版本 <2.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| <16/10/2018> | <1.0> | <架构文档初稿> | <第四组全员> |
| <4/12/2018> | <2.0> | <架构文档修改> | <第四组全员> |
| <10/1/2019> | <3.0> | <架构文档修改> | <第四组全员> |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 范围 4

1.3定义、首字母缩写词和缩略语 4

1.4参考资料 4

2. 构架表示方式 4

3. 构架目标和约束 5

4. 用例视图 5

4.1 EHR管理 5

4.2 SQL操作记录查询 5

4.3 数据挖掘 5

5. 逻辑视图 6

5.1 三层架构 6

5.2 Application Layer（表现层） 7

5.3 业务逻辑层 7

5.4 数据层 8

6. 部署视图 9

7. 数据视图 10

8. 进程试图 11

9. 质量 11

# 1. 简介

## 1.1 目的

本文档将从构架方面对系统进行综合概述，其中会使用多种不同的构架视图来描述系统的各个方面。它用于记录并表述已对系统的构架方面作出的重要决策。

## 1.2 范围

本文档将用于第4组正在开发中的基于区块链的数据共享系统，该系统是为一个区域内的所有愿意 共享自己的数据的机构设计的共享平台，注重数据的隐私安全，提供查询其他机构的数据、身份验 证、权限认证等功能，并可以部署数据挖掘应用。本文介绍时以医疗机构之间共享电子病历为实际 应用场景进行详述。

## 1.3定义、首字母缩写词和缩略语

**区块链（英语：blockchain或block chain）：**是借由密码学串接并保护内容的串连交易[记录](https://zh.wikipedia.org/wiki/%E8%AE%B0%E5%BD%95)（又称区块）。每一个区块包含了前一个区块的[加密散列](https://zh.wikipedia.org/wiki/%E5%AF%86%E7%A2%BC%E9%9B%9C%E6%B9%8A%E5%87%BD%E6%95%B8)、相应时间戳记以及交易数据（通常用[默克尔树](https://zh.wikipedia.org/w/index.php?title=%E9%BB%98%E5%85%8B%E7%88%BE%E6%A8%B9&action=edit&redlink=1)算法计算的散列值表示），这样的设计使得区块内容具有难以篡改的特性。用区块链所串接的分布式账本能让两方有效纪录交易，且可永久查验此交易。

**Smart Contract:**智能合约（英语：Smart contract ）是一种旨在以信息化方式传播、验证或执行[合同](https://zh.wikipedia.org/wiki/%E5%90%88%E5%90%8C)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。智能合约概念于1994年由[Nick Szabo](https://zh.wikipedia.org/wiki/Nick_Szabo)首次提出。智能合同的目的是提供优于传统合同方法的安全，并减少与合同相关的其他[交易成本](https://zh.wikipedia.org/wiki/%E4%BA%A4%E6%98%93%E6%88%90%E6%9C%AC)。

**EHR：**电子健康纪录（Electronic Health Record），又称为电子健康档案，简称EHR，是电子化的个人健康纪录（病历、心电图、医疗视频等），电子健康纪录可以经由计算机或网上访问，可以包含现今与过去个人的健康信息。除此之外，电子健康纪录也能够包括医学相关的引用数据、医疗处置、药物使用、人口统计数据、其他与非医疗的管理数据等等。然而，一个理想的电子健康纪录系统，现今还没有一个软件或是供应商所能够建置出来。

