**尊敬的杨老师：**

您好，这是我结合10月18号杨老师对我的指点（即了解国内外通用的开发方法和实现的技术手段做相应的了解），通过查阅文献和查阅一些网站，总结出本周的周报告。

首先，最大的收获就是本周末，在读外文文献的时候，通过文章的内容找到了专门发布关于复杂事件的的<http://www.complexevents.com/>这个网站，下面的图1是这个网站的主页，这个网站将复杂事件处理的现有研究成果（很多东西只有个索引，像IBM这类的企业研究的东西很难获取资料），还有一些专业术语的解释，都在这上面写出来了，同时他们还给出了现有研究的系统的基本信息，如下面图2（需要放大才能看得比较清晰）。这有利于我后续的继续跟进，同时和国外的比较成熟的做对比（下面第3小节中蓝色字体部分。）能找到应该怎么去做，或者是利用他们现有的成果。

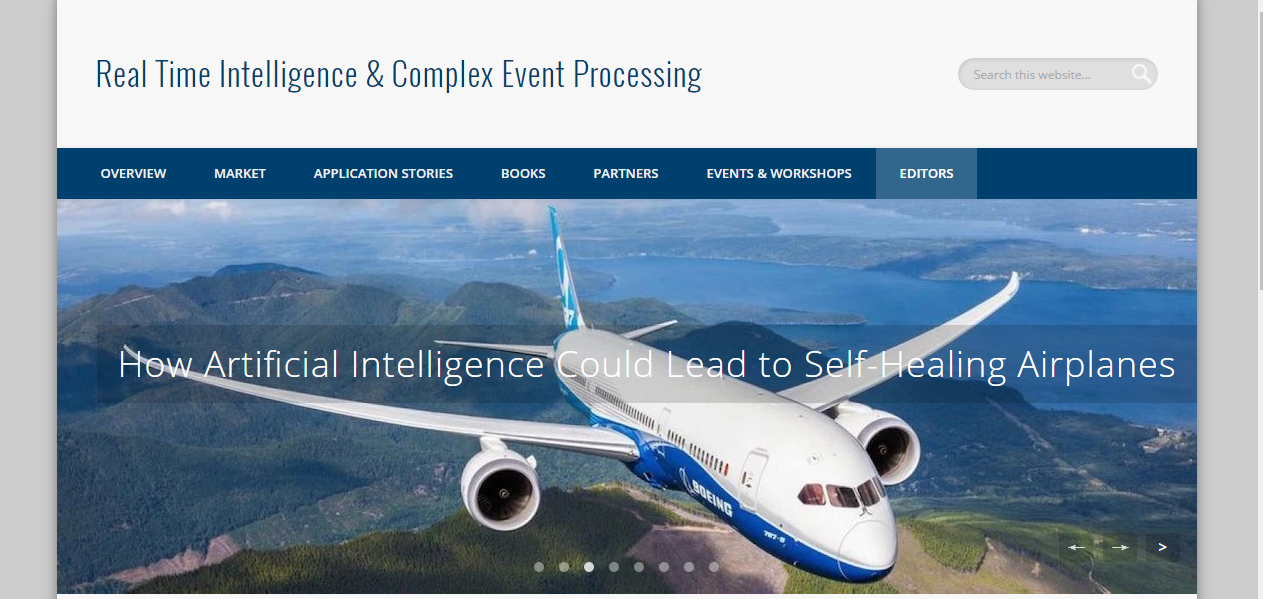


图1

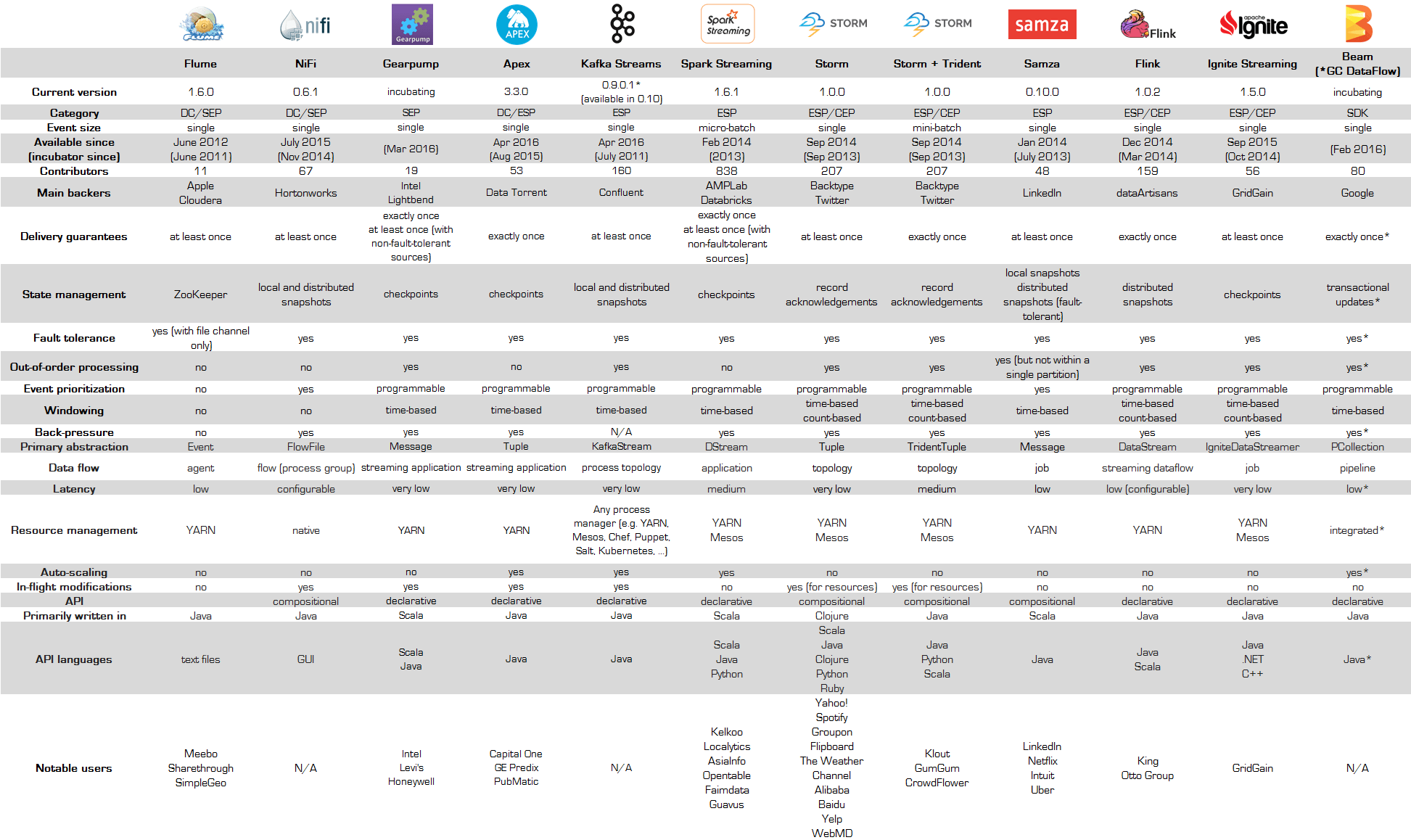
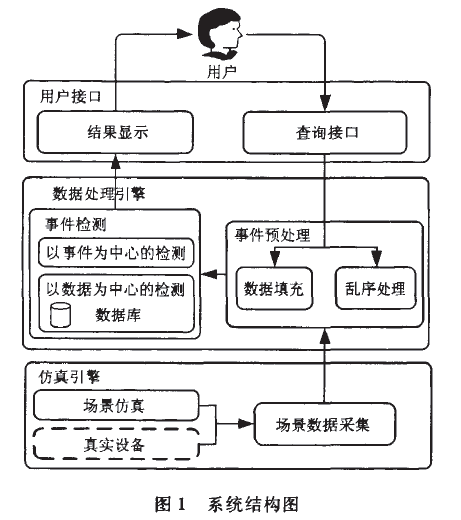


