(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 111813836 A
(43)申请公布日 2020.10.23

(21)申请号 202010814174.7

(22)申请日 2020.08.13

(71)申请人 广州东港安全印刷有限公司
地址 510700 广东省广州市高新技术产业开发区科学城南云五路5号
申请人 广州科创空间信息科技有限公司

(72)发明人 郭俊 姜忠伟 刘金庆

(74)专利代理机构 广州鼎贤知识产权代理有限公司 44502
代理人 刘莉梅

(51)Int.Cl.
G06F 16/2458(2019.01)
G06F 16/23(2019.01)
G06F 16/27(2019.01)

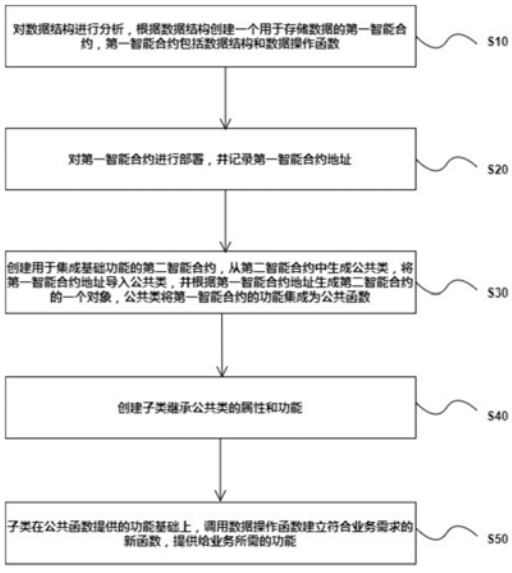
权利要求书1页 说明书4页 附图2页

(54)发明名称

一种提高Ethereum区块链系统扩展性的方法

(57)摘要

本发明公开一种提高Ethereum区块链系统扩展性的方法，包括以下步骤：对数据结构进行分析，根据数据结构创建用于存储数据的第一智能合约；对第一智能合约进行部署，并记录第一智能合约地址；创建第二智能合约，从第二智能合约中生成公共类，将第一智能合约地址导入公共类，并根据第一智能合约地址生成第二智能合约的对象，公共类将第一智能合约的功能集成为公共函数；创建子类继承公共类的属性和功能；子类调用数据操作函数建立符合业务需求的新函数，提供给业务所需的功能。本发明通过对智能合约进行功能上的分割，保证存储数据层面与逻辑层面的分离，无需修改智能合约的数据层，只需通过智能合约的逻辑控制即可完成修改升级操作。



CN 111813836 A

1.一种提高Ethereum区块链系统扩展性的方法,其特征在于,包括以下步骤:

S10、对数据结构进行分析,根据所述数据结构创建一个用于存储数据的第一智能合约,所述第一智能合约包括数据结构和数据操作函数;

S20、对所述第一智能合约进行部署,并记录第一智能合约地址;

S30、创建用于集成基础功能的第二智能合约,从所述第二智能合约中生成公共类,将所述第一智能合约地址导入所述公共类,并根据所述第一智能合约地址生成所述第二智能合约的一个对象,所述公共类将所述第一智能合约的功能集成为公共函数;

S40、创建子类继承所述公共类的属性和功能;

S50、所述子类在所述公共函数提供的功能基础上,调用所述数据操作函数建立符合业务需求的新函数,提供给业务所需的功能。

2.根据权利要求1所述的提高Ethereum区块链系统扩展性的方法,其特征在于,所述步骤S20中,所述第一智能合约通过所述区块链中的虚拟机将其编译为计算机底层的编译文件,并将所述编译文件发布至所述区块链中的全部节点,全部所述节点达成共识并存储所述第一智能合约,当所述区块链返回所述第一智能合约地址后,则说明所述第一智能合约部署完成。

