编排图驱动的区块链业务过程管理框架

俞东进，韦懿杰，孙笑笑，倪可，沈沪军
（杭州电子科技大学计算机学院，浙江 杭州 310018）

摘 要：针对现有基于区块链的业务过程管理系统中过程实例化成本较高、版本迭代困难等问题，提出了一种编排图驱动的区块链业务过程管理框架。该框架包含一个可用于存储业务过程编排元模型、模型部件演化版本和实例执行状态的通用智能合约，其通过延迟模型元素实例化时机和集成多过程实例，可大幅降低区块链上过程模型实例化成本。同时，该框架引入基于模型数据复用和投票机制的版本控制方法，使其能够在单个智能合约中创建不同版本编排模型的过程实例。通过一个真实案例验证了该框架在分布式业务过程管理中的有效性。

关键词：编排图；业务过程管理；区块链；智能合约；版本控制
中图分类号：TP302
文献标识码：A
DOI: 10.11959/j.issn.1000−436x.2021171

Choreography-driven business process management framework based on blockchain

YU Dongjin, WEI Yijie, SUN Xiaoxiao, NI Ke, SHEN Hujun
School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018,China

Abstract: Aiming at the problems of high process instantiation cost and difficult version iteration in the current blockchain-based business process management systems, a choreography-driven blockchain business process management framework was proposed. The framework included a general smart contract that could be used to store business process choreography meta-models, model component evolution versions, and instance execution status. The cost of process model instantiation on the blockchain was significantly reduced by delaying model elements instantiation timing and integrating multiple process instances. At the same time, a version control method based on model data reuse and voting mechanism was introduced, so that process instances of different versions of orchestration models could be created in a single smart contract. A real case validates the effectiveness of the framework in distributed business process management.

Keywords: choreography, business process management, blockchain, smart contract, version control

1 引言

传统的业务过程管理技术为企业提供了设计、记录和执行其业务过程的功能。然而，随着业务变得越来越复杂，许多过程不再局限于单个组织。这种跨组织业务过程的出现给传统的业务过程管理技术带来了许多挑战，其中最突出的是如何选择负责过程管理的中央控制者，即业务过程由谁来进行主导。由于中央控制者利用自身权限谋取私利的现象难以避免，传统分布式跨组织业务过程系统只有在组织之间相互信任或者求助于第三方权威机构的情况下才能够保证组织间交互消息的可靠性。最

收稿日期：2021−03−08；**修回日期**：2021−05−16
基金项目：国家自然科学基金资助项目（No.61702144）；工信部工业互联网创新发展工程基金资助项目（No.TC200802G, No.TC2008033）；浙江省重点研发计划基金资助项目(No.2020C01165)；浙江省自然科学基金资助项目（No.LQ20F020017）
Foundation Items: The National Natural Science Foundation of China (No.61702144), Industrial Internet Innovation and Development Project of Ministry of Industry and Information Technology (No.TC200802G, No.TC2008033), The Key Research and Development Program of Zhejiang Province (No.2020C01165), The Natural Science Foundation of Zhejiang Province (No.LQ20F020017)

新的研究已经意识到了这一问题并且建议使用区块链技术来解决上述信任缺失问题[1]。

目前已经有许多研究者在基于区块链的业务过程管理领域提出了相关的研究方法与实现原型，这些工作的重点在于过程模型的执行控制[2-6]。但是，它们一般都是通过将过程模型硬编码为区块链智能合约来支持过程实例的链上执行，而无法适应业务过程随时发生变化的实际需求[7]，由此造成了过程实例化成本过高、版本迭代困难等问题。

针对上述问题，本文提出了一种编排图驱动的区块链业务过程管理框架，并给出了相关的以太坊智能合约的原型实现，最后使用助听器行业的一个真实案例来评估本文框架的有效性。

与现有方法相比，本文具有以下创新之处。

1) 设计了包括业务过程编排元模型、模型部件版本和实例执行状态的通用智能合约。与该领域其他方法生成适应于特定业务过程的智能合约不同，本文框架采用通用智能合约设计，将编排图中包含的模型部件部署到智能合约的编排元模型内，同时在创建过程实例时采用集成多过程实例和延迟元素实例化时机的方法，大幅降低创建过程实例成本开销。

2) 提供了一种基于投票机制和复用模型数据的版本控制方法，以实现基于区块链的业务过程模型版本管理。该方法通过引入投票机制提高了版本控制的去中心化程度。

2　预备知识

2.1　业务过程建模标注与编排图

业务过程建模标注（BPMN，business process modeling notation）是目前得到广泛认同和使用的建模语言[8]。BPMN 中包含编制图、协作图和编排图，其中，协作图和编排图都可被用于描述跨组织业务过程的执行过程。然而，协作图没有以简洁的方式描述编排过程；编排图从组织间内部编排细节中抽象而出，在一个以交互为中心的级别上描述业务过程，能够在不暴露参与者内部行为的基础上描述参与者之间的交互逻辑。这些特性适用于基于区块链的跨组织业务过程建模。

图 1 描述了编排图常用的建模元素，包括开始事件、结束事件、排他网关、并行网关、事件网关、编排任务和序列流。其中，开始事件和结束事件分别表示编排的开始和结束。各类网关用于控制编排中序列流的走向，其中排他网关和并行网关又可各自分为分离网关和合并网关。编排任务用于描述 2 个参与者之间的消息交互，在图 1 中表示为 3 个区域，中间为任务名称，其余 2 个为任务参与者。编排任务可以根据携带的消息数量分为单向任务和双向任务。最后，序列流用于连接编排元素，控制业务过程的执行。

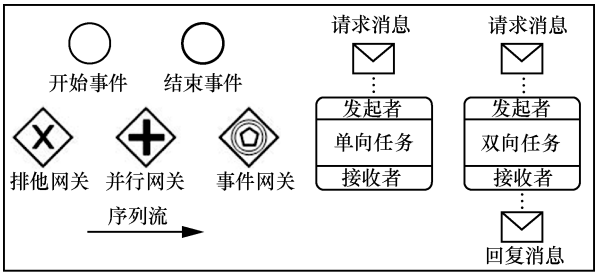


图 1　编排图常用的建模元素

2.2　区块链和智能合约

区块链最初源自比特币的底层技术，随后被用于支持不受信任的用户之间的价值传输。以太坊是第一个以智能合约为基础的可编程非许可链开源平台项目，支持使用区块链网络构建分布式应用[9]。智能合约的想法最早由 Szabo[10]提出，其被定义为一套数字形式的承诺，并由合约参与方遵守执行，但最初由于其环境和应用场景的缺失并未得到广泛关注。在区块链技术出现后，智能合约概念再次被提出。文献[11]指出，智能合约由于其去中心化、确定性、可观察等特点在金融交易、供应链等方面有着广阔的应用前景。以太坊通过图灵完备脚本语言 Solidity 实现复杂的程序逻辑，即智能合约。用户能够根据个人需求自定义智能合约逻辑，并且将智能合约部署到以太坊区块链上。

3　相关工作

近年来，区块链由于其去中心化、不可篡改和可追溯等特性得到广泛应用。这些特性尤其适用于跨组织业务过程消息交互场景，从而解决组织间的信任缺失问题。目前，许多研究者已经将区块链技术应用到了业务过程领域，并取得了不少成果，相关工作主要围绕应用理论、BPMN 编排图、BPMN 过程图、BPMN 协作图和场景应用等展开。