图2

1. **国内研究方法**

谷峪, 胡小龙, 李传文,等. REvent:一种面向RFID应用的复杂事件处理系统[J]. 计算机研究与发展, 2010, 47(z1). 谷峪的邮件：[guyu@ise.neu.edu.cn](mailto:guyu@ise.neu.edu.cn)（Ps给他发了邮件，还没有回复。）

REvent的系统框架图

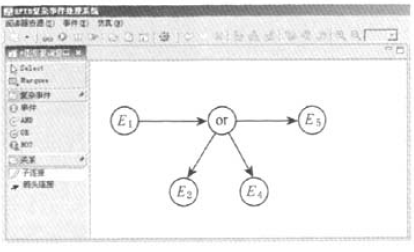


涉及到的关键技术：

**数据预处理技术**

**事件检测技术**

评语：从系统界面上能看出他们是用的java开发的一个小程序，从上面的系统结构图我们也可以看出他们的处理系统主要还是以人为中心，智能化处理程度还是偏低。



陈皓, 李瑜, 虎嵩林,等. 基于S4框架的并行复杂事件处理系统[J]. 通信学报, 2012(s1):165-169.

**关键技术：**

**基于操作符的负载分流优化方法**：当面对高速数据流时，如果能将负载均匀地分流到集群的各个节点，并行地处理事件流，则可以增加系统的吞吐量。然而复杂事件处理的特殊性在于事件往往不是孤立地存在，在处理中需要根据事件流上下文或其他事件流进行决策，单纯地拆分数据流到不同节点，会影响事件间的关系，导致处理出错。

**系统概述：**

S4（simple scalable streaming system）是一个去中心的、分布式的、可扩展的流式处理系统。类似于Storm 的消息分组策略，S4 框架通过对流事件的关键字进行散列计算，得到处理事件的节点编号，再将事件发送到对应节点进行处理。通过修改S4框架的事件分流机制，基于操作符的不同，使用不同的分流策略，实现了并行的通用CEP 系统。（从文章中未能发现他们所做的开发内容）

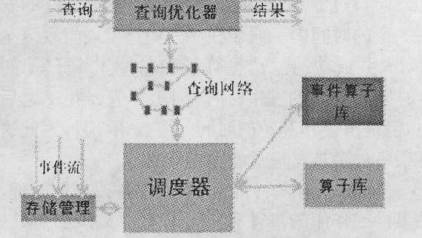
魏永超, 陈立军. 数据流上复杂事件处理系统Eagle的设计与实现[C]// 中国数据库学术会议. 2008.

涉及到的关键技术：

事件代数

事件操作符

聚集操作符  
系统架构：



**实验环境：**Inter酷睿E4400双核 2.00GHz Windows XP

根据文章资料可以推断数据库是用的Oracle，主要使用java语言开发的（和我实验室的应该差不多）。

1. **国外研究方法（这里主要是翻译了一篇英文文献，在8号汇报中提到的Cayuga）**

Brenna L, Demers A, Gehrke J, et al. Cayuga: a high-performance event processing engine[C]// Proceedings of the 2007 ACM SIGMOD international conference on Management of data. ACM, 2007:1100-1102.（这个系统主要是一个Web系统，根据文中的信息，应该是用Python+ajax开发的，可以参考其开发事件处理的方法。）

