业务流编排中基于RESTful的半自动派生

摘要

企业为了向其他组织提供合作，为其提供复杂的产品和服务市场。信息促进了不同组织之间的这种合作与协调，获得了很好的技术分享.BPMN流程编排是一种建模语言，用于指定业务级别上不同组织之间的信息和服务交换。最近，REST体系结构风格在网络上提供服务的使用激增，但很少有系统的工程方法来设计它们的协作。在本文中，我们通过定义用于从过程编排中推导出RESTful编排的半自动方法，以全面的方式解决这个缺陷。该方法基于自然语言分析技术来从过程编排中的文本信息导出相互作用。所提出的方法在效率方面进行了评估，从而导致网络工程师仅对所有生成的RESTful交互作用的约10％进行干预。

关键字:业务流程编排 RESTful编排 自然语言分析

# 1 引言

传统上，BPM的研究集中在组织的内部流程上。 更复杂的服务趋势将BPM扩展到多个流程之间交互的视角。 这种由信息技术支持的交互需要标准模型，如BPMN [1]，所有参与者都可以理解。 特别是，BPMN业务流程编排指定了两个或多个参与者之间的交互以及这些交互在业务级别发生的顺序。 在技术层面上，REST [2]正日益成为Web上提供服务的首选架构风格，从而导致RESTful API的主流开发。

涉及具体网络交互的RESTful API的设计应基于通过编排图建模的业务交互。但是，将业务交互降低到RESTful交互级别具有挑战性。编排图的设计者通常是业务流程领域的专家，并且不具备软件开发的知识。网络工程师在业务流程编排方面也是如此，例如，在设计URI时，他们通常与实现的角度太接近，这使得RESTful API的客户端更难以理解URI [3]。在[4]中描述了解决这个问题的第一个概念，然而，概念评分和评估存在局限性。

在本文中，我们以全面的方式解决了这一研究缺陷，并提出了一种将标准BPMN编排图作为输入的方法，并将其作为输出生成RESTful编排[5]。 RESTful编排是一种非常有用的[6,7]基于BPMN的建模语言，专门针对不同业务参与者之间的RESTful交互规范。我们的方法基于自然语言处理技术，该技术使用编排任务的文本描述来映射到具有相应REST URI的最合适的REST动词。我们在研究原型中实现了我们的方法，并将其应用于来自不同领域的一组编排图。派生的REST请求也由REST专家进行了评估，确认了我们方法的有用性。创建的RESTful编排用于派生有助于开发REST API的代码框架。应用我们的方法意味着所有参与者都使用相同的REST接口生成逻辑，这有助于更好地理解，维护和演进参与者的RESTful API。

这种方法扩展了[4]中的工作，提供了一个增强型标签分析方法和一个更大的评估数据集的扩展评估。这种扩展可以在正确生成RESTful交互的过程中产生更高的效率，其中仅需要大约10％的时间来进行网络工程师干预。

本文组织如下。第2节介绍了编排图和RESTful编排图。这些概念通过一个运行的例子来说明。第3部分介绍了我们的派生RESTful编排图的半自动方法。第4节讨论了用户评估的设置和结果。第5节提供了第Sect节前的相关工作。 6结束本文并介绍了未来的工作。

# 2 预备

本节通过示例的帮助简要描述编排图。 此外，在引入RESTful编排图的概念之前，将解释REST架构风格。

## 2.1 编排图

BPMN 2.0 [1]中介绍的业务流程编排图是一种建模语言，它侧重于描述两个或更多参与者之间的交互作用，他们通常是企业主角，例如企业，客户或组织。 与业务流程模型相比，编排图抽象出参与者的内部流程，并指定参与者之间交换消息的顺序。

图1描绘了一个编排图的例子。 该图描述了参与提交，审核和组织过程的不同参与者之间的相互作用，以安排科学会议。 会议中的一些主要利益相关者包括组织者，作者和评论者。 该图描述了这三位参与者之间的相互作用，这些参与者从发表电话会议（CFP）开始，最终结束，确认纸质出版物。

为了促进这些互动，参与者使用了一个评论管理系统（RMS），在我们的案例中，这个系统受到了http://easychair.org的启发。 RMS负责在整个协作过程中协调这三名参与者。

如图1所示，编排图的主要元素是编排任务（图形描绘为圆角矩形）。它代表两个参与者之间的消息交换。发起消息交换的参与者称为发起者，而另一个参与者称为接收者。返回消息是可选的，可以从接收者发送到发起者。为了以图形方式区分发起者和接收者，后者总是以灰色突出显示。这同样适用于启动和返回消息，尽管消息不需要以图形方式描述。

编排图定义了进行交互的顺序。编排任务具有通过顺序流建模的顺序依赖性。序列流，事件和网关以与常规BPMN业务流程模型中类似的方式使用。但是，在编排图中只能使用事件和网关的一个子集。图1中的事件是开始事件，计时器事件最后期限和三个结束事件：拒绝论文;短论文下降;论文接受发表。