Mendling 等[1]提出在传统的业务过程背景下利用区块链技术的挑战和机遇，指出新兴区块链技术有可能彻底改变现有的组织间业务过程执行方式。

Viriyasitavat 等[12]提出了一个基于区块链的业务流程架构，以克服传统业务流程执行过程中的时间不一致和共识偏差问题。Ladleif 等[13]分析了业务过程中外部数据交互模式在区块链上的语义支持，扩展了区块链 Oracle 的实现策略。Köpke 等[14]分析了智能合约在业务过程管理中体现的可观察性、可执行性和隐私性，并介绍了区块链上执行业务过程涉及的密钥交换模式。尽管这些工作为后续研究指明了方向，但这些工作只停留在概念层面，缺乏相应的案例和原型来支持其理论。

Weber 等[2]提出了将区块链技术集成到过程编排中的方法，并为其实现了相关功能组件。该方法能够有效支持编排过程的链上执行，但其智能合约一旦部署就无法修改，使用工厂合约来创建过程实例智能合约的方式也额外增加了以太坊的成本开销。García-Bañuelos 等[3]针对该问题通过简化过程模型为 Petri 网以及在智能合约中合理利用位向量来优化合约部署和执行成本开销，最后降低了以太坊智能合约部署和过程实例执行的 gas 消耗，但该方法难以支持业务过程模型的版本迭代。Ladleif 等[4]根据区块链特性提出了区块链编排图的操作语义，通过智能合约全局变量来支持链上编排业务过程的执行，并且使用 3 个案例原型对其工作进行评估。Corradini 等[5]提出了一种基于区块链技术和编排图模型驱动的业务流程管理框架。该框架将输入的 BPMN 编排图模型转化为特定的智能合约，并且为编排的整个生命周期提供支持，但存在智能合约缺少灵活性的问题。同时，该框架还存在为每个实例生成的智能合约中有大量冗余函数代码、对于消息的支持停留在全局变量之上、缺乏消息的隐私控制机制等缺点。Lichtenstein 等[6]针对区块链数据存储的开销问题，通过不同的编排过程复用相同的编排消息数据，在不改变数据完整性的情况下降低区块链数据存储成本。

López-Pintado 等[15]提出了 Caterpillar，这是全球第一个开源区块链业务过程执行引擎。该引擎基于 BPMN 过程图开发。与其他实现原型相比，Caterpillar 支持更多类型的 BPMN 元素，允许通过工厂合约创建过程实例智能合约并且支持跟踪过程实例的执行状态，但在执行过程中忽略了组织间的消息交互内容，将跨组织协作业务过程视为组织内部业务过程。此外，López-Pintado 等[16]还提出了一种用于协作业务过程执行的动态绑定模型和相应的绑定策略规范语言，同时将该规范语言赋予 Petri 网语义来检测绑定策略是否会发生死锁，最后将相关工作内容集成到 Caterpillar 框架中，但动态绑定模型的引入增加了角色授权的复杂性，增大了在智能合约中进行访问控制的成本开销。为了解决智能合约缺乏灵活性问题，López-Pintado 等[7]还在 Caterpillar 框架中加入基于动态数据结构的 BPMN 过程模型解释器。该工作与本文类似，但没有考虑模型改动后导致的版本变动问题，即缺乏版本控制与修改控制机制。

Sturm 等[17]基于 BPMN 协作图提供了一个包含通用数据结构的智能合约，该合约支持存储过程模型和控制过程实例正确执行，但该文只关注业务过程的控制流逻辑，不支持创建多个过程实例。此外，Sturm 等[18]在其提出的智能合约中引入全局数据变量来支持基于数据的决策。Klinger 等[19]同样基于协作图提出了区块链业务过程执行框架，但其工作重点放在过程模型部署和部署成本的平均分配上。

还有一些研究者针对不同场景下的区块链业务流程管理提出相应的方法。Madsen 等[20]提出了一种在对抗环境下执行分布式声明性工作过程的方法，该方法基于动态条件响应（DCR, dynamic condition response）图解决对抗环境下组织间缺乏受信任第三方的问题。其中，DCR 图是一种基于事件的声明性过程模型描述，用于指定一个预先约定的工作流。Haarmann 等[21]等以决策模型和符号为基础，提出了一种在区块链上存储过程决策信息的方法，并使用概念验证原型对该方法进行了评估。为了适应决策信息包含敏感数据的实际场景，Haarmann 等[22]提出了一种不需要揭示敏感数据来支持决策的方法，允许参与者对决策结果提出质疑，通过揭示决策结果来解决相应的冲突并对恶意质疑者进行惩罚。Ladleif 等[23]提出了一种多链业务过程管理架构，针对不同级别的风险容忍度和保密性的实践需求可支持多链环境下的业务过程编排。

上述工作的重点在于过程模型中序列流的执行控制，相关的大多数实现框架和原型将业务过程模型硬编码为特定的以太坊智能合约来实现，这种方式忽略了以太坊智能合约一旦部署就无法进行修改的特性。在业务过程需要进行修改或者智能合约出现错误时会造成大量的以太坊 gas 资源浪费。少数原型考虑了这一问题并且采用通用智能合约

和通用数据结构的设计方式来解决灵活性低下问题，但对于模型实例如何集成以及模型修改如何控制还缺乏完整的设计方案。与现有方法不同，本文使用通用智能合约来控制跨组织业务过程的执行过程，在保持灵活性的基础上使用引入投票机制的版本控制方法，即利用群体智慧管理业务过程模型的修改和更新。

4　区块链业务过程管理框架

4.1　框架设计与概念定义

本节从模型过程控制、访问控制策略、模型版本控制和隐私控制策略 4 个角度分析区块链业务过程管理框架设计方案，并给出框架内部相关概念的定义说明。链上智能合约结构如图 2 所示。

1) 模型过程控制

本文所提框架将编排图模型中相应的模型数据存储于以太坊智能合约中，以控制区块链上过程实例的正确执行。

定义 1　业务过程编排元模型。业务过程编排元模型 M 包含所有编排模型中可能存在的元素、消息、决策三类模型部件及其版本号，即 $M=\{<\text{ME}|\text{MM}|\text{MD},\text{VID}>\}$，其中，ME 表示编排模型中的元素，包含开始事件、结束事件、网关和编排任务等模型元素；MM 表示编排模型中包含的交互消息；MD 表示编排模型中包含的决策条件；VID 表示相应模型部件对应的版本号。

定义 2　编排合约。编排合约 CC 是一个存储 M、V 和 I 的智能合约，即 $\text{CC}=\{M,V,I\}$，其中，模型部件版本 $V=\{<\text{ME}_i,\text{VID}_i>\}\cup\{<\text{MM}_j,\text{VID}_j>\}\cup\{<\text{MD}_k,\text{VID}_k>\}$ 表示任取相关模型部件的某一版本所组成的模型部件集合；实例执行状态 $I=<\text{IE},\text{IM},\text{IG}>$ 由实例元素 IE、实例执行消息内容 IM、实例全局数据存储 IG 组成。

本文提出的编排合约 CC 内含有表示业务过程编排元模型 M 的数据结构体。过程实例的执行受对应的模型部件控制，具体解析和执行过程将在执行步骤中详细说明。

2) 访问控制策略

当使用以太坊区块链来协作跨越组织业务过程的执行时，需要有授权机制来限制对于智能合约的访问。为此本文在合约框架中引入了基于角色访问控制方法的访问控制智能合约。

定义 3　编排过程组织。编排过程组织 CO 表示包含所有业务过程参与者的组织结构，即 $\text{CO}=\{U,R,\text{UR}\}$，其中，$U$ 表示编排过程组织的参与者，R 表示编排过程组织中包含的角色，UR 表示该编排过程组织中参与者与角色的映射关系。

定义 4　访问控制合约。访问控制合约 ACC 是一个用于提供授权机制与访问控制机制的智能合约，ACC 中存储了编排合约与编排过程组织的映射关系 CC_{map}，它将编排合约地址 CC_{addr} 映射到 CO 来唯一确定编排合约对应的编排过程组织。