不断增加的数据量作为高速事件流到达。库存报价数据，网络流量数据，在传感器网络中收集的数据，RFID流只是这种流的几个示例。此外，随着网络上的动态内容（博客，新闻，拍卖）的数量迅速增长，网络正在成为许多有趣和复杂的事件流的来源。诸如医疗保健，环境监测，供应链管理和合规性检查等新兴企业级应用以及个性化新闻馈送和信息聚合等消费者应用都需要过滤，关联和聚合各个事件。所得到的复杂事件查询超出了发布/订阅，因为查询涉及多个事件之间的相关，而不是对单个事件的简单谓词。在康奈尔，我们建立了Cayuga系统，一个用于复杂事件处理的高性能系统[1,2]。 Cayuga将用于组成状态查询的简单查询语言与基于具有缓冲区的非确定性有状态自动机的可扩展查询处理引擎组合。 Cayuga的一个重要特征是它不仅可以扩展流中事件的到达速率，而且还可以扩展查询数量;可扩展性的后一维对于消费者应用尤其重要，其中数百万用户可以具有向系统注册的查询。在本演示中，我们将展示Cayuga的应用程序，用于监控来自Web Feed聚合器的事件流，用于个性化新闻聚合和通知。我们首先给出Cayuga系统的高级概述，介绍一些我们将在演示中显示的Cayuga查询（第2节）。然后我们详细描述实际演示（第3节）。

在Cayuga中，每个事件流都有一个固定的关系模式，并且流中的事件被视为关系元组。

每个事件具有两个时间戳，开始时间和检测时间，对事件可以具有非零但有限的持续时间的事实进行建模。 事件按其检测时间的顺序串行化; 具有相同检测时间的事件被认为同时发生，并且不能保证它们的序列化顺序。 几个有趣的理论问题导致了这种设计。

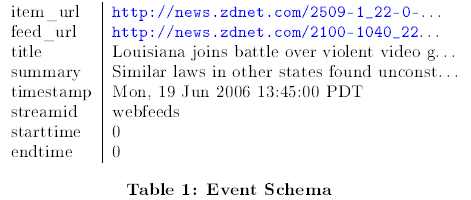
Cayuga查询语言是从事件代数派生的[1]; 它是代数运算符的简单映射一个SQL语法，在精神上类似于复杂事件SASE [4]中的语言。 每个查询具有以下形式：

SELECT <> attributes

FROM <> algebra\_expression

PUBLISH <> output\_stream

SELECT子句指定输出流模式中的属性，FROM子句指定Cayuga事件模式，PUBLISH子句给出输出流的名称。 事件模式可以用三个不同的运算符构建。 FILTER {？}运算符从满足谓词？的输入流中选择那些事件。



演示查询1：在演示中，我们将通过以下查询来说明FILTER运算符，其中包含由整个流中的Google新闻发布的所有新闻项目：webfeeds？ 具有表1中所示的模式。

SELECT \* FROM

FILTER {feed\_url ='http：//news.google.com/'}（webfeeds）

PUBLISH google\_news\_items

一个强大的结构允许我们随时间关联事件是排序运算符NEXT {？}。 当应用于两个输入流S1和S2作为S1 NEXT {α} S2时，运算符将来自S1的每个事件与S2中满足谓词α的下一个事件组合。 并且发生在S1中的事件的检测时间之后。

演示查询2.此查询包含某些网站发布的所有新闻项目，其后是一篇关于Google新闻的文章。 请注意，此查询将Query 1的输出作为其输入流之一。 用户定义函数包含执行子字符串匹配; $ 1和$ 2指的是NEXT运算符的两个输入流。

SELECT $ 2.summary，$ 1.item\_url

FROM（webfeeds）

NEXT {contains（$ 2.item\_url，$ 1.item\_url）= 1}（google\_news\_items）PUBLISH re？ed\_by\_google\_news

虽然NEXT允许我们关联两个事件，但是在许多情况下，我们需要遍历已知数量的事件，直到满足停止条件。 此功能由FOLD操作员提供。 直观地，FOLD是NEXT运算符的泛化，因为它查找包括两个或更多事件的模式。 我们将使用下一个例子来描述FOLD。