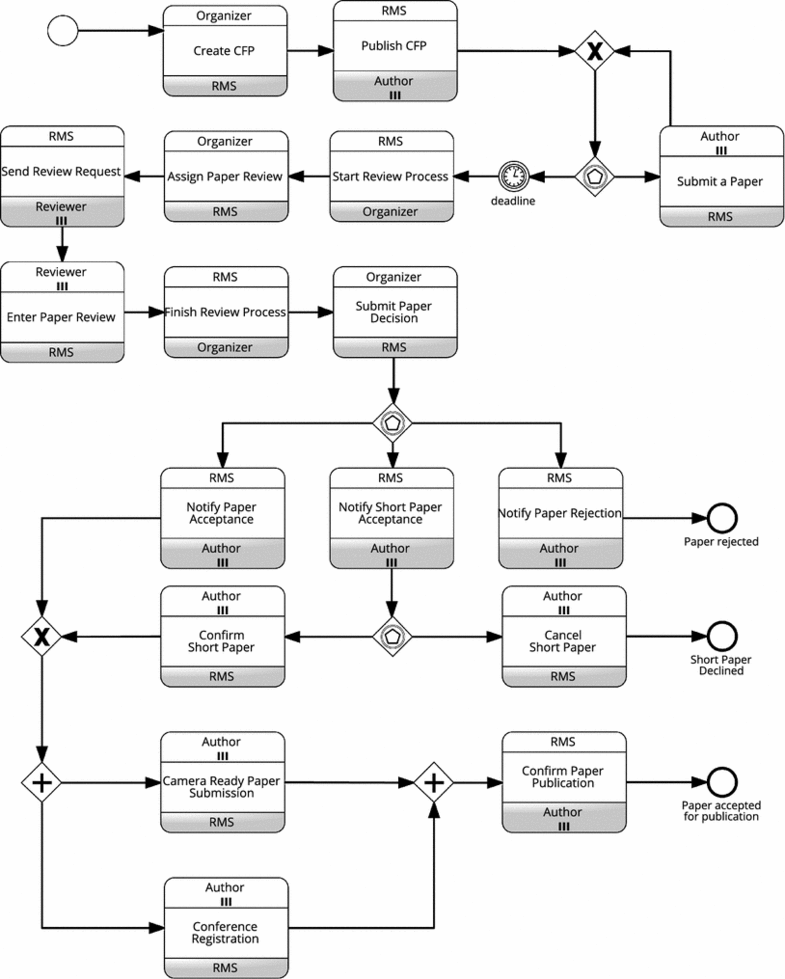


图1 提交论文和审查管理的编排图

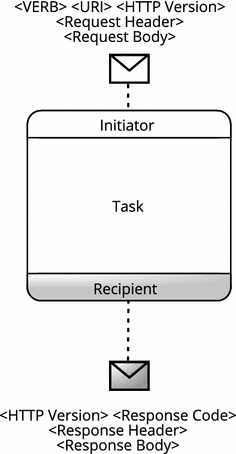


图2 RESTful编排中编排任务的注释[5]

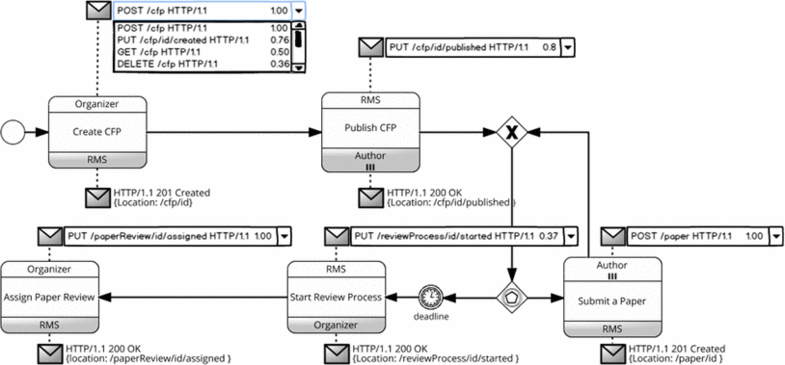


图3 RMS生成的RESTful编排的一部分

## 2.2 RESTful编排图

REST架构风格[2]越来越多地用于开发RESTful Web服务。其架构限制有助于提高可扩展性和可移植性。在几乎所有情况下，REST都使用HTTP协议作为不同参与者之间交互的手段。交互是通过对资源使用标准HTTP动词（GET，POST，PUT，DELETE）来实现的。驻留在服务器上的资源通过URL全局唯一地标识。客户可以通过这些REST动词改变其状态。由于无状态约束，服务器不需要记住之前与客户端的交互以了解客户端的请求。

RMS例子的假设是所有参与者都是RESTful服务，即他们通过发送REST调用来相互交互。在简单的浏览器设置中，组织者，审阅者和作者是RMS的用户。在这个例子中，我们假设他们也提供了一个RESTful API。将这些服务视为RMS移动应用程序，其中RMS可根据用户角色推送通知[8]，例如，向审阅者通知分配给审阅的论文，将论文决定发送给作者，通知组织者何时提交所有审阅。

业务流程可以用来模拟[9]中提出的参与RESTful会话的参与者的内部行为。但是，当涉及到多个参与者之间的交互时，重要的是关注全球视角，以便捕捉到公共资源的状态以及与这些资源的交互。

为此，Nikaj等人[5]介绍RESTful编排图 - 一个轻量级的BPMN编排图，增加了REST细节。这些细节包括表示RESTful交互的编排任务的注释，称为RESTful任务，如图1中的Submit paper decision编排任务。这是通过将两个编排任务消息分别改进为REST请求和一个REST响应，如图2所示。图3提供了一个RESTful编排图的摘录，模拟组织者，RMS和作者之间的RESTful交互。但是，负责使用REST符号增强编排任务的人员必须了解编排的业务方面和RESTful交互的实现方面。这个问题在我们的论文中提出，提出了一种从业务流程编排中派生RESTful编排的半自动方法。

# 3 RESTful编排的半自动生成

本节介绍我们用于生成RESTful编排的半自动方法。 第3.1节讨论了该方法的相关概念。 第3.2节花费在上一节中，侧重于编排特定的标签风格。 然后第3.3节将解释如何使用这些概念来标识在编排任务标签中表示的REST请求的类型。 3.4节展示了如何通过RESTful信息最终丰富编排任务。

## 3.1 基础

本小节从编排图的正式说明开始，作为我们方法的核心工件。 我们使用编排图的所有元素来生成RESTful编排图。 我们认为编排图是一个元组

C =（N，S，P，L，label），使得：

-N =T∪E∪G是一组节点。 T，E和G是成对相互排斥的;

T是一系列的编排任务;

E是一组事件;

G是一组网关;

S⊆N×N是一组序列流;

P是一组参与者;

L是一组自然语言文本标签;

label：T↦L是一个将文本标签分配给编排任务的功能。

正如第2节所指出的，编排任务可以表示单个消息交换或两个消息交换（即发送消息和回复消息）。 在我们的方法中，我们将每个案例映射到单个REST式交互，因为发起者向收件人发出REST请求。 由于根据BPMN规范[1]，可选的第二条消息是返回消息，因此我们不认为它是新的REST请求，而是来自HTTP中嵌入的接收方（REST术语中的服务器）的响应体（见图2）。 此外，我们观察到，受自然语言处理影响的编排任务标签与其相应的业务流程发送任务相比，更接近于接收任务。 这意味着编排任务标签不提供有关如何命名第二个REST请求的信息。

