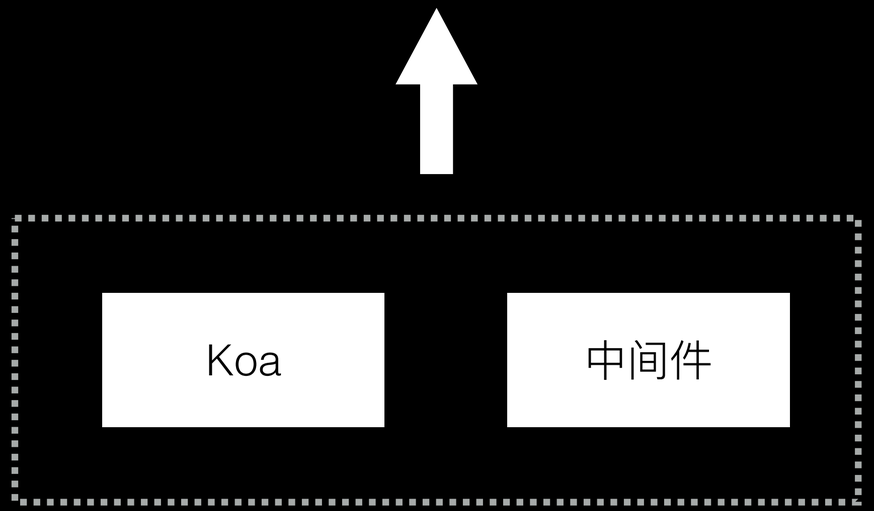
Node 在有赞的实践

KK

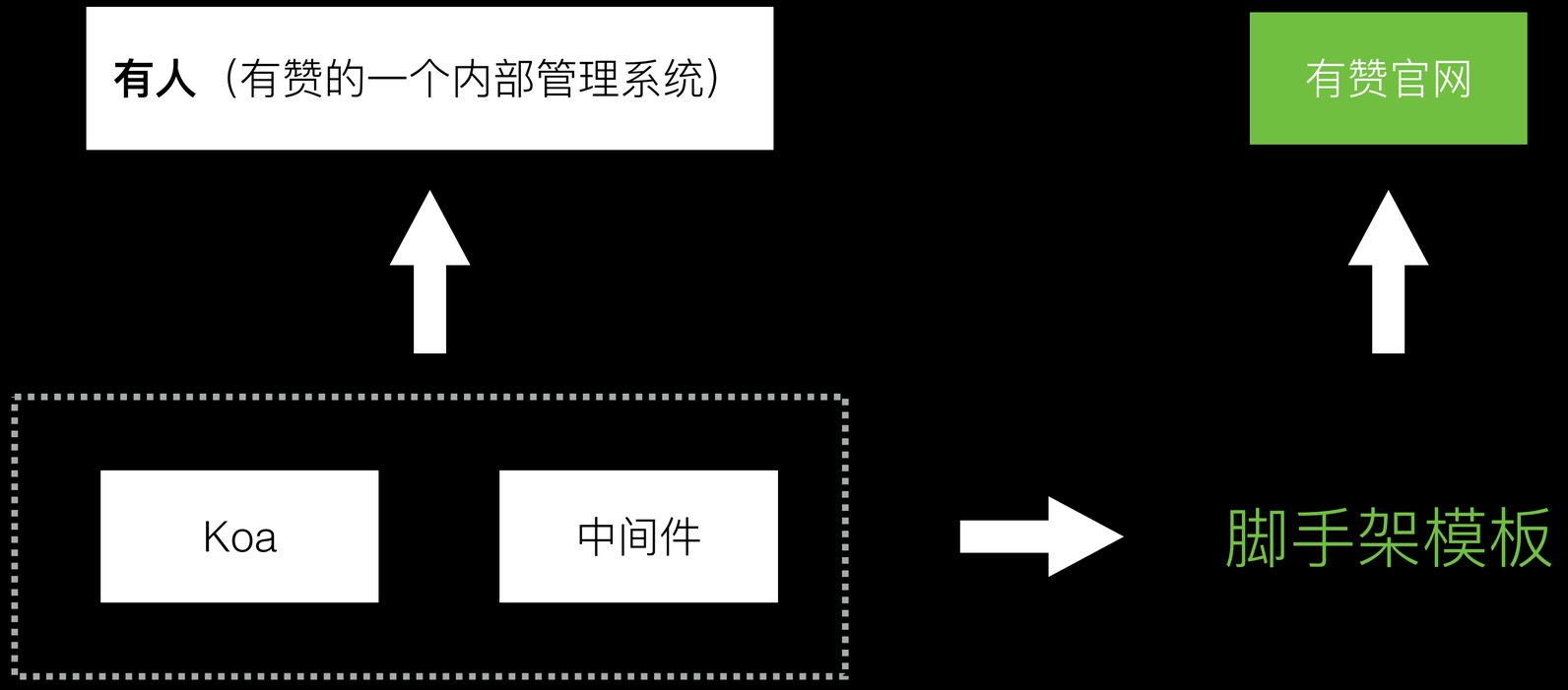
- 一、Node 基础框架的迭代与演进
- 二、Node 接入有赞服务化体系的历程
- 三、未来需要做的一些事情