3.根据权利要求1所述的提高Ethereum区块链系统扩展性的方法,其特征在于,所述步骤S10中,所述数据操作函数包括插入函数、更新函数、删除函数和查询函数。

4.根据权利要求1所述的提高Ethereum区块链系统扩展性的方法,其特征在于,所述步骤S50中,当需要更新业务逻辑时,只需新建一个新智能合约继承所述第二智能合约的公共类,并根据新的业务需求对所述新智能合约进行修正后,将所述新智能合约部署到所述区块链中,前端服务器将原来的所述子类的智能合约地址修改为所述新智能合约的地址,即可完成升级更新合约的操作。

5.根据权利要求4所述的提高Ethereum区块链系统扩展性的方法,其特征在于,所述步骤S50中,所述区块链需要调用所述第二智能合约时,只需通过所述第二智能合约的地址去指向需要执行的所述第二智能合约。

一种提高Ethereum区块链系统扩展性的方法

技术领域

[0001] 本发明涉及区块链技术领域，具体涉及一种提高Ethereum区块链系统扩展性的方法。

背景技术

[0002] Ethereum区块链是基于比特币系统中点对点的设计理念上，开发出来的一个开源的有智能合约功能的公共区块链平台。而在其底层有自己一套以太坊虚拟机，并作为配套使用的开发了一套名为Solidity的编程语言，从而让用户，在其基础上建造自己的业务逻辑。而如今虽然功能上能完成大部分的业务逻辑，但在此系统中的业务逻辑中，还存在着部分功能的缺陷。比如，智能合约在部署之后，是无法对其进行更新和升级的，存储的数据也是与智能合约绑定，若是业务上需要进行更新或修改而重新构建新的智能合约时，旧的智能合约上面的数据也会丢失。

发明内容

[0003] 本发明的目的在于提供一种提高Ethereum区块链系统扩展性的方法，解决合约的更新修改会引起数据丢失的问题。

[0004] 为达此目的，本发明采用以下技术方案：

[0005] 提供一种提高Ethereum区块链系统扩展性的方法，包括以下步骤：

[0006] S10、对数据结构进行分析，根据所述数据结构创建一个用于存储数据的第一智能合约，所述第一智能合约包括数据结构和数据操作函数；

[0007] S20、对所述第一智能合约进行部署，并记录第一智能合约地址；

[0008] S30、创建用于集成基础功能的第二智能合约，从所述第二智能合约中生成公共类，将所述第一智能合约地址导入所述公共类，并根据所述第一智能合约地址生成所述第二智能合约的一个对象，所述公共类将所述第一智能合约的功能集成为公共函数；

[0009] S40、创建子类继承所述公共类的属性和功能；

[0010] S50、所述子类在所述公共函数提供的功能基础上，调用所述数据操作函数建立符合业务需求的新函数，提供给业务所需的功能。

[0011] 作为提高Ethereum区块链系统扩展性的方法的一种优选方案，所述步骤S20中，所述第一智能合约通过所述区块链中的虚拟机将其编译为计算机底层的编译文件，并将所述编译文件发布至所述区块链中的全部节点，全部所述节点达成共识并存储所述第一智能合约，当所述区块链返回所述第一智能合约地址后，则说明所述第一智能合约部署完成。

[0012] 作为提高Ethereum区块链系统扩展性的方法的一种优选方案，所述步骤S10中，所述数据操作函数包括插入函数、更新函数、删除函数和查询函数。

[0013] 作为提高Ethereum区块链系统扩展性的方法的一种优选方案，所述步骤S50中，当需要更新业务逻辑时，只需新建一个新智能合约继承所述第二智能合约的公共类，并根据新的业务需求对所述新智能合约进行修正后，将所述新智能合约部署到所述区块链中，前

端服务器将原来的所述子类的智能合约地址修改为所述新智能合约的地址，即可完成升级更新合约的操作。

[0014] 作为提高Ethereum区块链系统扩展性的方法的一种优选方案，所述步骤S50中，所述区块链需要调用所述第二智能合约时，只需通过所述第二智能合约的地址去指向需要执行的所述第二智能合约。