为了处理标签的文本信息，有必要以结构化的方式访问这些信息。 作为一个起点，我们观察到，如上所述，编排任务被类似地标记为活动，通常指业务过程模型中的相应发送任务[1]。 因此，我们假设编排任务的每个标签都包含以下组件：一个动作和一个应用动作的业务对象[10]。 例如，考虑图1中的提交文章决策的标签。它包含要提交的操作和业务对象文章决策。 重要的是要注意，这些组件可以用不同的语法变体进行通信。 例如，标签相机准备好的文章提交通过使用表达要提交的动作的名词在不同的语法结构中传达动作。

为了独立于语法标签结构，我们依赖Leopold等人的标签注释方法[11]，它确定了相当程度的准确性（平均精确度：91％，平均召回率：90.6％）的行为和业务对象。 考虑到l = label（t）∈L是任意编排任务的标签，并且分别考虑WV和WN来描述所有动词和名词的集合，我们参考l的动作和业务对象如下：

α：L↦WV是一项将动作分配给编排任务标签的功能;

β：L↦WN是将业务对象分配给编排任务标签的函数。

作为一个例子，考虑从图1中提交文章决策标记的编排任务。根据之前的概念化，该行动由α（提交文章决策）=提交并且业务对象由β给出（提交文章决策 ）=文章决定。 我们将在下面使用这些标签组件来派生各自的REST请求，并从编排任务的文本标签中生成RESTful注释。

## 3.2 编排特定的标签分析

在前述的标签处理方法中，我们观察到关于编排任务和业务流程任务的相似性的假设并不总是成立。由于编排任务代表跨组织发送的消息，因此标签倾向于包含发送单词或类似的单词，如enter，submit，mail，它描述了从一个编排参与者到另一个参与者的消息传递。这些单词后跟单个或多个单词，例如，order, invoice, request, application request, paper submission, short paper acceptance。由于所有REST请求都是发送消息，因此send这个词表示对任何特定REST动词没有语义偏好。例如，可以发送用于检索资源表示的GET请求或用于删除资源的DELETE请求。因此，我们将注意力集中在包含单词send（或类似单词）的标签的其余部分，以寻找一些额外的语义信息，这些信息可以指示标签对特定REST动词的倾向。

在检查标签是否包含发送动作或类似内容之前，我们需要确定可用于编排中的动词，以表示正在发送的消息。 表1中提供了同义词和语义上类似的发送动词。下一节将介绍我们填充此集的方式。

表1 REST动词的同义词词组和发送

|  |  |  |
| --- | --- | --- |
| 动词 | 描述 | 同义词词组 |
| POST | 在服务器上创建新资源 | SynPOSTSynPOST = {{create, request, [produce, make, ...] } |
| PUT | 编辑存在的资源 | SynPUTSynPUT = {{confirm, edit, accept, send [support, redact, ...] } |
| GET | 从服务器检索现有资源 |  |
| DELETE | 删除现有资源 |  |
| Send | 从一个编排参与者向另一个编排参与者发送消息 |  |

如果标签有一个作为SynSend元素的动作，我们将分析标签的其余部分。我们对其余部分重新应用标签注释器（来自上一小节）以标识新的操作和业务对象。例如，标签发送审阅请求，通知拒收文章，通知短纸接受被重新分析为审查请求，拒收文章和短纸接受。应用相同的批注方法将产生以下操作 - 业务对象对：请求审查;拒绝纸;接受短纸。

但是，如果没有新的行为和业务对象，我们会像处理其他任何动词一样对待这个标签。例如，在标签发送评论中，我们有α（发送评论）=发送和β（发送评论）=评论。在这种情况下，我们将发送映射到REST动词PUT以表示发送的资源处于“已发送”状态（参见3.4节）。其余部分不需要是单个单词，例如，在发送短文件的标签中，分析剩余的短文件不会产生新的业务对象。再次在这种情况下，我们将发送映射到REST动词PUT。

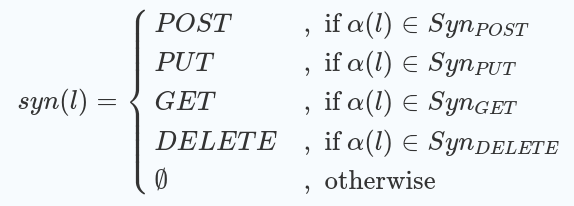
剩余分析提供了更多的语义信息，这是生成REST请求所需的，如以下两节所述。

## 3.3 通过自然语言分析派生REST动词

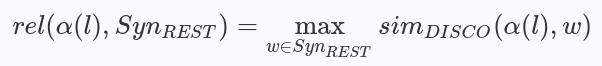
通过自然语言分析推导REST请求的总体思路是基于这样的假设，即编排任务标签提供所有相关信息。 具体而言，我们关注标签的行为，因为它们描述了必须执行哪些特定活动以及这些活动如何影响系统。 REST动词派生应用两个步骤。 第一步将相应的编排任务标签的动作与反映不同REST动词含义的同义词进行比较。 第二步涉及编排任务标签和同义词的动作的语言相似性分析，以防标签的动作与任何同义词不完全匹配。 在下文中，我们将进一步详细讨论这两个步骤。

首先，我们需要一组同义词，然后才能进行派生。一个挑战是，REST动词与特定的技术含义相关，并不一定与单词的语言含义相符。例如，REST动词POST指示服务器创建一个新的可区分资源，而要发布的动词通常描述在公告板上公布新闻的行为。因此，有必要在技术意义上定义反映POST意义的一组同义词。为此，我们向REST专家询问最接近REST动词含义的自然语言动词。这个过程的结果如表1所示。例如，专家一致认为POST的含义最好反映在动词创建或请求中。由于识别出的动词可能无法捕捉到语言的所有变化，我们进一步考虑可能从计算词典中提取的附加同义词，如WordNet [12]。例如，POST动词也可能与动词产生或制作有关。其他的例子可以用边框中的前一个表格检索。同样，我们使用相同的方法来推导表1中的发送同义词集。

同义词分析步骤调查编排任务标签的动作是否等于REST动词的同义词之一。 如果此条件评估为true，我们将相应的REST动词映射到编排任务。 否则，不会将REST动词映射到此任务。 作为一个例子，考虑编排任务创建CFP和确认短文。 第一项任务将映射到POST，因为其创建的操作是SynPOST集合的同义词的成员。 第二个任务将映射到PUT，因为它的确认操作是集合SynPUT的成员。 这个逻辑由以下函数表示，假设REST是REST动词的集合：