演示查询3.此查询发送通知

每当一个iPod在新闻中受欢迎（即，有

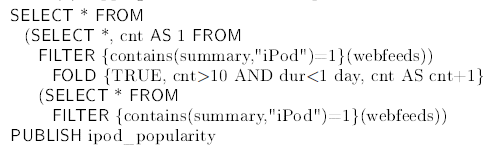
至少十篇文章在一段时间谈论iPod

持续时间）。 FOLD运算符包含三个表达式：

（1）迭代的条件

（2）迭代的停止条件

（3）迭代步骤之间的映射。



Cayuga在时代处理事件;在历元期间，处理具有相同检测时间的所有事件。查询被实现为具有如下工作的缓冲器和自循环的非确定有限状态自动机。自动机中的每个状态都分配有固定的关系模式。每个边缘，例如在状态P和Q之间，由三元组hS，α，fi标记，其中S标识输入流; ？是模式（P）×模式（S）的谓词;和f，模式映射是将模式（P）×模式（S）转换为模式（Q）的函数。 NFA操作如下。假设NFA实例处于具有存储数据x的状态P（注意x符合模式（P））。让事件e在流S上到达，使得满足α（x，e）。然后，机器非确定地转变到状态Q，并且存储的数据变为f（x，e）。 Cayuga支持重新订阅，一个类似于关系数据库系统中的查询计划的概念：来自一个查询的输出事件流可以用作一个或多个其他查询的输入流。重新订阅启用非常复杂的事件模式查询，并且它显着扩展了查询语言的表达性。该系统的更多细节可以在我们最近的出版物中找到[1,2]。

**3演示概要**

在本节中，我们提供了如何演示上面显示的查询的概述，以Demo Query 2为运行示例。 我们的演示包括以下三个组件：（i）基于Web的前端，供用户输入查询; （ii）轨迹可视化器，其显示Cayuga引擎的内部状态和基于自动机的查询表示; 和（iii）显示查询评价结果的输出可视化器。 在演示期间，我们将对来自Web Feed聚合器的事件流运行查询。 让我们现在通过查询2的执行来呈现上述每个组件。该查询是针对包含来自于2006年6月和10月之间记录的418个频道的225365个源项目的事件流来处理的。

**4 提交Cayuga查询**

基于Web的前端正在自定义的Python Web服务器上运行，基于AJAX的控件用于异步通信和浏览器中的用户友好界面。 Cayuga有一个基于Web的前端，用户可以输入持久性查询并使用正在运行的Cayuga引擎注册它们。为了方便起见，用户可以从下拉菜单中为所述查询语言选择预定义的模板。然后，他们可以根据自己的需要修改它们，或从头开始写一个。由于Cayuga查询可以轻松跨越多行，因此我们提供的多行输入框比通常在基于Web的查询界面中看到的多。图1中选择进行编辑的查询是显示查询2的查询前端的屏幕截图。请注意，此版本的查询是内联的，没有重新订阅。在提交查询时，查询将被编译并且在Cayuga之外。用户将收到一条带有查询ID或错误消息的消息，查询未编译，并且必须在更改后重新提交。用户可以选择批处理多个查询，然后再按EXECUTE向Cayuga注册查询。可以通过按清除来清除批次。

跟踪Visualizer

Cayuga可以输出其内部状态在事件之间如何变化的连续跟踪。此跟踪将写入文件，其中包含查询自动机的描述，后跟事件和状态实例。我们的可视化器读取跟踪文件，并使用基于Java Swing的GUI来显示事件如何匹配查询自动机中的谓词。

为了精确地查看加载的查询如何对输入流中的事件起作用，用户可以选择向前播放，在任何给定点暂停，或者逐步进行跟踪。这允许用户详细调查任何单个事件如何影响查询自动机的内部状态。在图2和图3中的可视化器的工具栏上看到的控件是（从左到右的顺序），用于放大和缩小自动机，调整与工具栏的垂直距离，以及用于回放。输入和输出流在动画中被命名和给定不同的颜色。在任何时候，如果用户点击自动机中的一个状态，将弹出一个文本框，以显示处于此状态的实例。单击边缘将显示该边缘上的谓词。图2示出了具有用其谓词注释的过渡边缘的查询2的自动机。在图3中，自动机已经处理了一些事件。流到目前为止产生了状态1的7个实例，状态2的4个实例。右边的文本框显示状态2的内容，向下滚动以查看最后一个实例达到这个状态。从文字中可以看出，http://news.zdnet.com在3月12日（星期三）06:00发布的故事是由http://news.google.com在当天的13:00链接到的。