**SQL：**是一种[特定目的编程语言](https://zh.wikipedia.org/wiki/%E7%89%B9%E5%AE%9A%E7%9B%AE%E7%9A%84%E7%A8%8B%E5%BC%8F%E8%AF%AD%E8%A8%80)，用于管理[关系数据库管理系统](https://zh.wikipedia.org/wiki/%E5%85%B3%E7%B3%BB%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F)（RDBMS），或在[关系流数据管理系统](https://zh.wikipedia.org/wiki/%E5%85%B3%E7%B3%BB%E6%B5%81%E6%95%B0%E6%8D%AE%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F)（RDSMS）中进行流处理。SQL基于[关系代数](https://zh.wikipedia.org/wiki/%E5%85%B3%E7%B3%BB%E4%BB%A3%E6%95%B0_(%E6%95%B0%E6%8D%AE%E5%BA%93))和[元组关系演算](https://zh.wikipedia.org/wiki/%E5%85%83%E7%BB%84%E5%85%B3%E7%B3%BB%E6%BC%94%E7%AE%97)，包括一个[数据定义语言](https://zh.wikipedia.org/wiki/%E6%95%B0%E6%8D%AE%E5%AE%9A%E4%B9%89%E8%AF%AD%E8%A8%80)和[数据操纵语言](https://zh.wikipedia.org/wiki/%E6%95%B0%E6%8D%AE%E6%93%8D%E7%BA%B5%E8%AF%AD%E8%A8%80)。SQL的范围包括数据插入、查询、更新和删除，[数据库模式](https://zh.wikipedia.org/w/index.php?title=%E6%95%B0%E6%8D%AE%E5%BA%93%E6%A8%A1%E5%BC%8F&action=edit&redlink=1)创建和修改，以及数据访问控制。尽管SQL经常被描述为，而且很大程度上是一种[声明式编程](https://zh.wikipedia.org/wiki/%E5%A3%B0%E6%98%8E%E5%BC%8F%E7%BC%96%E7%A8%8B)（[4GL](https://zh.wikipedia.org/wiki/%E7%AC%AC%E5%9B%9B%E4%BB%A3%E7%A8%8B%E5%BC%8F%E8%AA%9E%E8%A8%80)），但是其也含有[过程式编程](https://zh.wikipedia.org/wiki/%E8%BF%87%E7%A8%8B%E5%BC%8F%E7%BC%96%E7%A8%8B)的元素。

## 1.4参考资料

1.基于区块链的数据共享系统需求规约，2.0版，高级软件开发课程第4组

2.基于区块链的数据共享系统立项建议书，高级软件开发课程第4组

3.RUP的软件架构文档模板

# 2. 构架表示方式

本文档将通过以下一系列视图来表示本系统的软件架构：用例视图，逻辑视图，部署视图，数据视 图。

# 3. 构架目标和约束

1.系统在开发过程中有如下设计约束：开发语言为Sodility，Javascript，采用面向对象方法的 开发方法，采用3-tier架构。

2.系统必须保证数据的安全，用户需要通过身份认证和权限认证才可以向其他医院申请访问数据。

3.实现系统的功能需求和非功能需求。

# 4. 用例视图



图1 系统用例图(以EHR数据共享为实际背景)

## 4.1 EHR管理

EHR管理用实现了为医疗从业人员提供EHR管理功能。医疗从业人员通过交互界面，登陆EHR 管理平台，完成EHR查询和EHR更新操作。EHR的所有管理操作，均会记录在分类账目中。

用户需要查询非本地的EHR数据时，向区块链网络发送SQL请求，当通过身份验证和权限验证后， 即可获得链上其他医疗机构的相关EHR数据。用户界面只需输入参数，具体的SQL查询语句会由智 能合约补全。

## 4.2 SQL操作记录查询

用户可以查询所有SQL请求操作，掌握数据的流向，保证数据的安全性。它们被记录在块上的分类 账中。

## 4.3 数据挖掘

用户可以选择将查询到的数据进行数据挖掘(具体细节见基于区块链的数据共享系统需求规约，1.0 版)，获得相应可视化结果。

# 5. 逻辑视图

本章是对软件架构的逻辑视图的描述。

## 5.1 三层架构

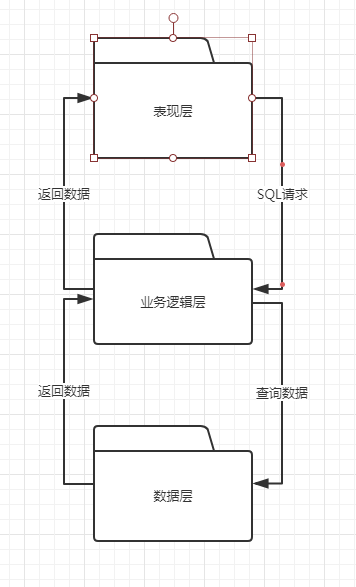


图2 系统抽象逻辑视图

系统采用3-tiers架构风格，分为表现层，业务逻辑层和数据层，用户可在表现层发起SQL请求， 表现层调用业务逻辑层（Smart Contract）进行身份和权限认证，再去调用数据层，请求其他块上的数据，得到数据后先返回业务逻辑层，将此次操作记录进分类账中，再将查到的数据展现给用户。