第一个 Node 项目

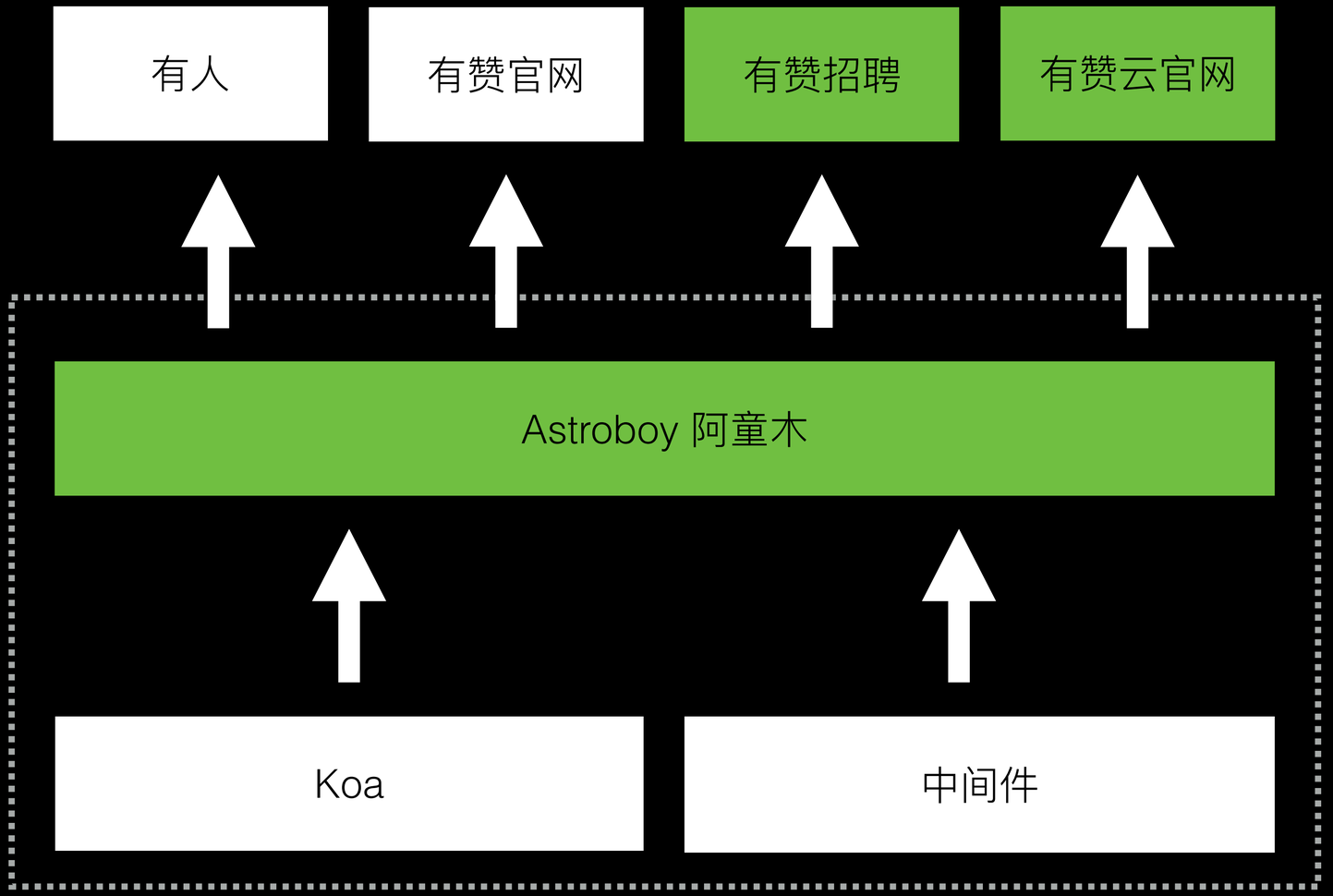
有人（有赞的一个内部管理系统）



第二个 Node 项目



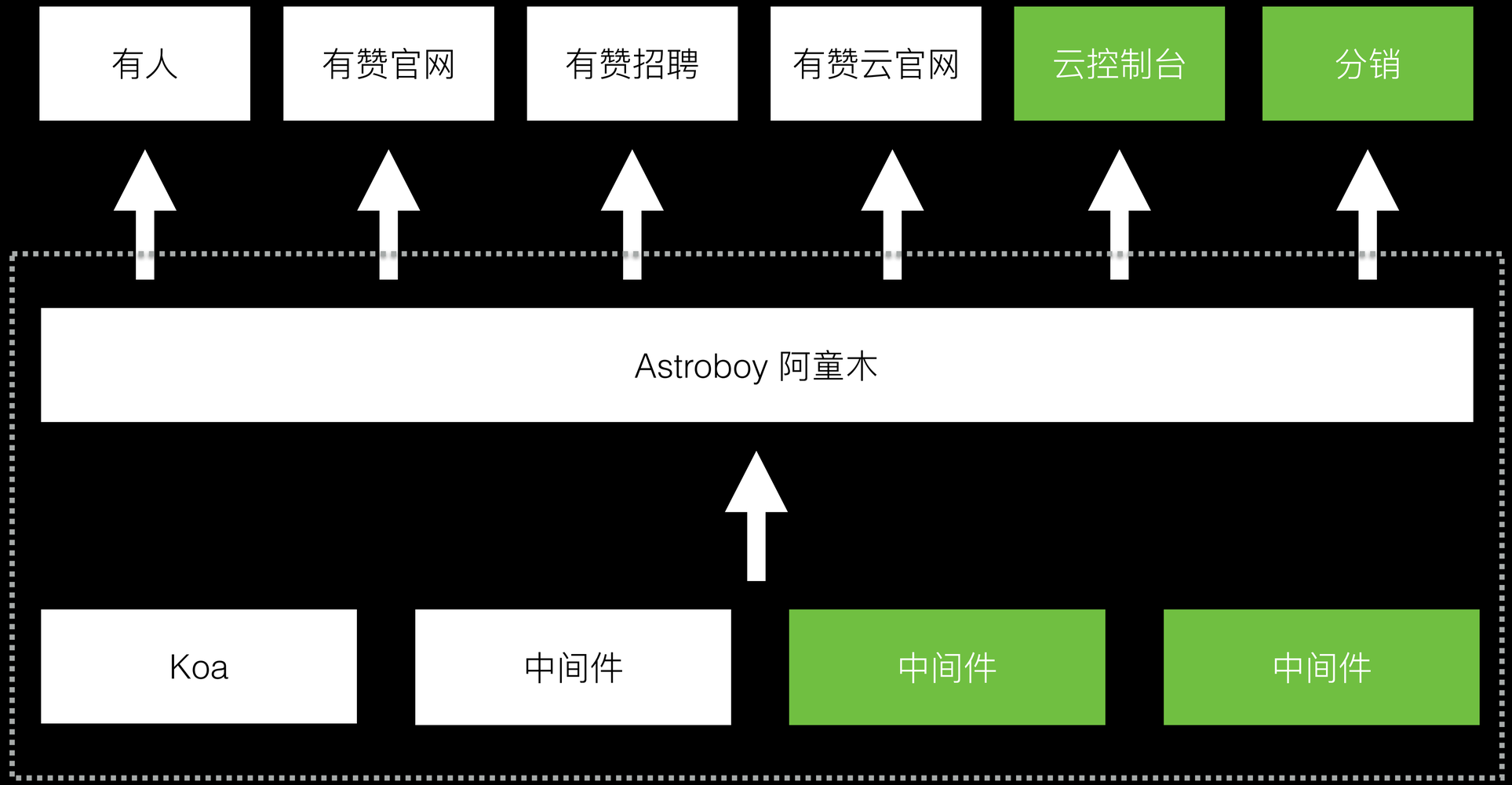
阿童木 0.0.1 诞生



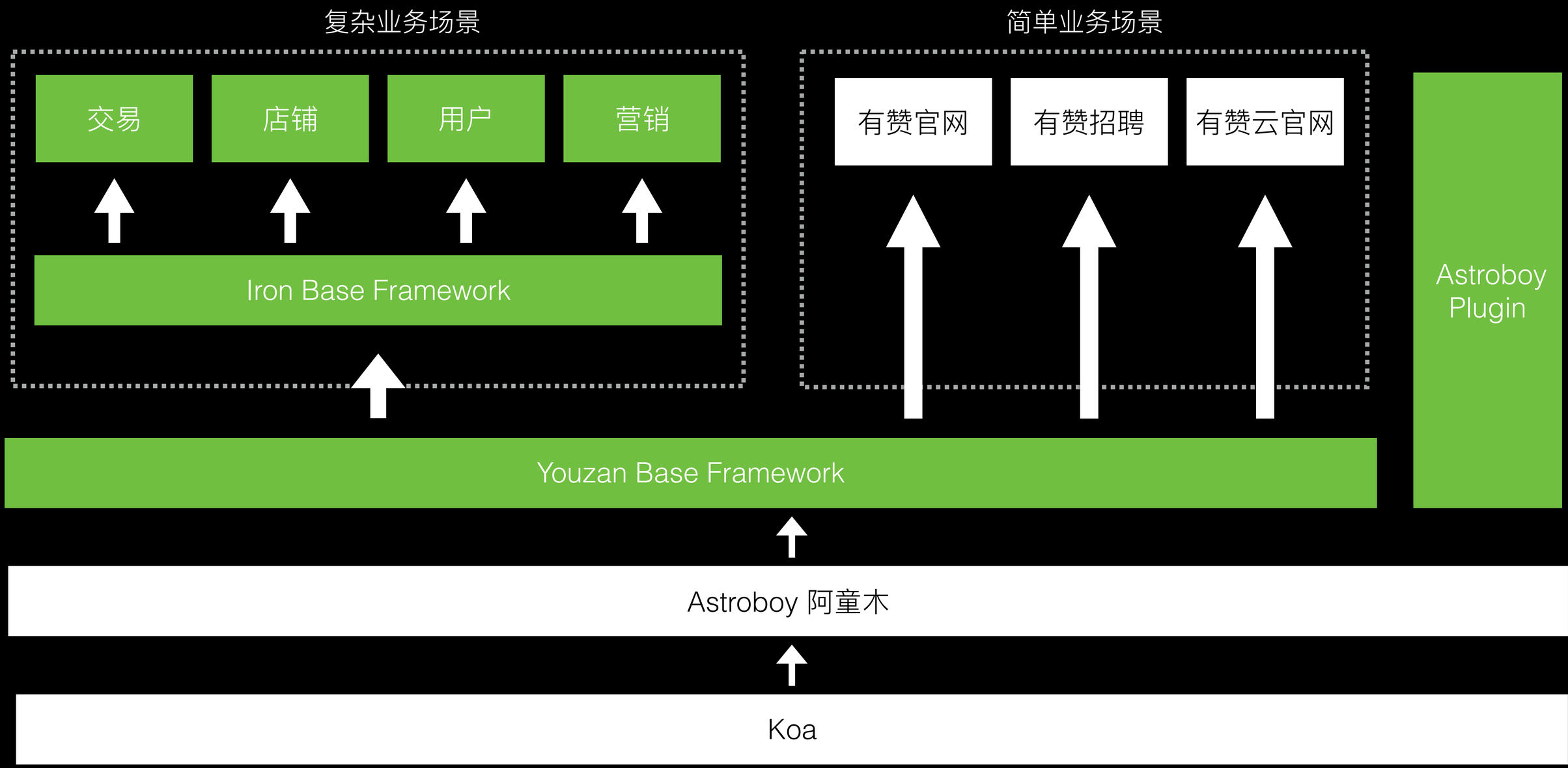
