面定义两个实时稳态的转换, 即定义转移 $\rightarrow$. 设当前实时稳态为 M_n, 若进入下一个实时稳态, 则需要经历以下两步: (1) E_n 的事件实例集合进入到网系统中, p_0 产生新的托肯, 此时的标识 $M'=M_n+\{(p_0, E_{n+1})\}$, 网系统是不稳定的; (2) 变迁或库所按照以下发生规则触发动作, 直到不能发生为止:

(1) 事件流检测网系统 $EFD\text{-System}=(P', T'; A', U', C, N', E', G, B, \text{Input}, \text{Output}, M, p_0, \sigma)$ 的变迁触发规则是:

当变迁的局部条件满足, $|\bullet t|\geqslant 1$, 且满足 G 给出哨函数成真, 则该变迁触发, 触发后, 消耗相关库所中的托肯, 并按照 B 给出的操作, 生成托肯, 将它们放到相应的库所中. 新的标识为 $M'_{n+1}=M'_n-\{(p, E'(p))\mid p\in\bullet t\}+\{(p, E'(p))\mid p\in t\bullet\}$.

(2) 事件流检测网系统 $EFD\text{-System}=(P', T'; A', U', C, N', E', G, B, \text{Input}, \text{Output}, M, p_0, \sigma)$ 的库所消耗规则是:

如果库所 p 中的托肯的起始时间点与第 $n+1$ 个关注时间点之差大于 $U(p)$, 即托肯到达其失效时间, 则消耗该库所, 且新的标识 $M'_{n+1}=M'_n-\{(p, \text{token})\}$,

其中 token 表示达到失效时间的托肯.

4.2 事件检测模型构造

使用事件流检测网系统来构造事件检测模型的基本思路是关注点分离,将事件逻辑结构和事件约束(包括时间约束和属性约束)分层次表示,利用变迁库所之间的连接关系描述逻辑结构,而如哨函数和失效函数等这些与库所和变迁关联的元素,则描述事件时间约束和其他属性约束,以及复合事件实例属性的计算. 这样使得网结构可以清晰地反映复合事件本身的结构,同时网结构的实现更节省系统资源.

基本逻辑网结构

Conjunction、Disjunction 和 Negation 三个逻辑事件操作符规约复合事件的内部逻辑结构. 这三个操作符对应的三个基本网结构如图2所示. 基本逻辑网结构是复杂的事件检测网的基本构造块,同时本身是事件检测网. 例如图 2 中的 Conjunction 结构,其事件检测网可以描述为 $ED\text{-}net'=(P', T'; A', U', C', N', E', G', B', Input', Output')$,其中 $P'=\{p_1, p_2, p_3\}$, $T'=\{t\}$, $A'=\{\langle p_1, t\rangle, \langle p_2, t\rangle, \langle t, p_3\rangle\}$, $U'=\{(p_1, 0), (p_2, 0), (p_3, 0)\}$, $C'=\{A, B, E\}$, $N'=\{(p_1, A), (p_2, B), (p_3, E)\}$, $E'=\{(\langle p_1, t\rangle, (1, A)), (\langle p_2, t\rangle, (1, B)), (\langle t, p_3\rangle, (1, E))\}$, $G'=\varnothing$, $Input'=\{p_1, p_2\}$, $Output'=\{p_3\}$. B' 在没有明确指定的情况下,只进行事件时间的计算,即 $B'=\{E.\text{begin}=\min(A.\text{begin}, B.\text{begin}), E.\text{end}=\max(A.\text{end}, B.\text{end})\}$,其中对于任意事件类型 E, $E.\text{begin}$ 表示检测中所绑定的某一实例 e 的 begine 的值. 如果弧 $\langle p, t\rangle$ 或 $\langle t, p\rangle$ 上没有表达式特别声明,设该弧为 a,则默认 $E'(a)=(1, N(p))$.