在同义词分析步骤未能将REST动词分配给编排任务的情况下，相似性分析步骤用作回退策略。 在这种情况下，有必要找到与动作最相关的REST动词。 因此，有必要确定一个动作与同义词的关联性。 在我们的方法中，我们使用语义相似性的概念（例如，参见[13,14,15]）来量化这种相关性。 我们利用DISCO字词相似度工具[16]的分布相似性，用simDISCO表示，因为它比现有的相似性度量更好[17]。 给定编排任务标签l，其动作α（l）和任意REST动词SynREST的同义词集合，编排任务标签和同义词REST集合的动作的相关性如下给出：



作为一个例子，我们考虑编排任务输入图1中的文章评论。由于输入的动作不是REST动词同义词集的成员，因此我们确定它与每个同义词集的相关性。 使用二阶分布相似度，我们收到以下相关性值：rel（enter，SynPOST）= 0.48，rel（enter，SynPUT）= 0.92，rel（enter，SynGET）= 0.92，rel（enter，SynDELETE）= 0.55。

最后，我们考虑所有的相关性分数来为给定的编排任务标签派生最合适的REST动词。 在这种情况下，我们假设最高关联度分数反映了给定编排任务中最适合的REST动词。 因此，我们将这个REST动词分配给最高的相关性分数。 然而，可能出现这样的情况，即多个相关性分数相等，从而导致REST动词的多于一个分配，强调用户选择正确的REST动词的必要性。 在形式上，我们描述相似性分析步骤如下：



作为一个例子，再次考虑编排任务输入文章评论及其相关性分数。由于PUT和GET的得分相等，相似性分析策略将两个REST动词PUT和GET分配给编排任务。

标签的行为不是唯一可以定义适当的REST动词的实体。我们也考虑确定的业务对象。 REST调用中PUT和POST之间的区别在于服务器处理资源的不同方式。当使用POST时，服务器将自行创建一个分配标识符的资源。但是，使用PUT时，客户端必须在请求中指定资源的标识符。例如，请求POST / paper后面跟着一个response / papers / id，其中id由服务器选择。否则，作者应该知道他或她想要修改或创建的论文的ID，当他或她请求PUT / paper / id时。

我们观察到，POST和PUT之间的这种内在差异可以分别在不定冠词和定冠词的自然语言概念中找到。在这些情况下，动作映射到PUT，我们会对英文不定冠词“a”或“an”的存在进行额外检查。如果事情变成这样，我们将这个行为映射到POST。例如，提交图1中的文章任务被映射到POST / paper，因为客户端没有引用特定的文章。

以下部分将解释如何使用各自标识的REST动词为编排任务生成RESTful请求。

## 3.4 生成REST请求

生成REST请求的任务涉及生成唯一的资源标识符（URI），以解释如何通过HTTP访问资源。为了生成URI，我们将其生成视为语言生成问题，该问题使用编排任务的可用信息和上一步中的REST动词派生。许多语言生成系统采用三步流水线方法，首先确定所需的句子信息，然后计划表达这些信息，然后将其转化为正确的句子[18]。与这些系统相比，我们不需要完全灵活的方法，因为最终的链接遵循常规结构[5]。因此，我们使用基于模板的方法[19,20,21]来生成REST URI。具体来说，我们使用编排任务和上一步中的REST动词并选择相应的链接模板。之后，我们为模板填充必要的信息，即编排任务标签的操作和业务对象。必须注意的是，有些情况下，当请求派生显示多于一个REST动词或需要用户更正链接的REST动词时，编排与其他REST链接无关。

表2 基于REST请求的链接模板

|  |  |
| --- | --- |
| 链接模板 | 例子 |
| POST /<β(l)<β(l)><<HTTP Version> | POST /CFP HTTP/1.1 |
| PUT /<β(l)><β(l)>/id/<Past Participle of α>α> <HTTP Version> | PUT /paperReview/id/entered HTTP/1.1 |
| GET /<β(l)><β(l)>/id <HTTP Version> | GET /paperReview/id HTTP/1.1 |
| DELETE /<β(l)><β(l)>/id <HTTP Version> | DELETE /shortPaper/id HTTP/1.1 |

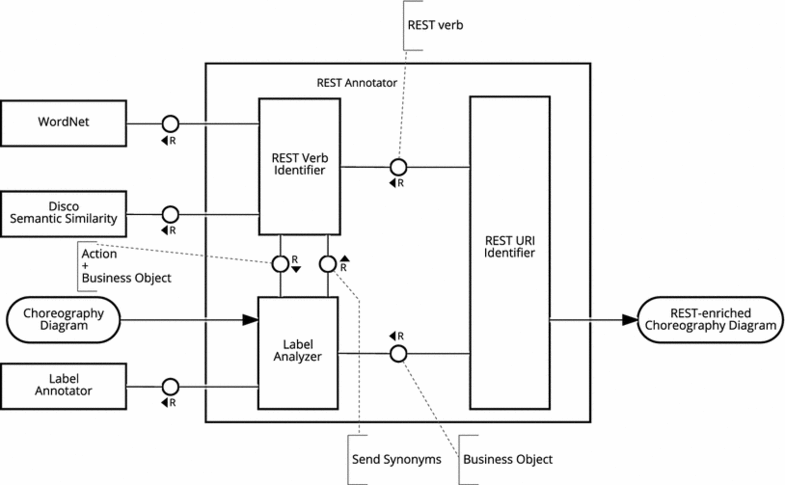


图4 REST Annotator软件体系结构

表2显示了不同REST动词的链接模板，并给出了从图1的编排任务创建的示例。模板强调编排任务标签（β（l））的业务对象对REST链接扮演着重要角色，它类似于需要由REST动词解决的服务器资源。因此，我们将业务对象与唯一标识符关联在一起。如果特定资源的状态必须更改，链接还会解释其状态如何随REST动词而变化。这个变化是由一个编排任务标签的过去分词表达的。

我们的方法的最终输出是一个RESTful编排。图3是一个模型，描绘了将我们的方法应用到运行示例中生成的RESTful编排图的摘录。在这个图中，我们展示了REST工程师如何与生成的RESTful编排进行交互。 REST工程师提供所有四个生成的链接（每个REST动词一个），根据匹配得分（1为最好，0为差）进行排名。根据选择，假定交互始终有效,HTTP响应会自动生成。

