# ITC项目进展

## 概述

为了提升传统物联网的安全性，降低网络的运维成本，提出了ITC（IoT Chain）万物互联的构想。本项目使用的技术包括区块链、嵌入式开发、云服务配置、手机应用开发等。本文将从系统设计、应用场景、阶段规划三方面对方案进行阐述。

## 系统设计

### 系统架构组成

系统架构图如图1所示。

系统硬件包含云端服务器、手机和路由器、装载项目芯片的物联网设备等。基于设备用途、芯片自身容量、服务器通讯可行性的考虑，将搭载芯片的物联网划分为轻节点，将手机和路由器划分为重节点。轻节点可以发起交易、账本验证；重节点可以存储账本、发起交易、账本验证、交易验证、pow计算、传递轻节点的交易请求、监听服务器指令；服务器可以存储账本、监听重节点的交易请求。

系统软件包含云端服务器的伺服软件、设备芯片的通讯及区块链固件、用户交互的手机App、路由器应用软件。



图1 系统架构图

### 系统功能

系统操作按执行时间，可分为三个阶段：初始阶段，组网阶段，交易阶段。

|  |  |
| --- | --- |
| 功能 | 说明 |
| 服务器构建代币池 | 使用ERC20标准构建代币池，将代币数据存储于服务器数据库中 |

表1 初始阶段功能表

|  |  |
| --- | --- |
| 功能 | 说明 |
| 轻重节点组网 | 步骤1：每个节点监听并搜寻蓝牙通讯范围内的设备，通讯获取路由信息，组成物理子网络（每个物理子网络至少包含一个重节点）。轻节点间采用自动匹配方式，轻重节点、重节点间采用手动匹配方式。  步骤2：每个节点生成非对称公私钥对，并向物理子网络内所有重节点发送节点属性、公钥。  步骤3：每个重节点向服务器发送节点信息列表。保持一定间隔的动态更新。  步骤4：每个重节点建立本地DAG和账本。 |
| 服务器入网 | 步骤1：监听各个物理子网络重节点的反馈，获取该物理子网络反馈的节点列表最小集合，包含轻重属性。在一定时限内有效。  步骤2：主动询问各物理子网络的重节点，获取节点列表最小集合。保持一定间隔的动态更新。  步骤3：建立主链，初始化全局账本。 |

表2 组网阶段功能表

|  |  |
| --- | --- |
| 功能 | 说明 |
| 轻节点发起本地交易 | 步骤1：轻节点生成新交易C，先对新交易进行账本验证（有价交易的余额判断），再将新交易反馈给第一重节点。  步骤2：该重节点从本地DAG中获取未验证的两笔交易A、B，进行pow值计算、交易验证（有价交易的非对称验证，针对属于本地物理子网络的交易节点）、账本验证（有价交易的余额判断，针对属于本地物理子网络的交易节点），再将A、B和验证结果传递给服务器。  步骤3：服务器将A、B、新交易C传递给下一个后续重节点。  步骤4：重复步骤2、3，直至物理子网络内全部重节点完成或服务器将A、B加入主链完成。  步骤5：服务器进行交易验证（有价交易的非对称验证）、账本验证（有价交易的余额判断），并统计获取重节点反馈，若满足一定百分比则将A、B加入主链、更新主账本，否则不加入主链，将结果告知该物理子网络所有重节点。在一定时限内有效。  步骤6：该物理子网络的所有重节点依据服务器反馈，更新本地DAG、交易源和目标的账本。  步骤7：重节点反馈交易源轻节点，轻节点更新本节点账本。 |
| 重节点发起本地交易 | 步骤1：重节点生成新交易C，对新交易进行账本验证（有价交易的余额判断）。  步骤2~6：同轻节点发起本地交易（步骤2~6） |
| 轻节点发起异地交易 | 步骤1~5：同轻节点发起本地交易（步骤1~5）  步骤6：该物理子网络的所有重节点依据服务器反馈，更新本地DAG。  步骤7：重节点反馈交易源轻节点，轻节点更新本节点账本。  步骤8：服务器将交易内容告知交易目标所属物理子网络的所有重节点。  步骤9：该物理子网络的所有重节点更新交易目标的账本，反馈服务器交易状态。  步骤10：服务器告知交易源所属物理子网络的所有重节点。  步骤11：该物理子网络的所有重节点更新交易源的账本，反馈服务器交易状态。 |
| 重节点发起异地交易 | 步骤1：重节点生成新交易C，对新交易进行账本验证（有价交易的余额判断）。  步骤2~6：同轻节点发起异地交易（步骤2~6）  步骤7~10：同轻节点发起异地交易（步骤8~11） |
| 轻节点查询状态 | 步骤1：轻节点将请求发送给重节点。  步骤2：重节点向服务器发起查询请求。  步骤3：服务器反馈查询结果。  步骤4：重节点反馈给轻节点。 |
| 重节点查询状态 | 步骤1~2：同轻节点查询状态（步骤2~3） |