过程管理者在完成编排合约 CC 的部署后，使用 CC_{addr} 在 ACC 中创建 CO，在 CO 中导入相关的 U 和 R，并通过 UR 给参与者分配角色。相关操作都以区块链交易的形式存储于以太坊区块中，任何参与者都能够对过程管理者的所有操作进行追溯。

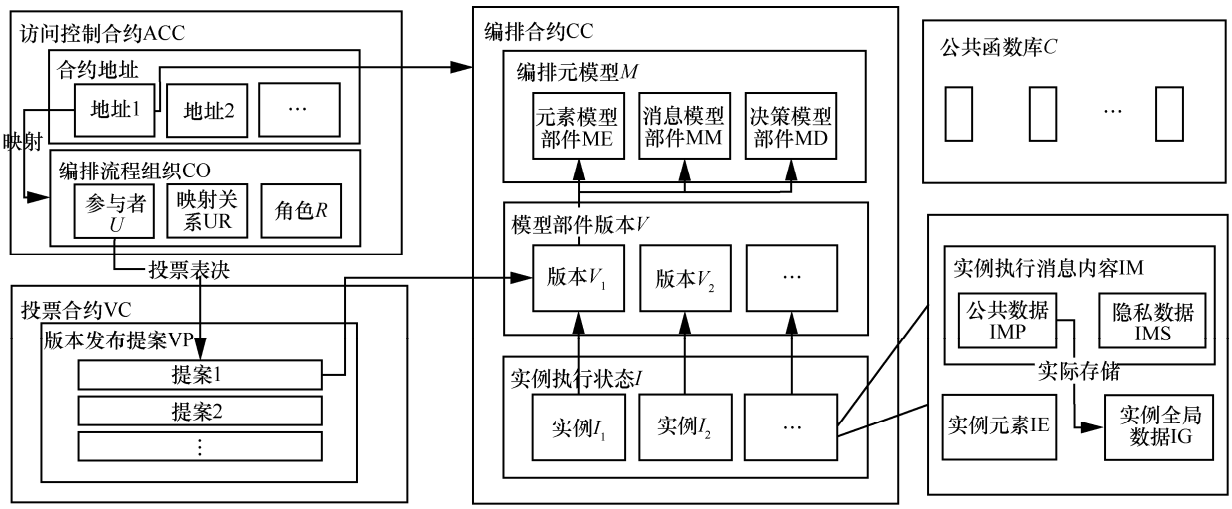


图 2　链上智能合约结构

3) 模型版本控制

业务过程模型往往会随着时间发生变化，而现有的实现原型中没有为基于区块链的业务过程模型版本控制提供支持。本文所提出的区块链业务过程管理框架通过复用模型部件版本 V 提供版本控制功能。同时，为了增强框架的去中心化程度，本文在模型部件版本 V 的发布过程中引入投票机制来限制过程管理者的权力。为了实现所述功能，本文在框架内增加了投票合约。

定义 5 投票合约。投票合约 VC 是一个用于生成版本发布提案 VP 和提供投票决策功能的智能合约。其中，$VP=\{CC_{addr},V_{id}\}$，表示通过 CC_{addr} 和模型部件版本标识 V_{id} 能够唯一确定一个编排模型。

CC 的模型部件版本 V 创建后，会自动使用 CC_{addr} 和 V_{id} 在投票合约 VC 中发布一个版本发布提案 VP，只有当对应的 CO 中所有的参与者 U 都投票赞成该版本发布 VP 后，该提案中指向的 V 才能投入使用。

4) 隐私控制策略

区块链上存储的数据能够被公共网络的任意节点访问，这不利于组织间的隐私数据的交互，为此本文引入非对称加密算法用于加密实例执行过程中的消息交互内容。

定义 6 实例执行消息内容。实例执行消息内容 IM 是存储于编排合约 CC 的过程实例所包含的消息交互数据，由公共数据 IMP 和隐私数据 IMS 组成，即 IM = {IMP,IMS}。

实例执行消息内容中的 IMP 存储于实例全局数据 IG 中，包括能够向所有参与者公开的消息内容和参与决策网关决策条件的数据内容；而 IMS 是指包含敏感数据的消息内容，只能被消息的发送者和接收者获取。本文框架采用非对称加密算法，其中每个参与者都在 CO 中存有各自的公钥，当消息发送者向消息接收者发送消息时，能够在 CO 获取消息接收者的公钥，并且使用公钥加密消息数据并将消息发布到 IM 中，消息接收者读取相应消息并使用私钥对加密消息数据解密，根据解密后消息内容选择接收或者拒绝该消息，被拒绝的消息能够由发送者重新发送。

4.2 框架架构

本文提出框架的架构如图 3 所示，主要分为链下组件和链上智能合约两部分，其中链下组件由建模器、翻译器、部署器和事件监听器等提供功能支持。

建模器。本文使用的建模器[24]基于 BPMN-JS 开发，能够涵盖本文支持的所有编排图元素，并且具有可扩展、直观性较强和易于集成等优点，使用该建模器能够生成 xml 格式的编排图文件 F。

翻译器。翻译器是用于解析建模器输出的编排图文件 F 的链下组件，使用该组件能够从 F 中解析出元素 ME、消息 MM 和决策 MD 等模型部件，并且以 JS 对象简谱（JSON, JavaScript object notation）文件格式输出。

部署器。部署器用于部署编排合约 CC、投票合约 VC、访问控制合约 ACC、公共函数库 C 以及业务过程编排元模型 M 中包含的元素 ME、消息

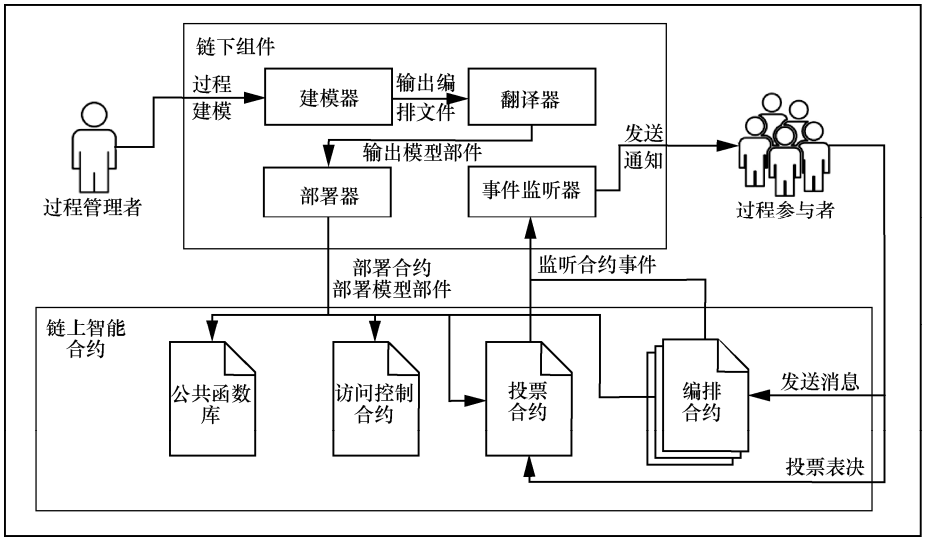


图 3 编排图驱动的区块链业务过程管理框架架构

MM 和决策 MD 等模型部件。以太坊智能合约可直接部署；业务过程编排元模型 M 中包含的模型部件则由部署器通过获取翻译器输出的 JSON 文件内容，同时对比编排合约 CC 内已存在的模型部件，进行选择性部署。