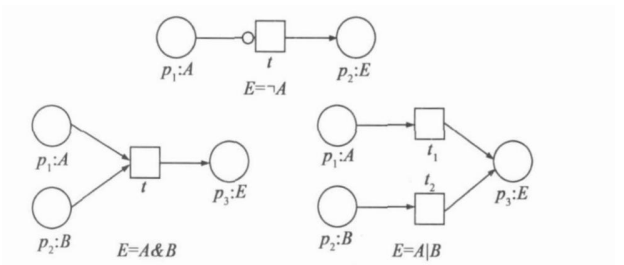


图2 基本逻辑网结构

时间约束和属性约束

对于时间约束,首先考虑 Sequence 事件操作符. Sequence 从某种意义上可以认为是具有时间约束的特殊逻辑操作符. 其网结构在 conjunction 结构上增加哨函数以表达时间约束,如图 3 所示. $(A \overrightarrow{\ } B)$ 的网结构和 $(A\&B)$ 唯一不同之处在于 G 集合不为空. 对于 A^n 这样一种特殊的 Sequence 结构,其网结构如图 4 所示,A^n 内部 n 个 A 之间的约束关系则可以在哨函数中表示. 属性约束可表达为布尔表达式,并用逻辑运算符与原有的哨函数进行进一步的组合. 例如对于 $A \overrightarrow{\ } B(A.x==B.x)$ 这样一个 Sequence 变元,网结构如图 5 所示.

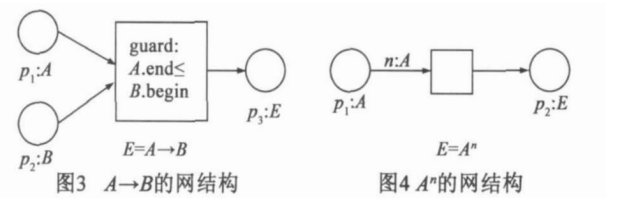


图3 $A \overrightarrow{\ } B$的网结构　　图4 A^n的网结构

对于 Within 操作符,假设事件表达式 $w(A, n)$,我们对给 A 的每个子事件关联一个时间 n,即任意 A 的子事件对应的库所 p,设 $U(p)=n$. 在图形表示上,我们将 n 写入库所内部,如图 6 给出了 $w(A \overrightarrow{\ } B(A.x==B.x), 5)$ 的网结构. 因为一个事件可能被多个 Within 操作符所支配,所以对于与多个 Within 关联的时间,其相应库所的 $U(p)$ 取其中的最小值. 对于 Within 的这种失效时间关联方法,并不能保证一个托肯一旦对产生复合事件实例变得无效就马上从系统中消除,但是触发规则保证了无效托肯在留在系统中内不会影响系统的检测. 如果时刻查询系统中是否有无用托肯会消耗大量的系统资源,而在每个关注的时间点上利用触发规则来进行无用托肯的删除可以提高系统检测效率.

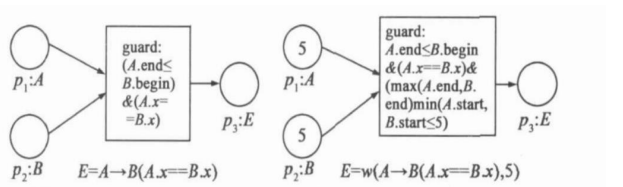


图5 $A \overrightarrow{\ } B(A.x==B.x)$ 的网结构　　图6 $w(A \overrightarrow{\ } B(A.x==B.x), 5)$ 的网结构

复合事件属性计算

托肯是事件类型的实例,其在运行时是一个具有一定属性的对象,因此对于变迁的输出一般情况下应该指出如何计算输出托肯的属性值,这可以通过设置网系统中 B 集合即变迁表达式来完成. 例如,假设 A 为 EPC 事件,定义复合事件 $E_1=A^n$, E_1 关注其子事件的实例个数,表示为 $E_1(\text{begin}, \text{end}, \text{num})$,则其网结构如图 7 所示;定义复合事件 $E_2=A^n$, E_2 关注所有实例的 *EPC* 值,表示为 $E_2(\text{begin}, \text{end}, \text{epcs})$,其中 epcs 为 EPC 标签的集合,则其网结构如图 8 所示. 图 7 和图 8 中变迁中的 body 部分即为变迁表达式,变迁表达式指定了复合事件实例的属性值计算过程.