**输出**

当生成见证事件（即查询结果）时，Cayuga当前将其转储到Web前端正在拖尾的文件中。 然后将这些文件放入一个BUYER中，其中频繁轮询AJAX函数可以获得它们并且在终端用户的web浏览器的HTML页面中显示，并按查询排序。 图4是屏幕截图，显示了查询2出现的第一个两个见证。

**结论**

Cayuga是一个成熟的事件处理系统，有趣处理模型。 我们相信它的功能演示将对数据库社区感兴趣。

**3．网站资料及对应译文**

# An Overview of Event Processing Software

A large variety of commercial and open source event processing software is available to architects and developers who are building event processing applications. These are sometimes called event processing platforms, complex-event processing (CEP) systems, event stream processing (ESP) systems, or distributed stream computing platforms (DSCPs).

种类繁多的商业和开源事件处理软件可供建筑师和开发商在构建事件处理应用。这些有时被称为事件处理平台，复杂事件处理（CEP）系统，事件流处理（ESP）系统，或分布式流计算平台（的DSCP）。

Some examples are:

1. Apache Samza
2. Apache Spark
3. Apache Storm
4. Codehaus/EsperTech’s Esper, Nesper
5. DataTorrent RTS (Real-time Streaming)
6. FeedZai Pulse
7. FujitsuInterstage Big Data Complex Event Processing Server
8. Hitachi uCosminexus Stream Data Platform
9. IBM InfoSphere Streams
10. IBM Operational Decision Manager (ODM)
11. Informatica RulePoint
12. LG CNS’ EventPro
13. Microsoft StreamInsight
14. OneMarketData OneTick CEP
15. Oracle Event Processor
16. RedHat Drools Fusion/JBoss Enterprise BRMS
17. SAP Event Stream Processor
18. SAS DataFlux
19. ScaleOut Software
20. SQLStream s-Server
21. Software AG Apama Event Processing Platform
22. Tibco BusinessEvents
23. Tibco StreamBase
24. Vitria Technology Operational Intelligence Analytic Server
25. WS02 CEP Server、

一些例子是：

1. 阿帕奇Samza
2. 阿帕奇星火
3. 阿帕奇风暴
4. Codehaus / EsperTech的蛾，Nesper
5. DataTorrent RTS（实时流）
6. FeedZai脉冲
7. FujitsuInterstage大数据复杂事件处理服务器
8. 日立uCosminexus流数据平台
9. IBM InfoSphere Streams的
10. IBM经营决策管理器（ODM）
11. Informatica的RulePoint
12. LG CNS'EventPro
13. 微软的StreamInsight
14. OneMarketData OneTick CEP
15. Oracle事件处理器
16. RedHat的Drools的融合/ JBoss企业BRMS
17. SAP事件流处理器
18. SAS DataFlux的
19. ScaleOut软件
20. 的SQLstream S-服务器
21. Software AG公司Apama事件处理平台
22. TIBCO BusinessEvents
23. TIBCO StreamBase
24. Vitria技术运营智能分析服务器
25. WS02 CEP服务器

These subsystems offer a wide range of different features, but all perform CEP in a technical sense. CEP (or ESP) is a technique in which incoming data about what is happening (event data) is processed more or less as it arrives to generate higher-level, more-useful, summary information (complex events). Event processing platforms have built-in capabilities for filtering incoming data, storing windows of event data, computing aggregates and detecting patterns. In formal terminology, CEP software is any computer program that can generate, read, discard or perform calculations on complex events. A complex event is an abstraction of one or more base (input) events. Complex events may signify threats or opportunities that require a response from the business. One complex event may be the result of calculations performed on a few or on millions of base events from one or more event sources.