# 4 评估

本节介绍我们的评估。 首先，我们解释我们原型实现的架构。 然后，我们从实践中呈现一组172个编排图的推导步骤的准确性结果。

## 4.1 评估配置

为了评估我们的方法，我们开发了一个名为REST Annotator的工具。 REST Annotator的体系结构在图4中被描述为FMC图[22]。 REST Annotator将一组编排图作为输入，并输出一组REST丰富的编排图。 该工具使用三个外部组件：Leopold等人的标签注释器。 [11]，WordNet [12]以及DISCO工具[16]的分布相似性组件。 构成该工具的主要组件由三个子组件组成：标签分析器，REST动词标识符和REST URI标识符。

标签分析器负责从模型中提取所有标签，并在标签注释器的帮助下对其进行分析。 后者用于表示编排任务标签的操作和业务对象。 标签分析器将动作和每个标签的业务对象映射到REST动词标识符和REST URI标识符组件。 REST动词标识符组件需要由标签分析器提供的操作以及WordNet的同义词，类似于各自的REST动词。 如果未找到同义词，则该组件需要动作和来自Disco语义相似性组件的REST动词的同义词集之间的语义相似性分数。 一旦识别出动作与每个REST动词的语义关系，就将REST动词及其各自的分数传递给REST URI标识符组件。 这个组件产生一组充满REST注解的编排图。

此外，标签分析器在提供最终操作和业务对象之前，需要应用3.2节中所述的方法。 为此，它需要SynSend，它由REST动词标识符生成。 另外，后者需要前者的业务对象来检查是否存在不定冠词。 正如在3.3节中解释的那样，不定冠词的存在可以改变REST动词标识符的输出。

作为评估数据，我们使用BPM 学者倡导的编排图。 该计划提供了来自不同领域的丰富的流程模型。 总的来说，我们检索了424个BPMN编排图。 由于这些图表是由专家和非专家一起创建的，因此有必要清理数据。 我们采用以下清理标准：

1.仅限英语的图表我们仅包含带英文文字标签的图表。 这个标准是必要的，因为大多数自然语言分析组件仅支持英语。

1. 语法正确的图表排除具有关于BPMN 2.0编排图规范的语法错误的图表。

关于评估程序，REST专家必须对每个编排任务进行三步评估：（a）标签的语法正确性，（b）REST动词生成的适当性，以及（c） 生成的REST URI的适应性。 在（a）成立的情况下，评估者还必须检查具有最佳匹配分数的识别的REST动词是否适当。 如果（b）成立，评估者必须检查生成的URI是否合适。

然而，评估有其自身的局限性。 单一评估者的存在可能会在确定正确匹配时施加主观性。 为了缓解这个问题，评估人员重复了几次检查程序，同时参考开发RESTful API的常见做法。 这样可以避免像相同标签 - RESTful请求对的不一致验证等错误，并有助于解决非明显的错误配对。

## 4.2 评估结果

本节讨论表3中汇总的结果。172个模型总共包含1213个编排任务标签。 模型大小（根据编排任务计数）为1到26，平均每个模型约7个编排任务。 从这些标签中，864个标签（71.14％）在语法上是正确的标签。 在下面的讨论中，我们只关注那些句法正确的标签，并讨论动词识别和链接生成在这些情况下如何执行。

动词识别策略在772个标签中识别出正确的REST动词，相当于所有句法正确标签的89.35％。在这些标签中，我们进一步区分已经用同义词策略和相似策略识别的动词。同义词策略能够在332个标签中导出正确的REST动词，而相似性策略则为450个编排标签导出正确的REST动词。结果强调了对REST动词相似性识别策略的需求。最明显的REST动词是PUT，因为它是在参与者改变业务对象状态的编排上下文中预期的，例如，订单被发送，接受，交付和支付。 POST发现139例，其中8例使用不定冠词。 DELETE标识的数量较少也反映了在REST环境中使用DELETE的罕见情况，这是由于资源通常存档或保存在特定状态而不是故意删除。

表3 用户评估的定量结果

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

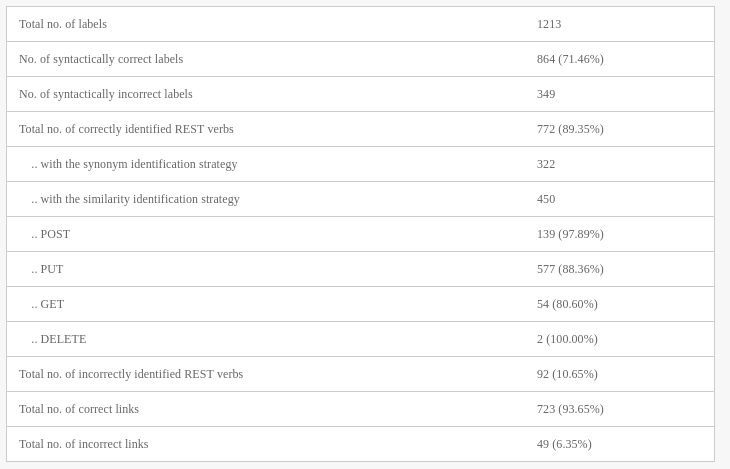
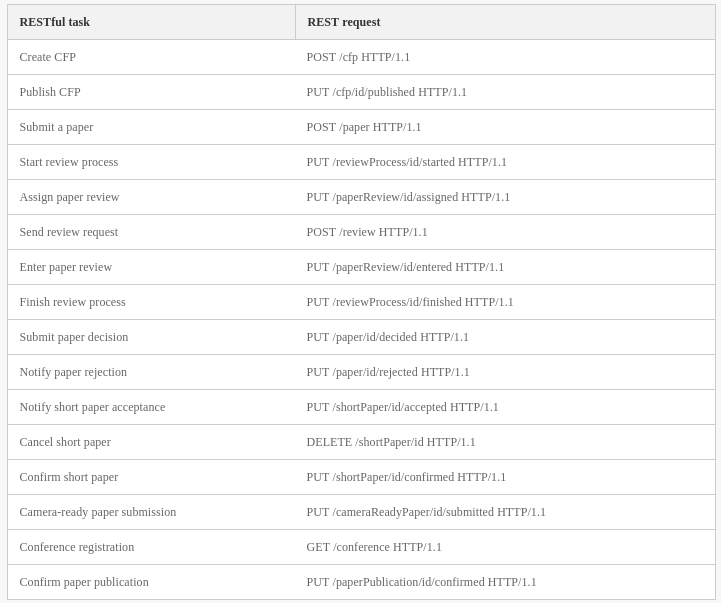