事件监听器。事件监听器用于监听编排合约 CC 内实例执行状态 I 的状态变化事件以及投票合约 VC 中版本发布提案 VP 的生成事件，并以此跟踪过程实例的执行，将相关信息通知过程参与者。

链上智能合约包括公共函数库 C、访问控制合约 ACC、投票合约 VC 和编排合约 CC。其中 ACC、VC 和 C 只需部署一次，访问控制合约 ACC 与投票合约 VC 用于控制不同编排合约 CC 中的访问控制权限和模型版本迭代，公共函数库 C 为其余合约提供了公共计算函数。

4.3　执行步骤

本文提供的框架执行过程主要包括 3 个阶段，具体步骤如图 4 所示。

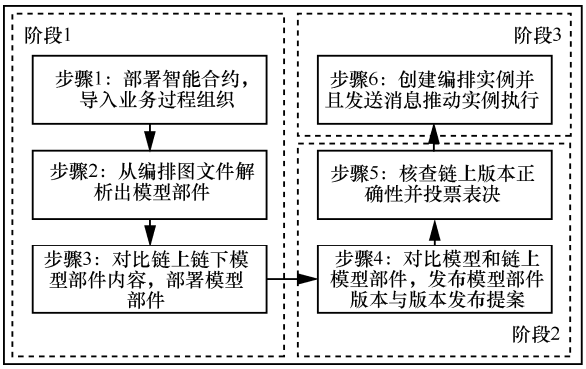


图 4　框架执行过程

阶段 1 为智能合约部署与数据上传阶段，主要包含以太坊智能合约部署、编排图文件解析以及模型部件部署 3 个步骤。

步骤 1　过程管理者部署访问控制合约 ACC、投票合约 VC、编排合约 CC 与公共函数库 C。在部署投票合约时需要传入访问控制合约地址 ACC_{addr}；在部署编排合约时需要传入访问控制合约地址 ACC_{addr} 以及投票合约地址 VC_{addr}，以便在后续的合约访问控制和投票决策过程中进行合约调用，相关智能合约的部署成本由过程管理者负担或者由过程参与者进行协商负担。完成 ACC 和 CC 部署后，过程管理者导入对应业务过程组织 CO 中包含的 U 和 R，并通过 UR 来完成参与者角色的静态分配。

步骤 2　链下翻译器输入一个描述跨组织业务过程的信息交互和过程执行过程的 BPMN 文件（该文件由过程管理者通过建模器建模获得），解析该文件中包含的元素、消息和决策等模型部件并以 JSON 格式输出。该步骤涉及的解析过程如算法 1 所示。

算法 1　parsingChor (BPMN-File F)

输入　建模器建模输出的 BPMN 编排图文件 F

输出　编排图文件 F 中所包含的元素 ME、消息 MM 和决策 MD 等模型部件

```
1) function   parsingChor (BPMN-File F)
2)      visited ←new Set<Element> //解析元素
3)      Model←readModelFromFile(F)//获取模型
4)      for each Edge in Model do //遍历序列流
5)       if containDecision(Edge) is true
6)       Edge.targetElement←Edge.Decision
7)       MD← MD⋃Edge.Decision
8)       end if
9)       ElementPre←Edge.sourceElement//前驱
10)       if ElementPre not in visited
11)       visited←visited⋃ElementPre
12)       ME←ME⋃ElementPre
13)         for each Message in ElementPre do
14)         MM←MM⋃Message
15)         end for
16)       end if
17)       ElementNext←Edge.targetElement//后继
18)            //序列流目标元素解析同第 10)～第 16)行
19)      end for
20)   return   (ME,MM,MD)
21) end function
```

在算法 1 中，第 3)行将 BPMN 文件 F 解析为包含模型部件 ME、MM 和 MD 的编排图模型 Model。第 4)～19)行遍历编排图模型中的每一个序列流，其中第 5)～8)行解析包含决策条件的序列流，获得决策模型部件 MD；第 9)～16)行解析该序列流的前驱元素为元素模型部件 ME，并将其内部包含的消息解析为消息模型部件 MM；第 17)～18)行以相同方式解析序列流目标元素。第 20)行返回该编排图文件中包含的模型部件 (ME,MM,MD)。

步骤 3　部署器将算法 1 解析所得的链下业务过程模型中模型部件 (ME,MM,MD) 与从编排合约

CC 的业务过程编排元模型 M 中读取的链上模型部件 (ME_{online},MM_{online},MD_{online}) 进行对比，并将其导入编排合约 CC 的业务过程编排元模型 M 中，如算法 2 所示。

算法 2 deployM(ME,MM,MD)

输入 算法 1 解析模型所得元素 ME、消息 MM、决策 MD 等模型部件

输出 导入编排元模型 M 后的编排合约 CC

```
1) function deployM(ME,MM,MD)
2)     ExistElement← getElementFromEth(CC)
3)     ExistMessage← getMessageFromEth(CC)
4)     ExistDecision← getDecisionFromEth(CC)
5)     for each Element in ME do
6)       if Element not in ExistElement
7)       CC.M←<Element,1>//版本号为 1
8)       else if Element in ExistElement
9)         if isDifferent(Element,ExistElement)
10)          VID←getMEMaxVID(Exist Element)+1
11)          CC.M←<Element, VID>
12)        else continue
13)        end if
14)      end if
15)    end for
16)    //消息和决策部署同第 5)～第 16)行
17)    return CC
18) end function
```

在算法 2 中，第 2)～第 4)行表示从当前 CC 中读取链上已存在的元素、消息和决策等模型部件。第 6)行比较链下元素模型部件是否已经存在于链上。如不存在，则创建版本号为 1 的元素模型部件并将其部署于 CC 的 M 中。第 9)行比较当前链下元素模型部件内容是否与链上元素模型部件相同，如果相同，则跳过当前元素，否则更新该元素模型部件的版本号为最新的版本号（链上对应最大版本号加 1），然后同样部署于 CC 的 M 中。消息和决策模型部件的部署方式与元素模型部件相同。

阶段 2 为模型的版本控制阶段，主要包含在 CC 中创建 V 、在 VC 中创建 VP 以及由过程参与者 U 对 VP 进行投票表决的过程，具体步骤如下。

步骤 4 在完成步骤 3 之后，CC 已存储 M ，此时需要对比链下模型部件和链上模型部件并得到 V ，对比方式与算法 2 相似。过程管理者调用编排合约的发布版本函数 addVersion，将 V 发布到 CC 中，随后在该函数内部发起对 VC 的合约函数调用，在投票合约中新增 VP 。然后，链下事件监听器组件监听投票合约内部事件，并通过 VP 内 CC_{addr} 和 V_{id} 检索到 V ，读取 V 中引用对应版本号的模型部件，并通知相应 CO 的 U 。

步骤 5 过程参与者 U 收到步骤 4 中的通知后查看 V 中引用的模型部件，通过对比链下过程模型与链上模型部件版本来核查过程管理者发布模型版本的正确性，此处也可使用现有的模型校验工具[25-26]校验链上模型部件版本的可执行性。参与者在确认 V 的正确性后，在 VC 中对该 VP 进行表决。只有当所有的过程参与者都支持 VP 时，对应的 V 才能投入使用。同时，编排合约中现有过程实例遵循原有模型部件版本继续执行。