[0015] 本发明的有益效果：本发明通过对Solidity编写的智能合约进行功能上的分割，保证存储数据层面与逻辑层面的分离，同时使用抽象继承的设计思路，对智能合约的逻辑重复内容进行抽象化。当数据结构确定好之后，前端服务器对数据增删改查的操作请求都需要经过智能合约的逻辑层后才能执行对数据的操作，这种方法使得在业务变更频率过快、升级版本过快或者存储数据过大时，无需修改智能合约的数据层，只需通过智能合约的逻辑控制即可完成修改升级操作，从而避免存储数据的丢失或者需要进行繁琐的数据转移。

附图说明

[0016] 为了更清楚地说明本发明实施例的技术方案，下面将对本发明实施例中所需要使用的附图作简单地介绍。显而易见地，下面所描述的附图仅仅是本发明的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

[0017] 图1是本发明一实施例所述的提高Ethereum区块链系统扩展性的方法的流程图。

[0018] 图2是本发明一实施例所述的提高Ethereum区块链系统扩展性的方法的结构图(一)。

[0019] 图3是本发明一实施例所述的提高Ethereum区块链系统扩展性的方法的结构图(二)。

具体实施方式

[0020] 下面结合附图并通过具体实施方式来进一步说明本发明的技术方案。

[0021] 其中，附图仅用于示例性说明，表示的仅是示意图，而非实物图，不能理解为对本专利的限制；为了更好地说明本发明的实施例，附图某些部件会有省略、放大或缩小，并不代表实际产品的尺寸；对本领域技术人员来说，附图中某些公知结构及其说明可能省略是可以理解的。

[0022] 本发明实施例的附图中相同或相似的标号对应相同或相似的部件；在本发明的描述中，需要理解的是，若出现术语“上”、“下”、“左”、“右”、“内”、“外”等指示的方位或位置关系为基于附图所示的方位或位置关系，仅是为了便于描述本发明和简化描述，而不是指示或暗示所指的装置或元件必须具有特定的方位、以特定的方位构造和操作，因此附图中描述位置关系的用语仅用于示例性说明，不能理解为对本专利的限制，对于本领域的普通技术人员而言，可以根据具体情况理解上述术语的具体含义。

[0023] 在本发明的描述中，除非另有明确的规定和限定，若出现术语“连接”等指示部件之间的连接关系，该术语应做广义理解，例如，可以是固定连接，也可以是可拆卸连接，或成一体；可以是机械连接，也可以是电连接；可以是直接相连，也可以通过中间媒介间接相连，

可以是两个部件内部的连通或两个部件的相互作用关系。对于本领域的普通技术人员而言，可以具体情况理解上述术语在本发明中的具体含义。

[0024] 如图1至图3所示，本发明实施例中的提高Ethereum区块链系统扩展性的方法包括以下步骤：

[0025] S10、对数据结构进行分析，根据数据结构创建一个用于存储数据的第一智能合约，第一智能合约包括数据结构和数据操作函数；

[0026] S20、对第一智能合约进行部署，并记录第一智能合约地址；

[0027] S30、创建用于集成基础功能的第二智能合约，从第二智能合约中生成公共类，将第一智能合约地址导入公共类，并根据第一智能合约地址生成第二智能合约的一个对象，公共类将第一智能合约的功能集成为公共函数；

[0028] S40、创建子类继承公共类的属性和功能；

[0029] S50、子类在公共函数提供的功能基础上，调用数据操作函数建立符合业务需求的新函数，提供给业务所需的功能。

[0030] 本实施例中的公共类是指把智能合约所共有的属性和功能生成了一个公共类，并把数据合约的一个对象当做共有属性，拥有基础的增删改查的功能以及对应的逻辑功能。当子类继承智能合约的公共类时，子类将拥有公共类的属性和功能，并且子类还具有扩展性，可以新增自己独有的属性和自己的功能，只需根据业务需求增删改查即可。

[0031] 本发明通过对Solidity编写的智能合约进行功能上的分割，保证存储数据层面与逻辑层面的分离，同时使用抽象继承的设计思路，对智能合约的逻辑重复内容进行抽象化。只要一开始确定好数据结构，前端服务器对数据增删改查的操作请求都需要经过智能合约的逻辑层后才能执行对数据的操作，这种方法使得在业务变更频率过快、升级版本过快或者存储数据过大时，无需修改智能合约的数据层，只需通过智能合约的逻辑控制即可完成修改升级操作，从而避免存储数据的丢失或者需要进行繁琐的数据转移。