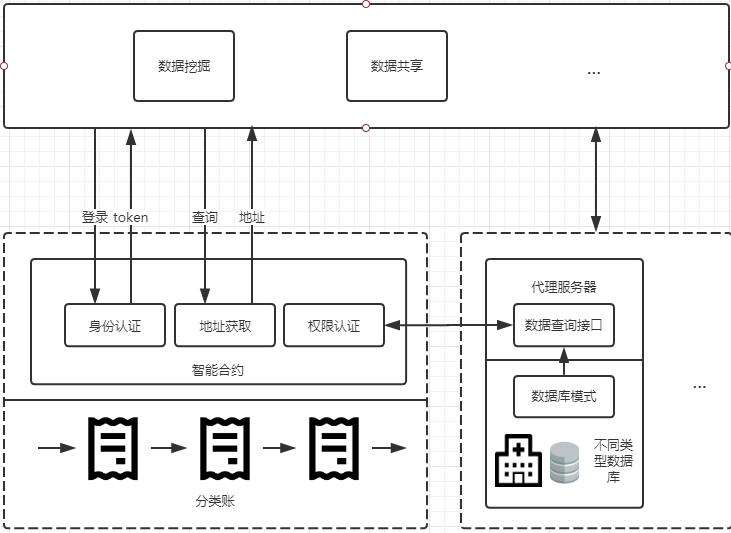


图3 系统具体逻辑视图

## 5.2 Application Layer（表现层）

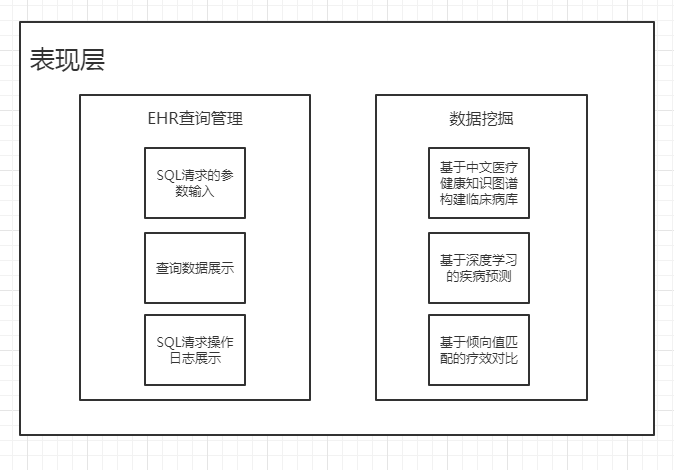


图4 表现层

在表现层中初定以web页面作为与用户交互的界面，目前暂定为两个功能，EHR查询和管理功能和对EHR进行数据挖掘的功能；页面负责接收用户输入的SQL请求的参数，或查询SQL操作日志的请求，并将查询到的结果展示给用户；数据挖掘操作用户可选，图中以挖掘EHR为例，部署了三种数据挖掘方法，将对应的结果可视化的展示在页面上。

## 5.3 业务逻辑层

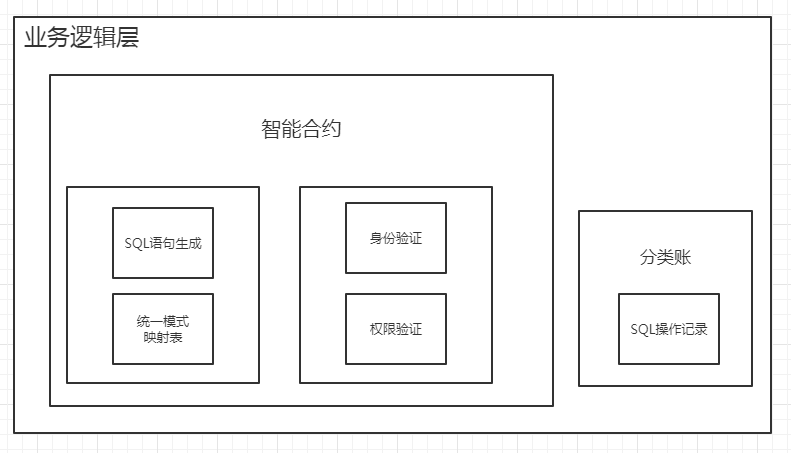


图5 业务逻辑层

本系统的业务逻辑主要依靠部署在各个块中，事先定义好的智能合约来实现，有三个主要的功能：统一模式映射，身份验证，权限认证。对每一次查询操作，将其视为一次交易记录在分类账中，以监控数据的流向和保证数据的安全性。