Koa + 中间件
脚手架模板

阿童木 0.0.1

很多项目都开始用 Node 了



阿童木 1.0.0 诞生

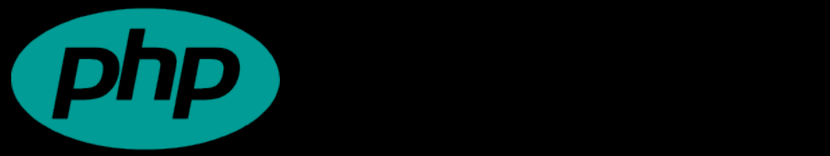

复杂业务场景
交易
店铺
用户
营销
Iron Base Framework
简单业务场景
有赞官网
有赞招聘
有赞云官网
Astroboy Plugin
Youzan Base Framework
Astroboy 阿童木
Koa

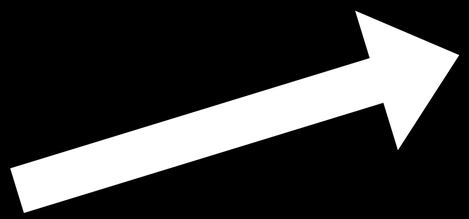
目前有哪些重要业务用了 Node

- 1、订单详情（灰度白名单）
- 2、下单（灰度白名单）
- 3、微页面（灰度白名单）