阶段 3 为模型的实例化以及链上的实例执行控制阶段，主要包括版本模型实例化、过程参与者间消息交互以及过程实例执行，具体步骤如下。

步骤 6 当模型部件版本 V 得到所有过程参与者 U 的认可后，过程管理者就能够使用该模型部件版本 V 创建过程实例。对应过程实例存储在实例执行状态中，每一个过程实例包含实例所属版本、实例状态、实例元素数据信息、实例消息数据信息以及实例全局数据变量等相关数据。其中，实例元素存在等待、可执行和已完成 3 种状态，实例消息存在发送、接收和拒绝 3 种状态。当实例元素处于可执行状态时，流程参与者即可发送该实例元素包含的消息，此时对应消息被创建并被置为发送状态，消息中用于决策网关路由的变量被存储到实例全局数据中，消息中包含的隐私数据能够使用消息接收者的以太坊公钥进行加密。随后，消息接收者选择接收或者拒绝该消息，并触发对应的消息状态转换函数。当该实例元素中所有消息都处于已接收状态时将会触发实例元素的状态转换函数，其将该实例元素转换为已完成状态，并根据该过程实例绑定的模型部件版本和实例全局数据启用后继元素，将相应元素转换为可执行状态。此外，本文框架采用延迟元素实例化时机的设计方法，过程实例包含的元素不会在创建实例时生成，而是根据过程实例的执行路由动态实例化对应的实例元素。值得注意的是，上述操作都会在 ACC 中进行权限验证，没有对应角色的参与者相关操作会被 CC 拒绝。

5　实验评估

5.1　案例与实验环境说明

本文建模、执行和分析了以 HE 助听器公司为主导的一个真实商品订购业务过程，如图 5 所示。

图 5 所示场景涉及了 4 个参与者（加盟门店、生产商、物流商、发货承运商）、14 个编排图元素、11 条消息。如果在消息交互过程出现延误或差错，参与者之间可能会相互指责，而且加盟门店没有自己的数据存储，当其与生产商产生冲突时，生产商能够通过篡改数据来使自己获利。引入区块链业务过程管理系统后可解决上述问题。例如，当门店与生产商产生冲突时，门店只需要追溯过程实例的消息内容与对应以太坊交易的时间戳即可维护利益，不再需要受信任第三方。此外，本文框架还支持商品订购编排过程的实时更新，对于模型的更新需要得到所有参与者认可后才会生效。

本文基于所提区块链业务过程管理系统框架重现了 10 个该案例的过程实例执行过程，相关过程实例覆盖了案例模型中包含的所有元素、消息和决策，链下组件基于 node.js 和 Java 实现，链上智能合约使用 Solidity 编辑器 solc-js 编译，通过 remix-ide 部署于本地以太坊私链之上，相应智能合约代码已在 GitHub 上公开。以太坊私链则由以太坊节点仿真器 Ganache 搭建，并选取其中的 10 个以太坊账户作为过程参与者。

5.2　成本分析

在以太坊区块链上计算、存储数据和部署智能合约需要向以太坊矿工支付加密货币作为手续费，而需要支付的手续费为衡量相关操作复杂性的 gas 数量和由调用者自行设置的 gas 单价的乘积，手续费较高的交易会优先被以太坊矿工打包到新的区块中。为了公平起见，本文将使用 gas 数量作为区块链业务过程管理框架的成本衡量指标。

本文所提框架的执行总成本主要分为合约部署成本、合约数据导入与版本投票决策成本、过程实例创建与执行成本三部分。

表 1 给出了部署框架内的访问控制合约 ACC 、投票合约 VC 、编排合约 CC 以及公共函数库 C 的合约部署成本。

值得注意的是，不同组织业务过程模型需要部署不同的编排合约，故合约的部署成本 $\mathrm{SC}_{\mathrm{deploy}}$ 可由式(1)得到。

$$\mathrm{SC}_{\mathrm{deploy}} = \mathrm{ACC}_{\mathrm{cost}} + \mathrm{VC}_{\mathrm{cost}} + C_{\mathrm{cost}} + i\mathrm{CC}_{\mathrm{cost}} \quad (1)$$

其中，$\mathrm{ACC}_{\mathrm{cost}}$ 为访问控制合约部署成本，$\mathrm{VC}_{\mathrm{cost}}$ 为投票合约部署成本，C_{cost} 为公共函数库部署成本，$\mathrm{CC}_{\mathrm{cost}}$ 为编排合约部署成本，i 为业务过程模型个数。

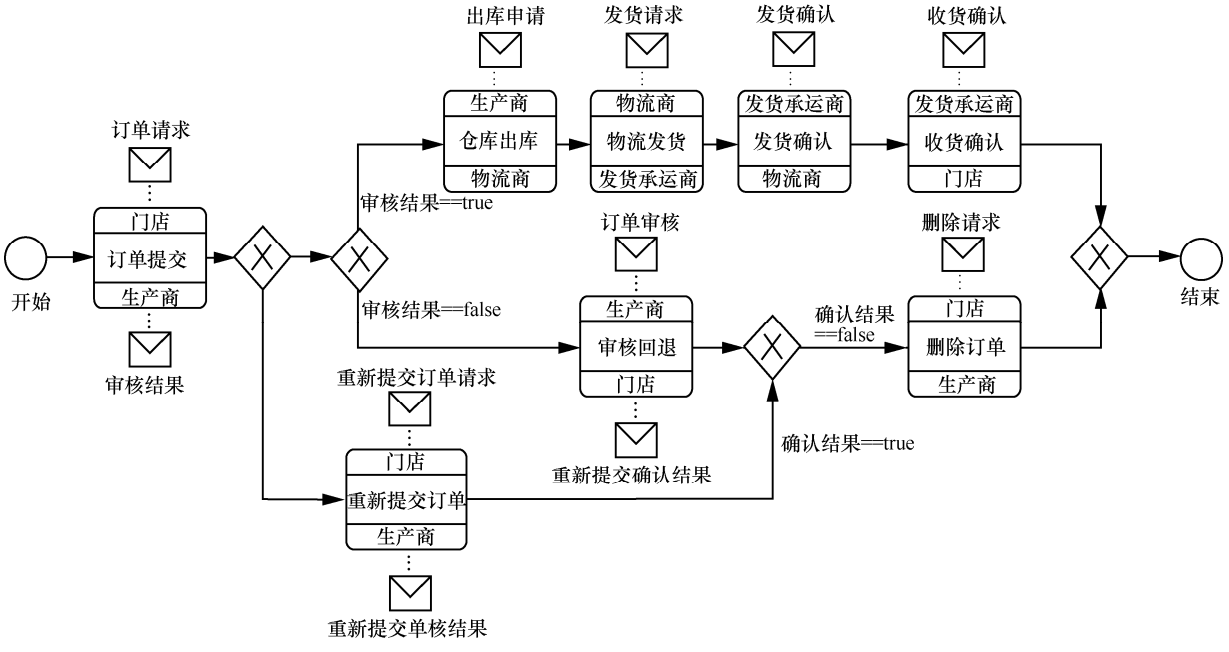


图 5　商品订购编排图

表 1 合约部署成本

合约名称	部署 gas 消耗
访问控制合约 ACC	1 644 441
投票合约 VC	623 563
公共函数库 C	756 201
编排合约 CC	5 046 186
合计	8 070 391

以太坊智能合约部署完成后，无法直接创建业务过程实例，需要在创建实例前导入 CO、M 和 V，并且由 U 对 VC 中的 VP 进行投票表决。将本地测试环境 Ganache（以太坊节点仿真器）中包含的 10 个以太坊账户作为过程参与者导入访问控制合约，并为其随机分配过程角色 R。执行上述操作需要调用的智能合约函数和相应的以太坊 gas 消耗如表 2 所示。

表 2 合约数据导入与版本投票决策成本

函数名称	gas 总消耗	调用次数/次	平均 gas 消耗
addParticipant	1 178 650	10	117 865
addRole	331 076	4	82 769
allocationRole	524 770	10	524 177
addElement	4 230 463	14	302 175
addMessage	2 058 554	11	187 141
addDecision	566 732	4	141 683
addVersion	755 741	1	755 741
vote	639 248	10	63 924
合计	10 285 234	64	160 706