## 5.4 数据层

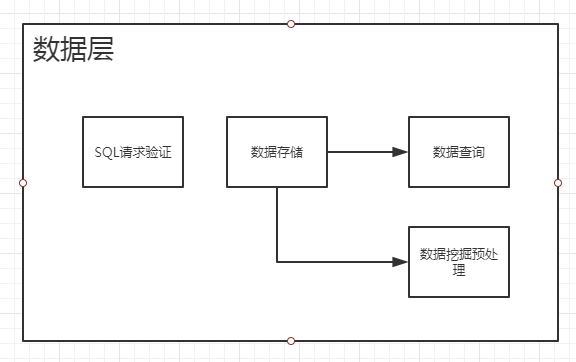


图6 数据层

基于医疗机构EHR共享背景为例，数据层会验证收到的SQL请求，通过验证后会开放自身查询EHR的接口，允许查询医疗机构自身已在维护的，存放有自己的EHR的本地数据库；对从其他医疗机构请求来的数据，会额外维护一个统一模式的数据库，将数据存入其中，供用户查看，或作为数据挖掘的数据集，进行一定的预处理，为数据挖掘功能服务；数据挖掘完成后会清除这些数据。

# 6. 部署视图

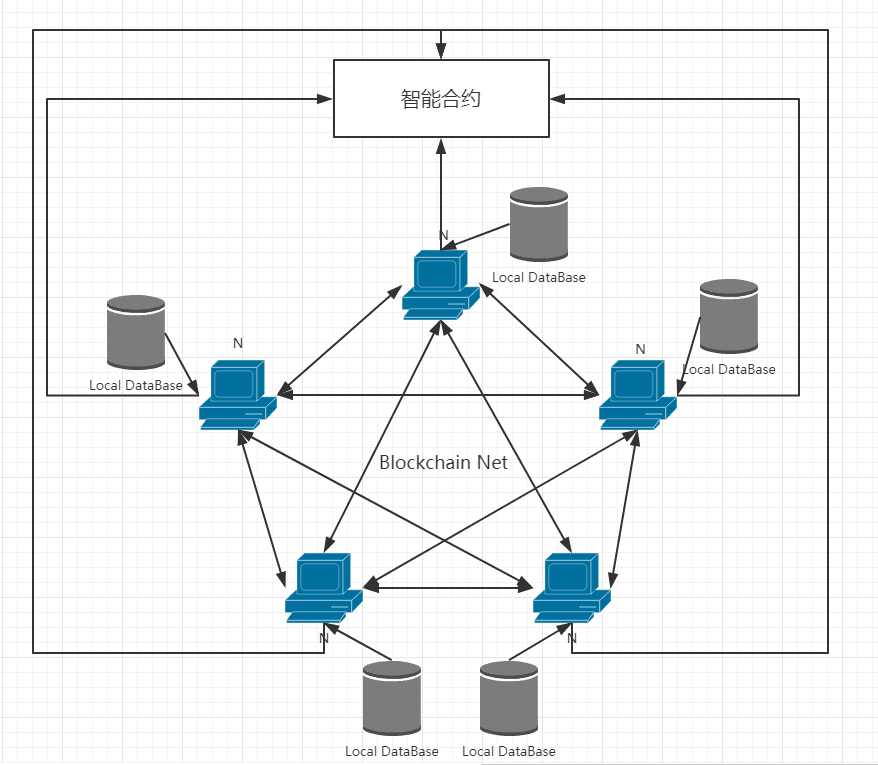


图7 系统部署视图

（1）节点N

每个机构为区块链上一个节点。新的医疗机构可以创建新的节点加入链中。

（2）本地数据库（Local DataBase）

该数据库是机构本身就已经在维护使用的数据库，存放有机构运营多年积累的数据。

（3）Unified Schema DataBase

统一模式数据库是针对医疗机构使用的不同数据库的统一查询的解决方法，发送SQL请求后获得的数据都会转为Unified Schema，存放在Unified Schema DataBase中，免去映射转化成Local DataBase表结构的时间，增加了效率和用户体验。