表4 来自图1的相应RESTful任务的REST请求结果



总共有92个编排标签（10.65％）被标注了错误的REST动词。我们观察到GET被检测到最少，DELETE总是被检测到。我们确定了两类可能导致错误注释的错误，其中第一类错误在本示例集的上下文中已修复，并且不计入错误的REST谓词标识。该第一类包含编排标签，其中相似性策略显示两个或更多个相等的相似性分数。 101个标签就是这种情况。在识别出这个特定样本集的这些REST模糊行为的列表后，REST专家被要求选择最合适的映射。以下不完全清单是消除歧义的：{start-PUT, pay-PUT, invoice-PUT, article-PUT, enter-PUT, publish-PUT, allocate-PUT, explain-PUT, disburse-PUT, receipt-PUT, show-GET, book-PUT}。这个列表可以用在语义相似性方法中得分相同的动词进一步丰富。

第二类涵盖了我们的方法识别错误动词的情况。 REST评估揭示了92个编排标签，我们的方法没有找到正确的REST动词。 这些情况必须由用户纠正。

生成RESTful链接的方法在772个正确的动词标识中创建了723个正确的和49个不正确的链接。我们将标签质量确定为不正确链接的主要原因。例如，我们发现通过参考特定状态未正确指定编排任务，例如确认付款并将发票发送到PUT /payment/id/confirmeded HTTP/1.1 和PUT /invoice/id/sented HTTP/1.1，分别。为标签确认付款/付款确认和发送发票生成正确的结果。错误链接代的另一个原因是业务对象的错误识别。例如，标签出货物品被标记为α (ship article) = article (action) and β (ship article) = ship (business object)。正确的标签是将船舶识别为行为和物品作为业务对象。尽管如此，我们的结论是，链接生成工作令人满意，并产生大量正确的REST链接。

我们还通过将我们的方法应用于图1中的示例性编排图来举例说明我们评估的结果。表4显示了针对相应编排任务生成的REST请求。

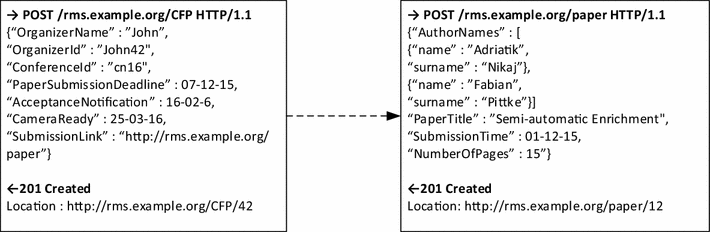


图5 一个RMS实现的具体框架实例

## 4.3 讨论

定量评估结果中出现了三个主要观察结果。第一个观察与使用REST URI的编排任务的正确注释有关。例如，它将PUT标识为确认短文任务的正确REST动词，并生成URI PUT / shortPaper / id / confirmed。然而，我们也遇到了一些问题，其中该方法为REST动词检索多个可能性并且未能为一个特定的REST动词做出决定。在这个例子中，编排任务进入文章审查属于这个组。该方法识别REST动词PUT和GET，因为输入的动作不是任何REST动词同义词列表的成员，并且两个REST动词的语义相似度得分相等。基于此结果，链接生成器组件创建两个可能的链接，其中用户必须选择。尽管如此，链接本身已被正确创建。正如前一节所提到的，我们通过对REST动词映射进行消歧来解决这个特定测试集的问题。但是，歧义动词的列表并不完整，因为其他动词可能不是我们评估中使用的标签的一部分。当应用我们的方法来获得包含这些动词的编排标签的更好结果时，该列表可以用作输入。

第三个观察涵盖了REST请求不正确，需要用户手动更正。例如，考虑编排任务会议注册，为此我们的方法创建一个GET链接。但是，我们会期待POST或PUT请求。此类型的不正确链接可能有多个错误来源。一方面，标签注释器组件（参见图4）可能将编排任务错误分类，并错误地更改了操作和业务对象。另一方面，REST动词识别组件可能导致了错误，因为该动作是同义词单词列表的直接成员，或者其与其他REST动词之一的同义词的相似度得分最高。在我们的例子中，前者适用。 REST动词GET已被识别，因为要注册的动作是要读取的WordNet同义词，因此也是同义词词组SynGET的成员。因此，目前还没有考虑其他替代方案，最终要求用户更正此REST请求。

最后，图5描述了RMS RESTful交互的具体实例。粗体部分和REST交互的顺序由REST Annotator工具生成，并作为开发RESTful API的框架提供给开发人员。在RSM上下文中，两个矩形分别表示创建CFP的具体实例，并提交图3中的文章编排任务。虚线箭头表示第二个实例只能在第一个实例执行后才能执行。对于给定的RESTful编排，可以为提供RESTful API的每个参与者派生骨架图，例如会议组织者的RMS移动应用程序，该应用程序从RMS接收有关审阅过程状态的通知。因此，我们从全局编排视图跳转到至少一个仅着重于REST行为界面的编排视图，即REST请求和响应在单个参与者应用程序内执行的顺序。应用我们方法的好处在于，所有参与者都使用相同的URI生成逻辑，这有助于更好地理解，维护和发展REST API [23]。从RESTful编排导出骨架的自动化将留作未来工作。

# 5 相关研究

我们确定了与我们的方法相关的三大类研究。首先，我们的方法与模型驱动方法有关，这些方法专注于设计和设计REST API或RESTful服务的过程。例子包括Valverde和Pastor [24]或Schreier [25]的工作，他们通过提供元模型来支持这一过程。虽然前一个元模型侧重于REST服务的规范和机器可读规范的生成，但后一种方法解决了REST应用程序的形式方面问题，如应用程序结构和行为。 Laikorpi等人[26]将RESTful API的设计视为模型转换问题，并描述开发RESTful服务所需的转换和中间模型。我们的方法通过以半自动方式从编排图中获取REST信息来为模型驱动的方法做出贡献。与这些方法相反，我们的方法基于BPMN编排标准，该标准从全局的角度指定业务交互，以获得具有实现细节的REST骨架。