层次网结构构造

基本逻辑网结构加上约束构成了网的基本构造结

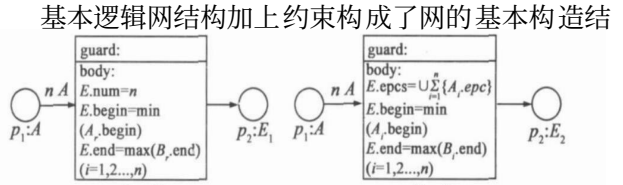


图7 E_1的网结构　　图8 E_2的网结构

构, 而多层次的复合事件即由基本构造结构根据事件表达式逐层构造而成. 复杂网结构的构造可以分解为多步网组合的过程. 对于过程中的每一步, 用一个基本构造块组合已经构造出来的网结构. 组合的过程是当前已构造的网结构的输出库所和基本构造结构的输入库所进行融合的过程. 例如对于复合事件 $E=(A \mid B)\&C^n$, 当前已经有$(A \mid B)$和 C^n 的网结构, 接下来需要用 Conjunction 基本构造块对这两个网结构进行组合, 其过程如图 9 所示. 图 9 只表示了构造过程中库所和托肯结构的变化, 网系统中的其他元素如哨函数等进行相应的变化即可.

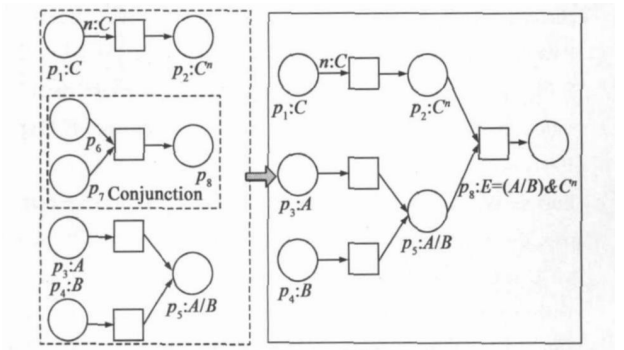


图9　层次网结构构造示意

可检测性分析

对于不完整事件, 根据上文中的定义, 按照构造规则构造出来的事件检测网系统, 可发现其中一定存在如图 10 所示的网结构. 根据事件流检测网系统的变迁触发规则, 图 10 中的变迁 t 永远不能发生, 所以不可能有复合事件的实例产生. 即对于一个事件检测网 ED-net=$(P, T; A, U, C, N, E, G, B,$ Input, Output$)$, 如果$\exists\, t \in T$: $|^\circ t| = 1 \wedge {}^\bullet t = \varnothing$, 则其对应的复合事件定义是不完整的. 由于被动事件的网结构中必然会出现约束弧, 所以如果被动事件可检测, 则其检测网中一定存在如图 11 所示的结构, 库所 p 就是根据被动事件表达式特征加入到网系统中的时间事件类型库所. 事件流检测网系统中, 基于时间事件类型库所产生的时间事件流和运行环境中的原子事件流一起构成了由 p_0 产生的事件流 σ.

图10　不完整事件的网结构

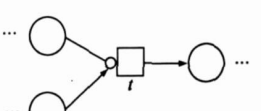


图11　可检测被动事件的网结构

时间事件产生规则

事件检测网系统与原子事件流进行融合成为事件流检测网系统, 而原子事件流中有的一个重要组成部分是中间件自身产生的时间事件实例. 产生时间事件的目的是为了检测被动事件. 伴随着时间事件的产生规则, 相应的需要对网结构进行一定的修改, 即增加表示该类型时间事件的库所, 并添加到网系统中. 本质上图 11 中库所 p 的托肯就是 RFID 中间件进行系统查询的触发器. 时间事件是根据被动事件的表达特征而产生, 产生规则如下:

(1) 如果 Negation 被 Within 操作符直接支配, 即事件表达式形如 $w(\neg A, n)$, 则增加时间事件类型库所 $(0, n, \infty)$;

(2) 如果 Negation 相邻的第一个事件操作符 Conjunction 或者 Sequence, 即事件表达为形如$(\cdots \neg A \& B \cdots)$或者$(\cdots \neg A \overrightarrow{\ } B \cdots)$, 则增加类型时间事件类型库所 $(B.\text{end}, 0, 0)$;