合约数据导入与版本投票决策成本 $\mathrm{CI}_{\mathrm{pre}}$ 可由式(2)计算得到。

$$\mathrm{CI}_{\mathrm{pre}} = M_{\mathrm{import}} + j(U_{\mathrm{cost}} + \mathrm{UR}_{\mathrm{cost}}) + iR_{\mathrm{cost}} + k\mathrm{V}_{\mathrm{cost}} + kj\mathrm{Vote}_{\mathrm{cost}} \tag{2}$$

其中，R_{cost} 为添加单个角色成本，U_{cost} 为添加单个过程参与者成本，UR_{cost} 为参与者分配角色成本，V_{cost} 为发布单个版本成本，$\mathrm{Vote}_{\mathrm{cost}}$ 为单个参与者投票决策成本，i 为角色个数，j 为参与者个数，k 为版本个数，M_{import} 为编排元模型导入成本。M_{import} 由元素模型部件导入总成本 $\mathrm{ME}_{\mathrm{cost}}$、消息模型部件导入总成本 $\mathrm{MM}_{\mathrm{cost}}$ 和决策模型部件导入总成本 $\mathrm{MD}_{\mathrm{cost}}$ 合计而成，如式(3)所示。

$$M_{\mathrm{import}} = \mathrm{ME}_{\mathrm{cost}} + \mathrm{MM}_{\mathrm{cost}} + \mathrm{MD}_{\mathrm{cost}} \tag{3}$$

在完成合约数据导入和版本投票表决后，CC 就能够创建基于 V 的过程实例，并且按照 V 中引用的模型部件实现过程实例消息的交互，完成过程实例的执行。表 3 给出了商品订购场景下执行 10 个过程实例的 gas 消耗。

表 3 过程实例创建与执行成本（10 个实例）

函数名称	总 gas 消耗	调用次数/次	平均 gas 消耗
newInstance	2 061 520	10	206 152
sendMessage	15 109 226	71	212 806
ackMessage	11 121 795	71	156 645
completeTask	6 228 064	47	132 512
合计	34 520 605	10	3 452 060

本文框架中的过程实例创建与执行成本 I_{cexec} 为

$$I_{\mathrm{cexec}} = i(I_{\mathrm{create}} + I_{\mathrm{exec}}) \tag{4}$$

其中，I_{create} 为实例创建成本，I_{exec} 为实例执行成本，i 为过程实例个数。

合并式(1)、式(2)和式(4)可得本文框架执行总成本 C_{sum} 为

$$C_{\mathrm{sum}} = \mathrm{SC}_{\mathrm{deploy}} + \mathrm{CI}_{\mathrm{pre}} + I_{\mathrm{cexec}} \tag{5}$$

由表 3 可知，对于本文案例的编排模型而言，单个过程实例创建和执行成本的平均 gas 消耗约为 345×10^4；由表 1 和表 2 可知，合约部署成本、合约数据导入与版本投票决策成本的 gas 数量消耗合计约为 $1\,800\times10^4$，而随着过程实例数目的增加，该成本能够被均摊到每一个过程实例中，故每一个过程实例的平均成本为

$$I_{\mathrm{avg}} = I_{\mathrm{create}} + I_{\mathrm{exec}} + \frac{\mathrm{SC}_{\mathrm{deploy}} + \mathrm{CI}_{\mathrm{pre}}}{i} \tag{6}$$

其中，i 表示过程实例的个数。

5.3 与现有工作的比较

本节将本文工作与当前区块链上针对 BPMN 编排图实现业务过程管理的最新研究进展的代表工作文献[2,4-6]进行定性比较，结果如表 4 所示，其中，★越多代表在相关方面越具有优势。

控制支持。大多数实现原型能够在区块链上控制业务过程的走向，同时提供了基于网关数据决策的支持。文献[4]提出的原型扩展了编排图在区块链上的定义，相比其他的实现原型支持更多的编排图元素。

访问控制。文献[2]最早提出在区块链上执行编排业务过程的框架，但其未对此添加访问控制支持，文献[6]的实现与其类似，而文献[4]提出的框架

表 4　　相关工作的定性比较

工作	控制支持	访问控制	版本控制	隐私控制	实例化成本	执行成本
文献[2]	★★☆	☆☆☆	☆☆☆	★★☆	★☆☆	★★★
文献[4]	★★★	★★☆	☆☆☆	★★☆	★☆☆	★★★
文献[5]	★★☆	★★☆	☆☆☆	☆☆☆	★☆☆	★★★
文献[6]	★★☆	☆☆☆	☆☆☆	☆☆☆	★★☆	★★☆
本文工作	★★☆	★★☆	★★★	★★☆	★★★	★☆☆

和本文框架都提供额外用于访问控制的智能合约。文献[5]提出的框架将参与者角色分为必须角色和可选角色，其中角色必须通过预先分配方式在合约部署时导入。

版本控制。本文框架是目前该领域少数提供区块链上过程模型的版本控制支持的实现原型。通过复用编排元模型和采用模型部件版本集合的设计方法来实现版本控制功能，以降低以太坊存储开销。

隐私控制。文献[5-6]提出的原型没有为业务过程中消息数据提供隐私控制支持，相应的消息内容被存储在智能合约中，任何参与者都能够访问消息内容。而文献[2, 4]和本文框架将编排消息作为通用数据结构存储，便于使用加密算法对链上消息数据进行加密。

实例化成本与执行成本。从表 4 可以看出，除本文框架外的其他所有实现原型都需要为单个过程实例创建智能合约，这种实现方式使过程实例化成本大幅度上升。而本文采用复用编排元模型和延迟实例元素创建时机的设计方式，过程实例化的成本大幅降低，而代价是过程实例的执行成本上升。

本文框架与 chorChain[5]的成本对比如图 6 所示。选择 chorChain 进行成本定量比较的原因是该框架支持通用编排过程的链上执行，并已公开其实现代码；而文献[2]、文献[4]和文献[6]各自的工作侧重点不同，导致相应的智能合约设计与实现方式截然不同，只针对特定编排过程进行验证或者尚未公开具体的实现代码，从而难以进行合理的成本定量对比。

根据前文所述，本文所提框架的执行总成本主要分为合约部署成本、合约数据导入与版本投票决策成本、过程实例创建与执行成本三部分。其中，合约部署成本、合约数据导入与投票决策成本对于单个过程模型只需消耗一次，所以在图 6 中统一表示为框架部署成本；而实例创建成本和实例执行成本随着实例数目的增加而增长。另一方面，chorChain 的执行总成本主要分为实例创建成本和实例执行成本，其中，实例创建成本反映了为每一个过程实例部署一个实例智能合约需要付出的 gas 消耗，实例执行成本反映了在每一个实例智能合约中完成对应实例的执行过程需要付出的 gas 消耗。显然，这两者都会随着实例数目的增加而增长。