- 4、有赞云、有赞官网、分销等等

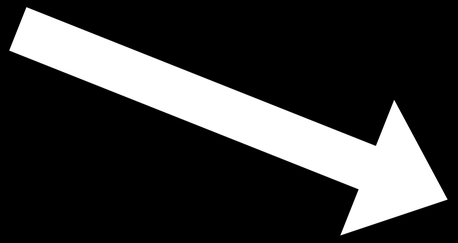
三、Node 接入有赞服务化体系的历程

服务化



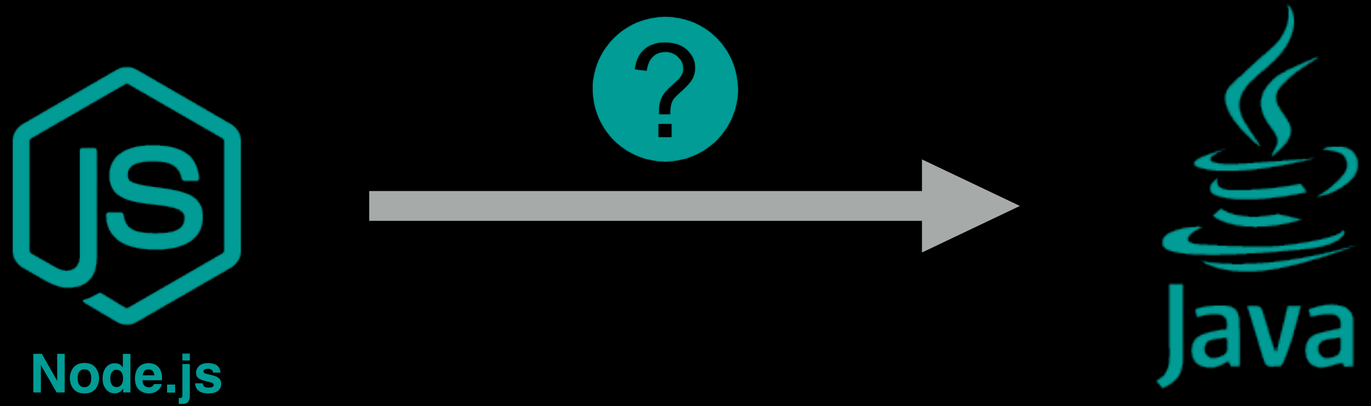


模板渲染
业务编排
接口转发



业务逻辑
服务接口

如何调用?