[0032] 于本实施例中，步骤S20中，第一智能合约通过区块链中的虚拟机将其编译为计算机底层的编译文件，并将编译文件发布至区块链中的全部节点，全部节点达成共识并存储第一智能合约，当区块链返回第一智能合约地址后，则说明第一智能合约部署完成。

[0033] 于本实施例中，步骤S10中，数据操作函数包括插入函数、更新函数、删除函数和查询函数。

[0034] 于本实施例中，步骤S50中，当需要更新业务逻辑时，只需新建一个新智能合约继承第二智能合约的公共类，并根据新的业务需求对新智能合约进行修正后，将新智能合约部署到区块链中，前端服务器将原来的所述子类的智能合约地址修改为所述新智能合约的地址，即可完成升级更新合约的操作。

[0035] 于本实施例中，步骤S50中，区块链需要调用第二智能合约时，只需通过第二智能合约的地址去指向需要执行的第二智能合约。

[0036] 专有名词的解释：

[0037] 智能合约：智能合约是一种旨在以信息化方式传播、验证或执行合同的计算机协议。智能合约允许在没有第三方的情况下进行可信交易，这些交易可追踪且不可逆转。

[0038] 公共类：也可以称为父类，父类具有众多属性和功能，在这里相当于一个智能合约。

[0039] 子类:指的是继承父类的一个类,子类包含着父类中所有的属性和功能,并且拥有对数据进行增删改查的能力。

[0040] 公共函数:函数指的是功能,公共函数指的就是父类中的功能,子类通过继承父类后都能获得的功能称为公共函数。

[0041] 业务逻辑:指的是开发人员根据业务的需求,为了实现具体的功能会进行逻辑的判断,例如,对传入字段进行判断、修改或者报错以及对权限的控制操作。

[0042] 节点:指的是区块链的系统节点,区块链是依靠节点去运行的。每个节点都有一份完整的区块链数据,且有权利去调起智能合约。而且每个节点具有审查监督区块链数据的任务,每次区块链的数据要发生改变时,其所有节点都必须对其结果进行认证,只有大部分的节点确认了这次结果,才会修改区块链的数据,以保证所有节点的数据的统一。

[0043] Solidity:Solidity是一种运行在以太虚拟机上的智能合约所用的高级编程语言。

[0044] 以太坊(Ethereum):以太坊(Ethereum)是一个开源的有智能合约功能的公共区块链平台,通过其专用加密货币以太币提供去中心化的以太虚拟机来处理点对点合约。

[0045] 以太虚拟机:以太坊虚拟机是以太网上智能合约的运行环境。以太坊虚拟机是在沙盒中运行,这是和区块链主链完全分开的,代码运行在以太坊虚拟机里没有网络,文件系统或是其他进程的智能合约会被限制访问其他的智能合约。

[0046] 抽象继承:指的是从众多的事物中抽取出共同的特征。父类就是把共有的属性和功能抽象出来形成的一个类,继承指的是子类继承父类的功能和属性。

[0047] 需要声明的是,上述具体实施方式仅仅为本发明的较佳实施例及所运用技术原理。本领域技术人员应该明白,还可以对本发明做各种修改、等同替换、变化等等。但是,这些变换只要未背离本发明的精神,都应在本发明的保护范围之内。另外,本申请说明书和权利要求书所使用的一些术语并不是限制,仅仅是为了便于描述。