(3) 如果被 Within 操作符直接支配, 且 Negation 右侧的第一个事件操作符为 Conjunction, 即事件表达式形如 $w(\cdots \neg A \& B \cdots, n)$, 则增加时间事件类型库所$(B.\text{end}, n, 1)$;

(4) 如果被 Within 操作符直接支配, 且 Negation 右侧的第一个事件操作符为 Sequence, 即事件表达式形如 $w(\cdots \neg A \overrightarrow{\ } B \cdots, n)$, 则增加时间事件类型库所$(B.\text{end}, 0, 0)$.

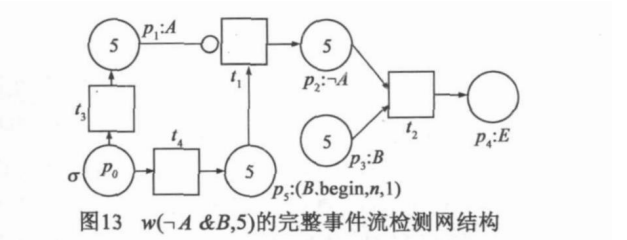


图13　$w(\neg A \& B, 5)$的完整事件流检测网结构

以规则 3 为例对产生规则进行解释. 假设被动事件为 $E=w(\neg A \& B, 5)$, 根据网构造规则产生的网结构如图 12 所示. 因为是被动事件, 还需要使用时间事件产生规则修改网结构, 增加时间事件类型库所$(B.\text{end}, n, 1)$即图 13 中的 P_5, E 的事件流检测网系统如图 13 所示. 从图 13 的网系统中可以发现, 对于每一个 B 事件实例 b, p_0 都会在 end_b 和 end_b+5 这两个时间点产生时间事件实例, 变迁 t_1 则会根据触发规则进行实时稳态的转换. 其对应的检测行为, 即在 B 事件实例发生的时间点和 B 事件实例发生之后 5 个单位时间的时间点, 对当前是否存在有效的 A 事件实例进行查询.

5　PRES

基于 RFID 事件流检测网系统, 我们开发了兼容 ALE 标准的纯 JAVA RFID 中间件 PRES. PRES 采用 ANTLR 进行事件表达式的语法分析, 在遍历事件表达式的抽象语法树的过程中, 根据前述的层次网结构构造规则和时间事件产生规则生成最终事件流检测网结构.

对于一个业务逻辑, 首先需要实现其所关注的原子事件的适配器接口, 从而定义传入 RFID 中间件的原

子事件流; 然后定义复合事件, 得到该复合事件的事件流检测网结构, PRES 支持复合事件运行时动态定义; 最后需要实现事件响应接口, 编写响应复合事件的代码. 以上定制和编码工作完成, 则 RFID 中间件在运行时就会根据系统环境和流入系统的原子事件实例自动产生具有语义的业务事件. PRES 不仅支持基于 5 个事件操作符的事件检测, 还提供了一个运行时容器以及一个基于事件响应的 RFID 应用开发框架. PRES 的结构如图 14 所示.