# 7. 数据视图

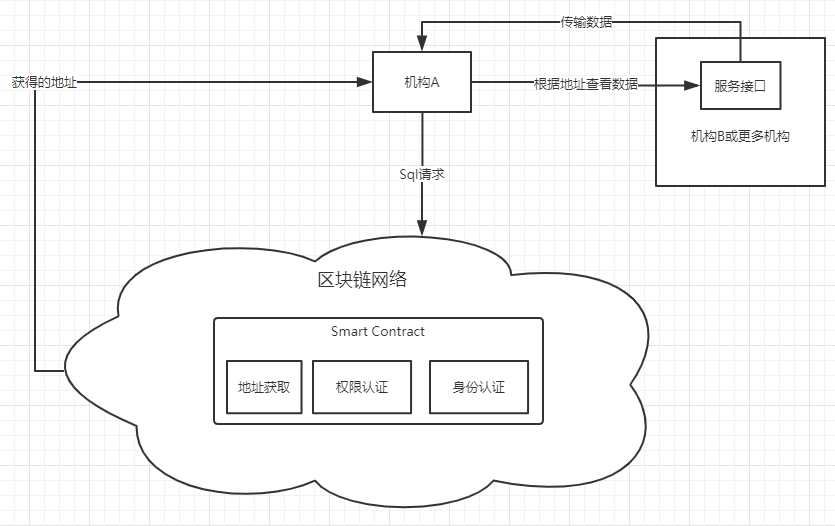


图8 系统数据流转视图(以EHR数据共享为实际背景)

# 8. 进程试图

当用户需要从其他节点的数据库中获取数据时，后台进程会去和智能合约进行交互，获取节点的地址列表，对于大数量级的数据需求，可以通过开多线程的方法，到多节点并发求取数据来减少时间，如数据库存取时间，数据传输时间等等，提升大数量级数据的共享效率，取得数据后再整合并返回给前端页面。

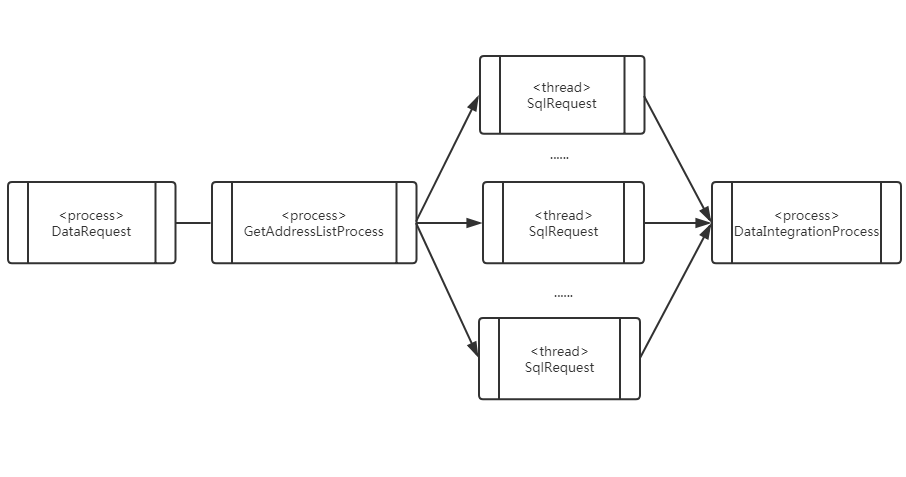


图9 数据获取进程视图

# 9. 质量

本系统采用的软件架构可以很好地支持系统质量方面的需求：

**8.1 易用性设计**

需求：系统应当方便所有用户的使用，培训时间短

设计：人机交互界面应简洁明了，并提供在线的支持帮助。

**8.2 可扩展性设计**

需求：添加新的智能合约，即添加新功能

设计：分离合约中的数据，使用一个单独的合约来存储数据，一个单独的合约编写业务逻辑。

**8.3 性能设计**

需求：查询请求可以在5秒内得到结果。

设计：最大化利用多线程到多节点求取数据，减少时间消耗。

**8.4 兼容性设计**

需求：系统支持在所有主流操作系统（Windows，Linux，MacOs）上运行

设计：搭建区块链时加入使用不同操作系统的节点。