表3 交易阶段功能表

### 数据交互

|  |  |  |
| --- | --- | --- |
| 数据流向 | 阶段 | 流量 |
| 轻节点->轻节点 | 组网 | 轻 |
| 重节点->轻节点 | 交易 | 轻 |
| 轻节点->重节点 | 交易 | 轻 |
| 重节点->重节点 | 交易 | 轻 |
| 轻节点->服务器 | 无 | 无 |
| 重节点->服务器 | 组网、交易 | 重 |
| 服务器->轻节点 | 无 | 无 |
| 服务器->重节点 | 组网、交易 | 重 |

表4 数据交互表

## 应用场景

|  |  |
| --- | --- |
| 场景 | 数据流向 |
| 代币生成 | 服务器 |
| 代币转账（充值） | 服务器->重节点  重节点->服务器  重节点->轻节点 |
| 代币消费 | 轻节点  重节点 |
| 节点报警（无代币、节点故障） | 轻节点->重节点  重节点->服务器 |

表5 应用场景表

## 研发内容

|  |  |  |
| --- | --- | --- |
| 设备 | 研发内容 | 研发平台 |
| 轻节点（芯片） | 1. 蓝牙通讯自动组网固件 2. 非对称算法生成密钥对 3. 代币钱包对称算法加密存储 4. 报警逻辑（无代币、节点故障） 5. 生成交易和接收币值更新逻辑 6. 节点功能逻辑 7. 固件远程更新 | 嵌入式硬件仿真平台  C语言  Windows操作系统 |
| 重节点（手机） | 1. 蓝牙通讯手动组网、3G通讯 2. 非对称算法生成密钥对 3. 代币钱包对称算法加密存储 4. 报警逻辑（无代币、节点故障） 5. 与轻节点的信息交互、与服务器的信息交互、生成交易、交易验证（非对称算法验签）、pow计算、账本验证（余额判断）、未验证交易选择、本地DAG和本地账本的加载存储更新 6. 节点功能逻辑 | Iphone/Android手机  IOS/Android开发平台  IOS操作系统 |
| 服务器 | 1. ERC20代币构建 2. 处理重节点的交易共识。 3. 交易验证（非对称算法验签）、账本验证（余额判断）、主链和全局账本的加载存储更新。 4. 账本的同步机制 5. 与重节点的信息交互 | Fabric  NodeJS |

## 后期工作

第一期主要参考轻节点单向指令、通讯数据量和存储需求少的特性，辅以简单的报警反馈，受限于低速低带宽的蓝牙通讯，可涵盖的场景包括灯节点网络等。第二期采用高速高带宽的wifi通讯，配置较大的存储空间，可涵盖的场景包括摄像头节点网络等。第三期可在原有网络的基础上升级固件，加入轻节点双向征询，可涵盖的场景包括租赁网络等。

在设备方面，除了手机作为重节点外，可使用路由器、机顶盒等常待机设备进行二次开发。

## 问题总结

* 嵌入式设备、服务器选择何种通讯方式组网？

答：轻节点间使用蓝牙自动连接，重轻节点、重节点间使用蓝牙手动连接，更换DAG时需手动连接匹配。重节点与服务器采用3G连接，需保证重节点服务长期前台运行。通讯介质的稳定性（掉线问题）、性能待测。

* 重节点长时间在线的原因？

答：轻节点的交易验证需要重节点参与，并反馈给服务器，重节点的服务需要常驻，只能让Iphone手机的app常开，或者让Android手机的服务常开。

* 重节点相对静态化的原因？

答：若采用动态移动化的方式，那么进出不同的物理子网络需要重新入网、更新本地DAG账本，有背于良好的用户体验。静态化是相对的，一段时间内仅服务于一个物理子网络，直到手动更换物理子网络。

* 轻节点、重节点、服务器各放置什么样的交易和账本？

答：出于嵌入式设备的存储空间考虑，轻节点不放置交易列表，仅保存本节点账本；出于存储空间与服务器同步流量的考虑，重节点未保存全局交易和账本，仅保存由本物理子网络DAG节点发起的交易和节点账本；服务器保存全局交易和账本。

* 轻节点间的通讯场景

答：轻节点无需参与共识，除组网外无其他通讯场景，因此轻节点无需固定化，也无需长时间在线。

* 若本地物理子网络的交易量不大，如何加快交易验证速度？

答：可生成新的无价交易，激发重节点获取两个未验证交易进行验证。