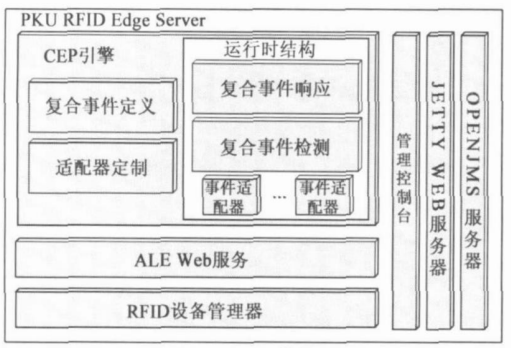


图14 PRES结构

6 总结

为了提高 RFID 应用的开发效率以及系统的可扩展性, 本文基于 PRES, 探讨了一种利用复杂事件处理技术处理高层业务逻辑的机制. 通过把业务逻辑表达为复合事件, 从 RFID 应用系统中分离出通用逻辑, 将对业务逻辑的处理转化为对复合事件的检测. 本文从 RFID 中间件检测行为的角度来探讨了事件时间、事件操作符、事件层次等问题. 然后定义了 RFID 事件检测网系统作为 RFID 事件检测模型的描述工具, 并给出了可执行的检测模型的网结构构造规则. 本文还着重讨论了含有否定事件操作符的事件表达式所描述的一类复合事件, 定义了时间事件来优化对这类事件的检测, 根据事件表达式特征给出了时间事件的产生规则, 并对不可检测事件的网结构特征进行了简要分析. 未来的工作包括对事件操作符的代数性质的进一步研究, 事件检测正确性验证以及对事件检测性能的提高.

致谢 感谢袁崇义老师在 Petri 网建模方面对我的耐心指导. 感谢讨论班上胡文惠老师以及刘学洋、刘殿兴和李信鹏等同学的建议. 感谢邓鹏鹏、黄开木同学在 PRES 复杂事件处理相关模块的开发中所做的工作.

参考文献:

[1] EPCglobal. The Application Level Events(ALE) Specification Version 1. 1[S/OL]. http://www. epcglobalinc. org/standards/ale/ale_1_1-standard-core-20080227. pdf, 2008.

[2] EPCglobal. The EPCglobal Architecture Framework[S/OL]. http://www. epcglobalinc. org/standards/architecture/architecture_1_2-framework-20070910. pdf, 2007.

[3] EPCglobal. EPC Information Services(EPCIS) version 1. 0. 1 specification[S/OL]. http://www. epcglobalinc. org/standards/epe-is/epcis_1_0_1-standard-20070921. pdf, 2007.

[4] David C. Luckham, Brian Frasca. Complex Event Processing in Distributed System[R]. USA: Stanford University Technical Report CSL-TR-98-754, 1998-03.

[5] David Luckham. Event Processing Glossary[Z/OL]. http://complexevents. com/? p=195, 2006.

[6] Antony Galton, Juan Carlos Augusto. Two approaches to event definition[A]. Proceeding of Database and Expert Systems Applications 13th Int. Conference(DEXA' 02)[C]. Aix-en-Provence, France, 2002. 547-556.

[7] S Chakravarthy, D Mishra. Snoop: An expressive event specification language for active databases[J]. Data Knowl, Eng, 1994, 14(1): 1-26.

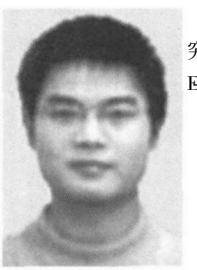
[8] Eugene Wu, Yanlei Diao, Shariq Rizvi. High-performance complex event processing over streams[A]. Proceeding of SIGMOD 2006[C]. Chicago, USA, 2006. 407-418.

[9] Fusheng Wang, Shaorong Liu, Peiya Liu, Yijian Bai. Bridging physical and virtual worlds: Complex event processing for RFID data streams[A]. The 10th International Conference on Extending Database Technology(EDBT)[C]. Munich, Germany, 2006. 588-607.

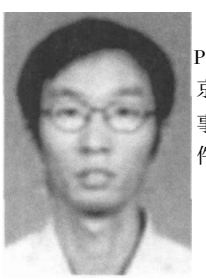
[10] S Gatziu, K R Dirtrich. Detecting composite events in active databases using petri nets[A]. Proceeding of 4th international Workshop on Research Issues in Data Engineering: Active Database Systems[C]. Houston, USA, 1994. 2-9.

[11] Esper Reference Documentation[R/OL]. http://esper. codehaus. org/esper-1. 0. 0/doc/reference/en/pdf/esper_reference. pdf, 2007-09.

作者简介:



叶 蔚 男, 1985 年出生, 博士生, 主要研究领域为软件工程和复杂事件处理技术.
E-mail: cactus. ye@pku. edu. cn



黄 雨 男, 1979 年 1 月出生, 理学博士, Petri 网专业委员会委员. 2007 年 7 月毕业于北京大学信息学院, 获理学博士学位. 目前主要从事 Petri 网、模型检测、Web 服务组合、RFID 中间件理论和应用的研究.