这些子系统提供范围广泛的不同特点，但都在一个技术意义上进行CEP。CEP（或ESP）是其中关于正在发生的事情（事件数据）的输入数据被处理或多或少即立即产生更高级别的，更是有用的，摘要信息（复杂事件）的技术。事件处理平台具有内置的功能，用于滤波输入数据，存储事件数据的窗口，计算聚集体并检测图案。在正式的术语，CEP软件是可以产生，阅读，丢弃或复杂事件进行计算的计算机程序。复杂事件是一个或多个基站（输入）事件的抽象。复杂事件可能意味着需要从企业的响应威胁或机遇。一个复杂的事件可以是在一些或上百万碱事件从一个或多个事件源执行的计算的结果。

**事件处理平台有很多：**

* 建立，被广泛使用的商用产品，如IBM的ODM和InfoSphere Streams的，SAP的ESP（以前的Sybase埃勒瑞），Software AG的Apama，TIBCO的业务活动和StreamBase，和其他人，是多方面的开发和运行软件套件，包括适配器与整合事件源，开发和测试工具，仪表板和报警工具和管理工具。
* 其他事件处理平台相结合，与功能，如查询，报表，互动分析，警报或​​关键绩效指标的工具，这是在运行智能应用专门针对。例子包括FeedZai脉冲，的SQLstream S-Server和Vitria Technology的作战情报。
* 新兴的DSCP，如Apache Samza，Spark和风暴，是没有全面的本地CEP分析功能和相关配件的通用平台。他们是高度可扩展的和可扩展的，因此开发人员可以添加逻辑来处理多种流处理应用程序。
* 这些所描述的DSCP，与蛾/ Nesper和RedHat的Drools的融合/ JBoss企业BRMS（两者均为CEP系统）一起，是开源产品。
* DataTorrent的RTS和IBM的InfoSphere流被划分为的DSCP或CEP系统，根据上下文和作者的观点。
* 有些事件处理平台捆绑在一起的规则引擎。例子包括IBM的ODM，RedHat的Drools的融合/ JBoss企业BRMS和TIBCO业务活动。

CEP使用量迅速增长，因为CEP，从技术意义上说，是让在实时或近实时事件流信息的唯一途径。该系统具有处理事件数据或多或少到达使得相应的操作可以快速地作出。在这篇文章中列出的子系统是通用开发和运行时工具，用于开发人员构建自定义，事件处理应用程序，而无需重新实现核心算法处理事件流。然而，这些子系统是唯一的几种方式来获得的CEP的功能之一，事实上，该实施CEP逻辑大多数应用程序不使用这些子系统。

最CEP是作为较大产品的一部分获得。公司收购打包应用程序或订阅已在幕后嵌入式CEP一个SaaS服务。公司正在购买这种情况，要求事件处理的溶液中，它可能没有意识到正在使用的CEP。例如，供应链可视化产品; 安全性信息，甚至管理（SIEM）产品; 某些类型的欺诈检测和治理，风险和法规遵从（GRC）的产品; 系统和网络监控系统; 业务活动监控（BAM）工具; 和许多其他类别的软件实现CEP逻辑的一些或大或小数额。在少数情况下，这些产品或SaaS产品的开发人员已经利用上面列出，以减少他们必须编写的代码量的通用事件处理平台。但在大多数情况下，开发人员实现的在新的代码事件处理算法的一个专门的子集，以满足他们的应用程序的目的。

一些用户企业都写有CEP逻辑的自定义应用程序，而不是利用一个现成的现成事件处理平台。这是20世纪90年代和21世纪初尤为常见上面列出的子系统广泛使用之前，一些开发商仍然选择写性能或成本的原因自己CEP逻辑。例如，大型银行及相关金融服务公司已经做了前端办公系统的资本市场有自己的嵌入式CEP逻辑交易。

每个公司都有它已经使用了幕后CEP实现一定量的应用程序或SaaS解决方案的地方。越来越多的公司需要的那种这里列出来支持高要求，高吞吐量，低延迟的应用中，事件处理逻辑是主要的业务问题，事件处理子系统。在这两种情况下，你越了解CEP，更好的将您的应用程序体系结构。

总结：本周的汇报很乱，内容也很多，还望杨老师见谅。我会尽快的消化吸收，将现有典型的系统开发方式，整理出来。

此致

敬礼

文波

2016/10/23