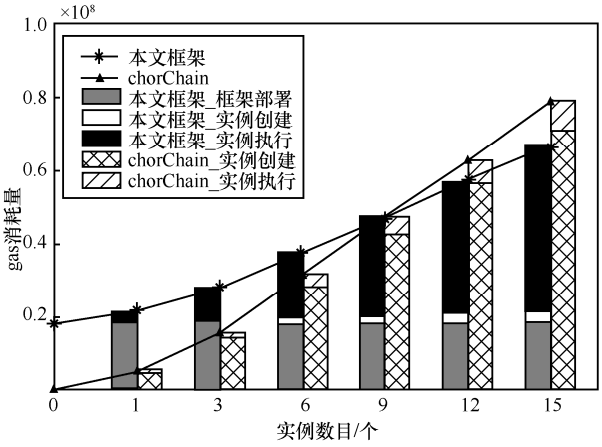


图 6　成本对比

由表 1、表 2 和图 6 可知，本文框架需要付出合约部署成本 gas 消耗为 8 070 391，合约数据导入与版本投票决策成本 gas 消耗为 10 285 234，即在创建过程实例前需要付出框架部署成本 gas 消耗 18 355 625。而在此之后，本文框架每创建一个过程实例仅需 gas 消耗 206 152，而完成实例执行的平均 gas 消耗为 3 245 908。另一方面，虽然 chorChain 不需要额外负担框架部署成本开销，但在执行过程中需要为每个实例部署一个实例智能合约，从而付出实例创建成本 gas 消耗为 4 730 220，这远大于本文框架的实例创建成本。对于已部署的实例智能合约，chorChain 完成该实例执行的平均成本 gas 消耗为 665 238。

综上所述，尽管本文框架需要在实例创建前付出额外的框架部署成本且实例执行平均成本大于

chorChain 框架，但由于区块链智能合约部署成本远大于调用合约函数成本，因此本文框架所需执行总成本在执行较多的实例数时整体上优于 chorChain。由图 6 可知，当实例数目大于 9 时，本文框架的 gas 消耗就低于 chorChain，并且随着实例数的增加，gas 消耗差距将被进一步拉大。因此本文框架在实例执行上付出的额外开销是能够被接受的。值得注意的是，本文框架的设计原理决定了其在其他场景下仍可表现出更低的成本开销。当然，如果编排图模型过于复杂，将可能导致本文框架中实例执行成本超过 chorChain 的实例执行成本。但事实上，单个以太坊区块存在 gas 消耗上限，难以支持过于复杂的编排图模型过程实例合约，故这种情况很难发生。

5.4 商品订购实例说明

本节给出某公司的一个商品订购实例，说明基于本文框架的执行过程，其消息发送时序的简化示意如图 7 所示。

该商品订购实例包含 4 个参与者，分别为某公司门店、生产商、物流商和发货承运商。其执行过程涉及 6 次发送消息函数调用和 6 次确认消息函数调用。该实例的执行总 gas 消耗为 2 934 666。

5.5 局限性

尽管本文提出的框架具有灵活、实例化成本低和支持版本控制等优点，但该框架并不适用于所有的编排过程。因此本节将介绍该框架的局限性。

过程模型。本文提出的框架只支持编排图中部分元素，缺乏对于编排子过程、标记等相关内容的支持，而区块链与外部数据交互的复杂性与智能合约基础技术的限制也导致智能合约难以支持过程模型中的计时器。因此，目前本文框架只支持基础编排图元素，该限制可以通过扩展智能合约结构、添加链下中介器等方式来解决，这超出了本文的工作范围，留待后续工作进行补充。

隐私性。尽管本文在编排过程的消息交互过程中提供了非对称加密算法来加密消息的内容，但涉及决策网关的消息数据必须显式地存储在实例的全局变量中，整个编排过程的过程模型也以编排元

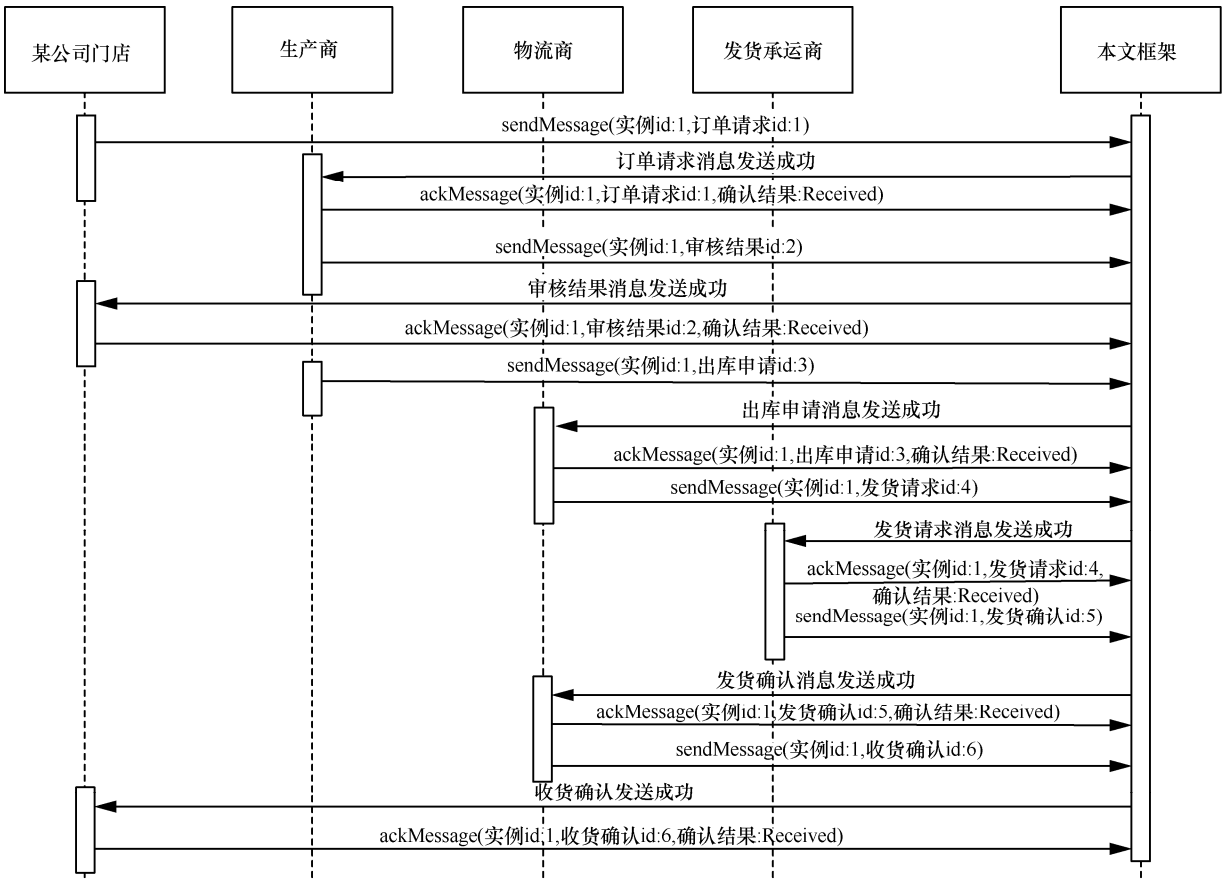


图 7 商品订购实例的消息时序示意

模型中模型部件的形式显式存储在智能合约中。当编排过程模型和决策条件中包含敏感数据时不建议使用本文提出的框架，该限制可以考虑使用许可的区块链或者基于现有的相关工具[27]来解决。

访问控制方法。本文框架支持传统的基于角色的访问控制方法，但参与者和角色需要管理者提前进行导入和分配，过程管理者的引入降低了在访问控制方面的去中心化程度。现有解决方案包括在访问控制方法中引入投票机制或者使用文献[10]提出的动态绑定模型和绑定策略规范语言来支持去中心的访问控制方法。

除上述局限性以外，基于区块链的业务流程管理框架普遍存在吞吐量低、交易验证时延等区块链本身所带来的局限性，这些限制需要依赖现有区块链技术的进一步发展来解决。

6　结束语

本文提出了一种编排图驱动的区块链业务过程管理框架，本文框架与传统业务过程管理技术相比，具有去中心化、过程透明和防止篡改等优点。与当前领域相关工作相比，本文框架通过复用模型数据和引入投票机制来支持区块链上业务过程的版本控制；能够在单个智能合约中支持多个版本的编排业务过程模型，且多过程实例集成和延迟元素实例化时机的设计方法也大幅降低了创建过程实例的以太坊成本开销。

下一步工作将优化本文提出框架内的以太坊智能合约、完善框架功能、开发可视化的框架执行界面，并且在添加了版本控制的框架中提供实例迁移支持。

参考文献：

[1] MENDLING J, WEBER I, AALST W V D, et al. Blockchains for business process management - challenges and opportunities[J]. ACM Transactions on Management Information Systems, 2018, 9(1): 1-16.