- 方案 1：Java 添加注解方式生成 Restful API
- 方案 2：Node 直接支持 Java Dubbo 接口调用
- 方案3：对方案 2 进行了优化

方案 1：Java 添加注解方式生成 Restful API

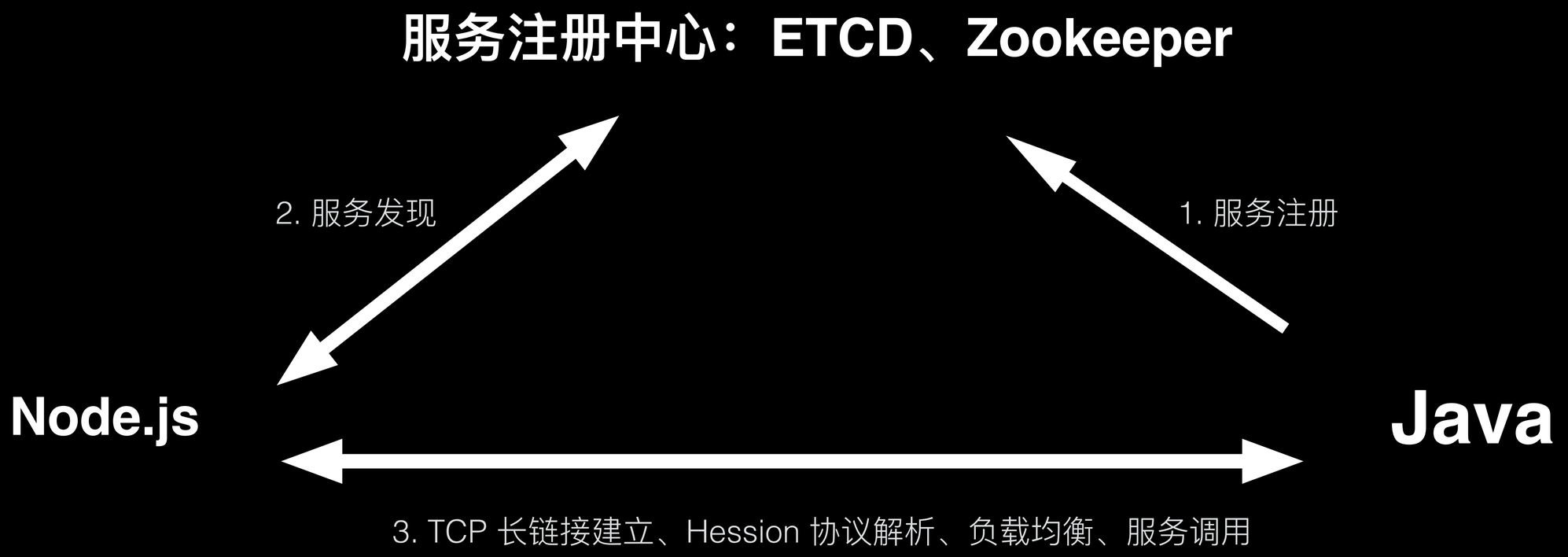
```
@POST
@Path("regist")
public String regist(User user) {
    ……
}
```