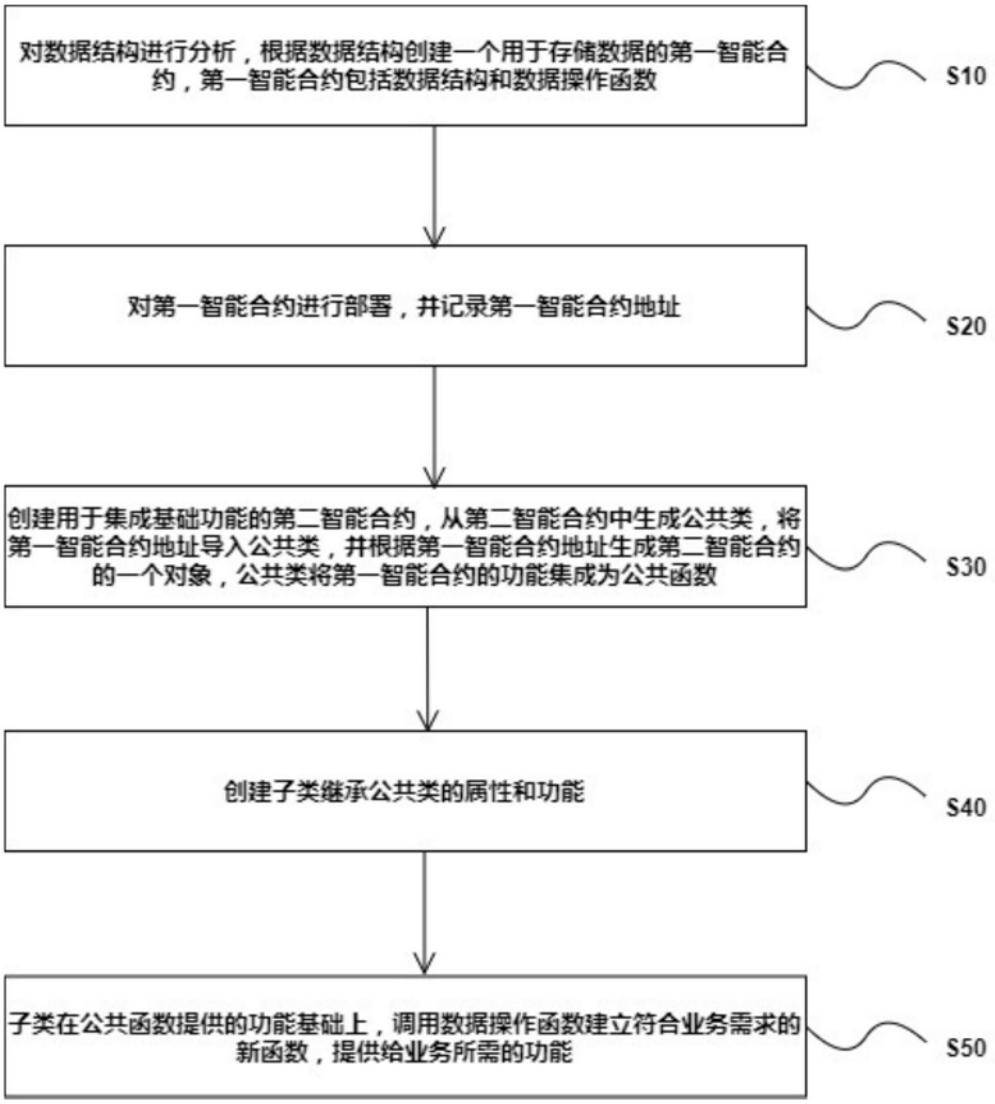

对数据结构进行分析，根据数据结构创建一个用于存储数据的第一智能合约，第一智能合约包括数据结构和数据操作函数
S10
对第一智能合约进行部署，并记录第一智能合约地址
S20
创建用于集成基础功能的第二智能合约，从第二智能合约中生成公共类，将第一智能合约地址导入公共类，并根据第一智能合约地址生成第二智能合约的一个对象，公共类将第一智能合约的功能集成为公共函数
S30
创建子类继承公共类的属性和功能
S40
子类在公共函数提供的功能基础上，调用数据操作函数建立符合业务需求的新函数，提供给业务所需的功能
S50