其次，我们的方法涉及到缩小业务流程编排与其基础编排系统之间的差距。在这方面，Decker et al。 [27]提出了扩展BPEL Web服务组合标准[28]，以缩小编排和编排之间的差距。 BPEL4Chor扩展的目标是通过集成现有的BPEL服务编排来编排流程编排。 BPEL4Chor是一种自下而上的方法，它基于SOAP和WSDL等Web服务标准[29]。与此相反，我们采用自顶向下的方法来获得REST式交互。另一种方法建立了BPMN和REST之间的关系[9]。作者建议，业务流程的一部分本身可以作为REST资源发布。虽然这种方法侧重于参与REST式交互的参与者的内部行为，但我们专注于全局角度，它允许推理在实现级别允许的交互。此外，这项工作的附加价值在于提供了一种半自动的方法来从原始业务流程编排中派生RESTful编排。

第三，我们的方法是业务流程到执行转换的更广泛环境的一部分。在[30]中，Mendling等人展示了如何从编排的每个参与者的全局WS-CDL [31]模型中导出BPEL流程定义。而且，对于某些块来说，这个推导是完全自动化的，对于那些其上下文起重要作用的块来说，这个推导是半自动化的。同样，欧阳等人。在[32]中提出了一组将BPMN模型转换为BPEL的技术。自动翻译不会对源BPMN模型施加结构限制，目标模型是可读的BPEL代码。从其他基于图形的流程建模语言转换到BPEL也存在[33,34]。与之前的工作方向一致，Weber等人在[35]中提出了一种新方法，该方法利用新型区块链技术来实现业务流程协作。区块链提供了一个全球计算基础架构，可以运行被称为智能合约的程序[36]。在[35]中，智能合约是从流程规范中派生出来的，并被部署到块链中以执行流程协作。但是，这些方法都没有使用自然语言处理来导出执行工件。因此，本文以这种方式提供了独特的贡献。

# 6 结论,不足与未来工作

本文定义了一种从BPMN编排图中派生RESTful编排的半自动方法。 所提出的方法基于自然语言分析技术来为交互派生最合适的REST动词并为派生的REST动词生成REST URI。 考虑到编排特定的标签风格。 我们的方法通过开发REST Annotator工具并将其应用于来自不同领域的编排图来评估。 该工具的输出由REST专家评估。 89.35％的测试用例中动词识别正确，93.65％的用例中URI正确。 这项工作为业务流程编排与其实施之间的研究差距迈出了一步。

我们的方法也存在局限性，这些局限性基于自然语言的不精确性和所使用的语言处理工具的能力。 这种不精确性是几个错误识别的REST动词和REST URI的重要原因，REST专家必须纠正这些错误和REST URI。 特定的限制与标签样式有关。 如果使用太多名词，很难确定预期的行为和业务对象。 例如，标签application letter submission将产生PUT letter/applied，而不是更优选的PUT applicationLetter /提交。

在未来的工作中，我们计划通过利用词义消歧技术和编排图的行为方面来解决这些限制。词义消歧利用外部知识库，如WordNet [12]或BabelNet [37]连同语境信息或言语行为[38,39]，以识别单词的正确解释。它的有用性已经在[40]中对过程模型进行了研究。行为方面与编排任务的顺序有关[41]。事实上，只有特定的序列和消息组合才有意义，可用于描述限制潜在解释数量的约束[42]。例如，如果已确定POST和GET请求，并且相应的编排任务处于交互的开始阶段，那么它更可能是POST请求。通过这种方式，我们旨在提高所提出方法的准确性。此外，这种方法不考虑消息及其标签。包含它们可能会导致URI生成准确性的增加，因为这些消息描述了传递给接收者的业务对象。

# 引用

1.OMG: Business Process Model and Notation (BPMN), Version 2.0. http://www.omg.org/spec/BPMN/2.0/ (2011)

2.Fielding, R.T.: Architectural styles and the design of network-based software architectures. Ph.D. thesis (2000)

Google Scholar

3.Massé, M.: REST API Design Rulebook. O’Reilly Media Inc., Newton (2012)

Google Scholar

4.Nikaj, A., Pittke, F., Weske, M., Mendling, J.: Semi-automatic derivation of RESTful interactions from choreography diagrams. In: Schmidt, R., Guédria, W., Bider, I., Guerreiro, S. (eds) Enterprise, Business-Process and Information Systems Modeling: Proceedings of the 17th International Conference, BPMDS 2016, 21st International Conference, EMMSAD 2016, Held at CAiSE 2016, Ljubljana, Slovenia, June 13–14, pp. 141–156. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39429-9\_10

5.Nikaj, A., Mandal, S., Pautasso, C., Weske, M.: From choreography diagrams to RESTful interactions. In: Norta, A., Gaaloul, W., Gangadharan, G.R., Dam, H.K. (eds) Service-Oriented Computing – ICSOC 2015 Workshops: WESOA, RMSOC, ISC, DISCO, WESE, BSCI, FORMOVES, Goa, India, Nov. 16-19, 2015, Revised Selected Papers, pp. 3–14. Springer, Berlin, Heidelberg (2016). https://doi.org/10.1007/978-3-662-50539-7\_1

6.Nikaj, A., Batoulis, K., Weske, M.: Rest-enabled decision making in business process choreographies. In: International Conference on Service-Oriented Computing, pp. 547–554. Springer (2016)

Google Scholar

7.Nikaj, A., Weske, M.: Formal Specification of RESTful Choreography Properties. In: 16th International Conference on Web Engineering, ICWE 2016, Lugano, Switzerland, June 6–9, 2016. Springer (2016)

Google Scholar

8.Pautasso, C., Wilde, E.: Push-enabling RESTful business processes. In: Kappel, G., Maamar, Z., Motahari-Nezhad, H.R. (eds.) Service-Oriented Computing: Proceedings of the 9th International Conference, ICSOC 2011, Paphos, Cyprus, Dec. 5-8, pp. 32–46. Springer, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25535-9\_3

9.Pautasso, C.: BPMN for REST. In: Proceedings of the 3rd International Business Process Modeling Notation Workshop (BPMN 2011), pp. 74–87 (2011)

Google Scholar

10.Mendling, J., Reijers, H.A., Recker, J.: Activity labeling in process modeling: empirical insights and recommendations. Inf. Syst. 35(4), 467–482 (2010)

CrossRefGoogle Scholar

11.Leopold, H., Eid-Sabbagh, R., Mendling, J., Azevedo, L.G., Baião, F.A.: Detection of naming convention violations in process models for different languages. Decis. Support Syst. 56, 310–325 (2013)