[2] WEBER I, XU X, RIVERET R, et al. Untrusted business process monitoring and execution using blockchain[C]//International Conference on Business Process Management. Berlin: Springer, 2016: 329-347.

[3] GARCÍA-BAÑUELOS L, PONOMAREV A, DUMAS M, et al. Optimized execution of business processes on blockchain[C]// International Conference on Business Process Management. Berlin: Springer, 2017: 130-146.

[4] LADLEIF J, WESKE M, WEBER I. Modeling and enforcing blockchain-based choreographies[C]//International Conference on Business Process Management. Berlin: Springer, 2019: 69-85.

[5] CORRADINI F, MARCELLETTI A, MORICHETTA A, et al. Engineering trustable choreography-based systems using blockchain[C]//Proceedings of the 35th Annual ACM Symposium on Applied Computing. New York: ACM Press, 2020: 1470-1479.

[6] LICHTENSTEIN T, SIEGERT S, NIKAJ A, et al. Data-driven process choreography execution on the blockchain: a focus on blockchain data reusability[C]//International Conference on Business Information Systems. Berlin: Springer, 2020: 224-235.

[7] LÓPEZ-PINTADO O, DUMAS M, GARCÍA-BAÑUELOS L, et al. Interpreted execution of business process models on blockchain[C]//2019 IEEE 23rd International Enterprise Distributed Object Computing Conference. Piscataway: IEEE Press, 2019: 206-215.

[8] DIJKMAN R, HOFSTETTER J, KOEHLER J. Business process model and notation[M]. Berlin: Springer, 2011.

[9] 曾诗钦, 霍如, 黄韬, 等. 区块链技术研究综述:原理、进展与应用[J]. 通信学报, 2020, 41(1):134-151.
ZENG S Q, HUO R, HUANG T, et al. Survey of blockchain: principle, progress and application[J]. Journal on Communications, 2020, 41(1):134-151.

[10] SZABO N. Formalizing and securing relationships on public networks[J]. First Monday, 1997, 2(9): 1-21.

[11] 贺海武, 延安, 陈泽华. 基于区块链的智能合约技术与应用综述[J]. 计算机研究与发展, 2018, 55(11): 2452-2466.
HE H W, YAN A, CHEN Z H. Survey of smart contract technology and application based on blockchain[J]. Journal of Computer Research and Development, 2018, 55(11): 2452-2466.

[12] VIRIYASITAVAT W, HOONSOPON D. Blockchain characteristics and consensus in modern business processes[J]. Journal of Industrial Information Integration, 2019, 13: 32-39.

[13] LADLEIF J, WEBER I, WESKE M. External data monitoring using oracles in blockchain-based process execution[C]//International Conference on Business Process Management. Berlin: Springer, 2020: 67-81.

[14] KÖPKE J, FRANCESCHETTI M, EDER J. Balancing privity and enforceability of BPM-based smart contracts on blockchains[C]//International Conference on Business Process Management. Berlin: Springer, 2019: 87-102.

[15] LÓPEZ-PINTADO O, GARCÍA-BAÑUELOS L, DUMAS M, et al. Caterpillar: a business process execution engine on the Ethereum blockchain[J]. Software: Practice and Experience, 2019, 49(7): 1162-1193.

[16] LÓPEZ-PINTADO O, DUMAS M, GARCÍA-BAÑUELOS L, et al. Dynamic role binding in blockchain-based collaborative business processes[C]//International Conference on Advanced Information Systems Engineering. Berlin: Springer, 2019: 399-414.

[17] STURM C, SCALANCZI J, SCHÖNIG S, et al. A blockchain-based and resource-aware process execution engine[J]. Future Generation

Computer Systems, 2019, 100: 19-34.

[18] STURM C, SZALANCZI J, JABLONSKI S, et al. Decentralized control: a novel form of interorganizational workflow interoperability[C]//IFIP Working Conference on The Practice of Enterprise Modeling. Berlin: Springer, 2020: 261-276.

[19] KLINGER P, BODENDORF F. Blockchain-based cross-organizational execution framework for dynamic integration of process collaborations[C]//15th International Business Informatics Congress. Berlin: Springer, 2020: 893-908.

[20] MADSEN M F, GAUB M, HØGNASON T, et al. Collaboration among adversaries: distributed workflow execution on a blockchain[C]//Symposium on Foundations and Applications of Blockchain. Wadern: Open Access Series in Informatics, 2018: 8.

[21] HAARMANN S, BATOULIS K, NIKAJ A, et al. DMN decision execution on the Ethereum blockchain[C]//International Conference on Advanced Information Systems Engineering. Berlin: Springer, 2018: 327-341.

[22] HAARMANN S, BATOULIS K, NIKAJ A, et al. Executing collaborative decisions confidentially on blockchains[C]//International Conference on Business Process Management. Berlin: Springer, 2019: 119-135.

[23] LADLEIF J, FRIEDOW C, WESKE M. An architecture for multi-chain business process choreographies[C]//International Conference on Business Information Systems. Berlin: Springer, 2020: 184-196.

[24] LADLEIF J, VON WELTZIEN A, WESKE M. Chor-js: a modeling framework for BPMN 2.0 choreography diagrams[C]//38th International Conference on Conceptual Modeling. Salvador: CEUR Workshop Proceedings, 2019: 113-117.

[25] 代飞, 赵文卓, 杨云,等. BPMN2.0 编排的形式语义和分析[J]. 软件学报, 2018, 29(4): 1094-1114.
DAI F, ZHAO W Z, YANG Y, et al. Formal semantics and analysis of BPMN 2.0 choreographies[J]. Journal of Software, 2018, 29(4): 1094-1114.

[26] POIZAT P, SALAÜN G. Checking the realizability of BPMN 2.0 choreographies[C]//Proceedings of the 27th Annual ACM Symposium on Applied Computing. New York: ACM Press, 2012: 1927-1934.

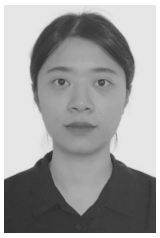
[27] KOSBA A, MILLER A, SHI E, et al. Hawk: the blockchain model of cryptography and privacy-preserving smart contracts[C]//2016 IEEE Symposium on Security and Privacy. Piscataway: IEEE Press, 2016: 839-858.

[作者简介]

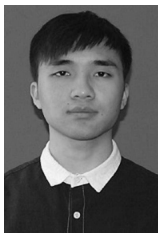
俞东进（1969− ），男，浙江平湖人，博士，杭州电子科技大学教授、博士生导师，主要研究方向为服务计算、大数据、智能软件工程等。

韦懿杰（1996− ），男，浙江温州人，杭州电子科技大学硕士生，主要研究方向为业务过程管理、区块链等。

孙笑笑（1991− ），女，浙江浦江人，博士，杭州电子科技大学讲师、工程师，主要研究方向为时空数据挖掘、大数据分析、业务过程管理等。



倪可（1997− ），女，浙江杭州人，杭州电子科技大学硕士生，主要研究方向为业务过程管理。



沈沪军（1997− ），男，浙江绍兴人，杭州电子科技大学硕士生，主要研究方向为业务过程管理、区块链等。