图1

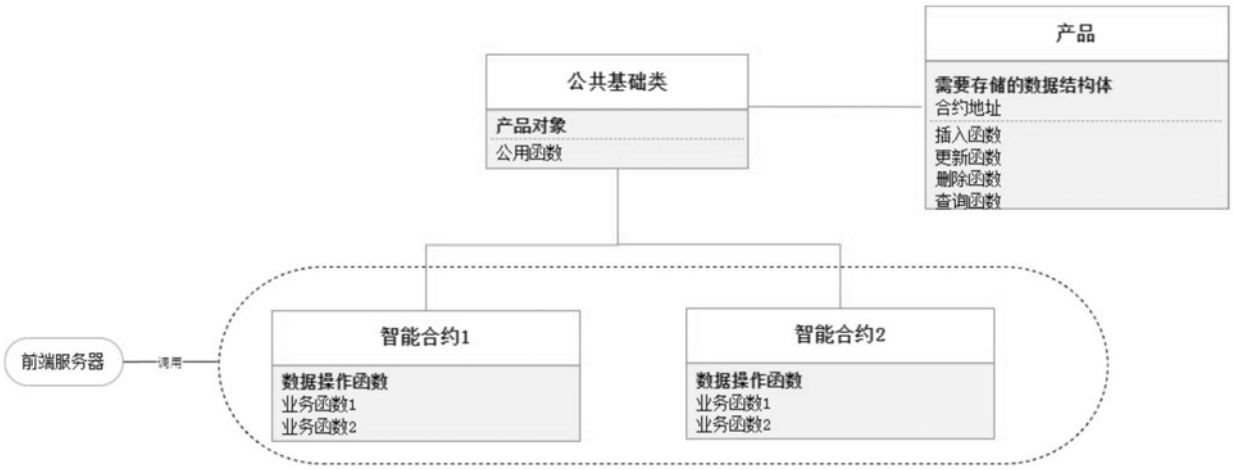

产品
需要存储的数据结构体
合约地址
插入函数
更新函数
删除函数
查询函数
公共基础类
产品对象
公用函数
前端服务器
调用
智能合约1
数据操作函数
业务函数1
业务函数2
智能合约2
数据操作函数
业务函数1
业务函数2

图2

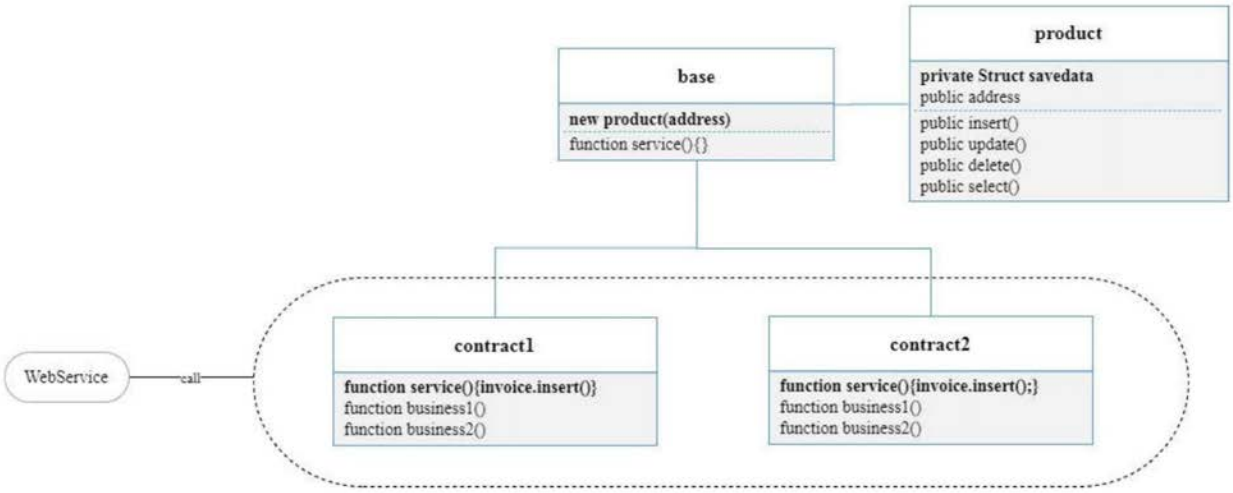

product
private Struct savedata
public address
public insert()
public update()
public delete()
public select()
base
new product(address)
function service(){}
contract1
function service(){invoice.insert()}
function business1()
function business2()
contract2
function service(){invoice.insert();}
function business1()
function business2()
WebService
call

图3