CrossRefGoogle Scholar

12.Miller, G.A.: WordNet: a lexical database for english. Commun. ACM 38(11), 39–41 (1995)

CrossRefGoogle Scholar

13.Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, pp. 133–138 (1994)

Google Scholar

14.Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, pp. 448–453 (1995)

Google Scholar

15.Lin, D.: An information-theoretic definition of similarity. ICML 98, 296–304 (1998)

Google Scholar

16.Kolb, P.: Disco: a multilingual database of distributionally similar words. In: Proceedings of KONVENS-2008, Berlin (2008)

Google Scholar

17.Kolb, P.: Experiments on the difference between semantic similarity and relatedness. In: Proceedings of the 17th Nordic Conference on Computer Linguistics (2009)

Google Scholar

18.Reiter, E., Dale, R.: Building applied natural language generation systems. Nat. Lang. Eng. 3(1), 57–87 (1997)

CrossRefGoogle Scholar

19.Denger, C., Berry, D.M., Kamsties, E.: Higher quality requirements specifications through natural language patterns. In: IEEE International Conference on Software—Science, Technology and Engineering, pp. 80–90 (2003)

Google Scholar

20.Leopold, H., Mendling, J., Polyvyanyy, A.: Generating natural language texts from business process models. In: Proceedings of the 24th International Conference on Advanced Information Systems Engineering, pp. 64–79 (2012)

Google Scholar

21.Leopold, H., Mendling, J., Polyvyanyy, A.: Supporting process model validation through natural language generation. IEEE Trans. Softw. Eng. 40(8), 818–840 (2014)

CrossRefGoogle Scholar

22.Knöpfel, A., Gröne, B., Tabeling, P.: Fundamental modeling concepts. Effective Communication of IT Systems, Wiley, England (2005)

Google Scholar

23.Palma, F., Gonzalez-Huerta, J., Moha, N., Guéhéneuc, Y.G., Tremblay, G.: Are restful apis well-designed? Detection of their linguistic (anti)patterns. In: Service-Oriented Computing. Lecture Notes in Computer Science. Springer (2015)

Google Scholar

24.Valverde, F., Pastor, O.: Dealing with rest services in model-driven web engineering methods. V Jornadas Científico-Técnicas en Servicios Web y SOA, JSWEB (2009)

Google Scholar

25.Schreier, S.: Modeling restful applications. In: Proceedings of the Second International Workshop on Restful Design, pp. 15–21. ACM (2011)

Google Scholar

26.Laitkorpi, M., Selonen, P.: Towards a model-driven process for designing restful web services. In: IEEE International Conference on Web Services, pp. 173–180. IEEE (2009)

Google Scholar

27.Decker, G., Kopp, O., Leymann, F., Weske, M.: Bpel4chor: extending bpel for modeling choreographies. IEEE Int. Conf. Web Serv. 2007, 296–303 (2007)

Google Scholar

28.Jordan, D., Evdemon, J., Alves, A., Arkin, A., Askary, S., Barreto, C., Bloch, B., Curbera, F., Ford, M., Goland, Y., et al.: Web services business process execution language version 2.0. OASIS Stand. 11, 1–10 (2007)

Google Scholar

29.Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services. Springer, Berlin (2004)

CrossRefMATHGoogle Scholar

30.Mendling, J., Hafner, M.: From WS-CDL choreography to BPEL process orchestration. J. Enterp. Inf. Manag. (JEIM) 21, 506–515 (2008)

Google Scholar

31.Kavantzas, N.: Web services choreography description language (ws-cdf) version 1.0. http://www.w3.org/TR/ws-cdl-10/ (2004)

32.Ouyang, C., Dumas, M., Van Der Aalst, W.M.P., Ter Hofstede, A.H.M., Mendling, J.: From business process models to process-oriented software systems. ACM Trans. Softw. Eng. Methodol. 19(1), 2–37 (2009)

CrossRefGoogle Scholar

33.Ziemann, J., Mendling, J.: EPC-based modelling of BPEL processes: a pragmatic transformation approach. In: International Conference on Modern Information Technology in the Innovation Processes of the Industrial Enterprises, Genova, Italy (2005)

Google Scholar

34.Mendling, J., Lassen, K.B., Zdun, U.: On the transformation of control flow between block-oriented and graph-oriented process modelling languages. IJBPIM 3(2), 96–108 (2008)

CrossRefGoogle Scholar

35.Weber, I., Xu, X., Riveret, R., Governatori, G., Ponomarev, A., Mendling, J.: Untrusted business process monitoring and execution using blockchain. In: La Rosa, M., Loos, P., Pastor, O. (eds.) Business Process Management: Proceedings of the 14th International Conference, BPM 2016, Rio de Janeiro, Brazil, Sept. 18–22, pp. 329–347. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4\_19

36.Omohundro, S.: Cryptocurrencies, smart contracts, and artificial intelligence. AI Matters 1(2), 19–21 (2014)

MathSciNetCrossRefGoogle Scholar

37.Navigli, R., Ponzetto, S.P.: Babelnet: the automatic construction, evaluation and application of a wide-coverage multilingual semantic network. Artif. Intell. 193, 217–250 (2012)

Google Scholar

38.Medina-Mora, R., Winograd, T., Flores, R., Flores, F.: The action workflow approach to workflow management technology. In: Proceedings of the 1992 ACM conference on Computer-supported cooperative work, pp. 281–288. ACM (1992)

Google Scholar

39.Cohen, W.W., Carvalho, V.R., Mitchell, T.M.: Learning to classify email into "speech acts". EMNLP 4, 309–316 (2004)

Google Scholar

40.Pittke, F., Leopold, H., Mendling, J.: Automatic detection and resolution of lexical ambiguity in process models. IEEE Trans. Softw. Eng. 41(6), 526–544 (2015)

CrossRefGoogle Scholar

41.Weidlich, M., Mendling, J., Weske, M.: Efficient consistency measurement based on behavioral profiles of process models. IEEE Trans. Softw. Eng. 37(3), 410–429 (2011)

CrossRefGoogle Scholar

42.Leopold, H., Niepert, M., Weidlich, M., Mendling, J., Dijkman, R., Stuckenschmidt, H.: Probabilistic optimization of semantic process model matching. Bus. Process Manag. 7481, 319–334 (2012)

Google Scholar