方案1 缺点

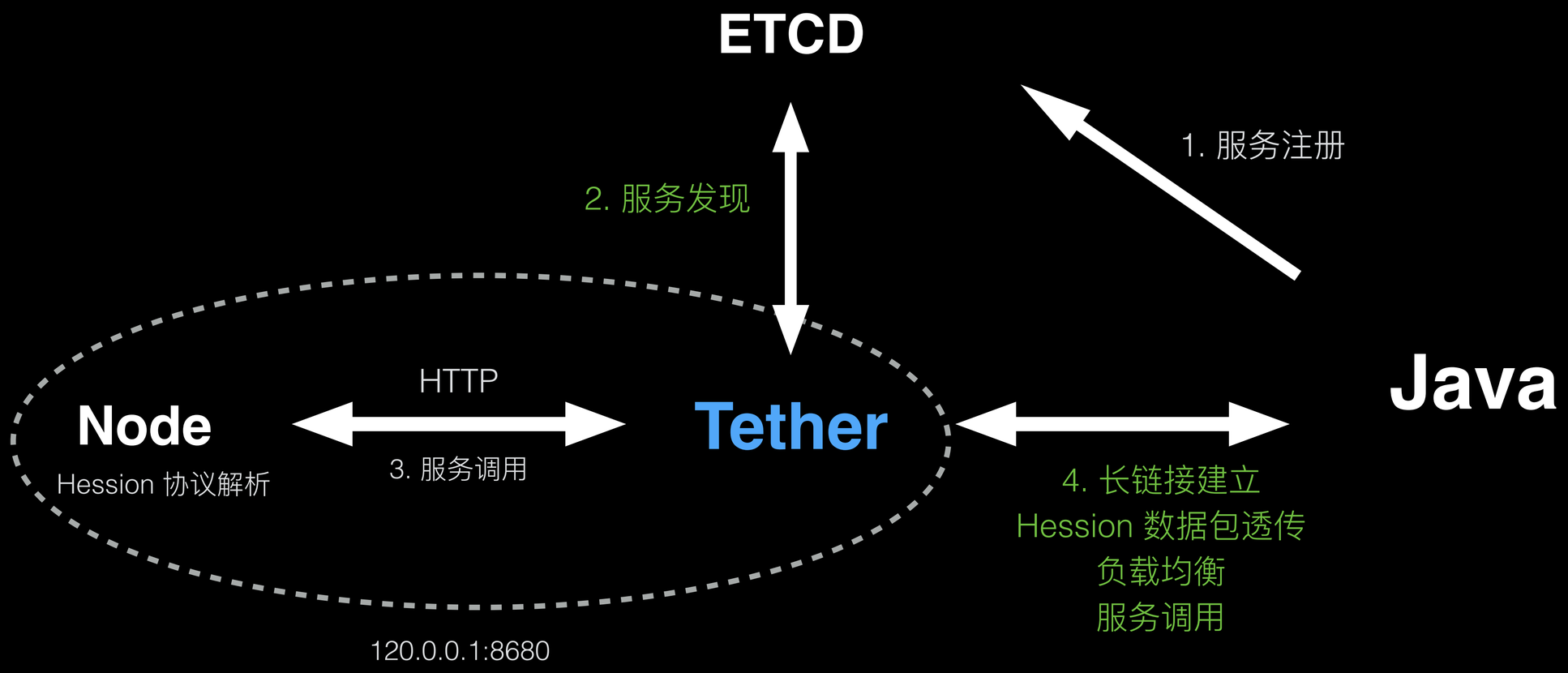
- 1. Java 开发需要添加**额外的注解**来生成 HTTP 接口；
- 2. 运维需要针对每个服务**配置域名**；
- 3. Node 需要维护一份很长的域名**配置文件**；
- 4. HTTP 方式调用，**性能**会低一些；

方案 2：Node 直接支持 Java Dubbo 接口调用

开源社区有哪些现成解决方案?



方案 2：Node 直接支持 Java Dubbo 接口调用



```
const Service = require('../base/BaseService');

class GoodsService extends Service {

  /**
   * 根据商品 alias 获取商品详情
   * @param  {String} alias 商品 alias
   */
  async getGoodsDetailByAlias(alias) {
    const result = this.invoke(
      'com.youzan.ic.service.GoodsService',
      'getGoodsDetailByAlias',
      [{
        $class: 'java.lang.String',
        $: 'nJDNtz2rl9'
      }]
    );
    return result;
  }

}

module.exports = GoodsService;
```

服务名

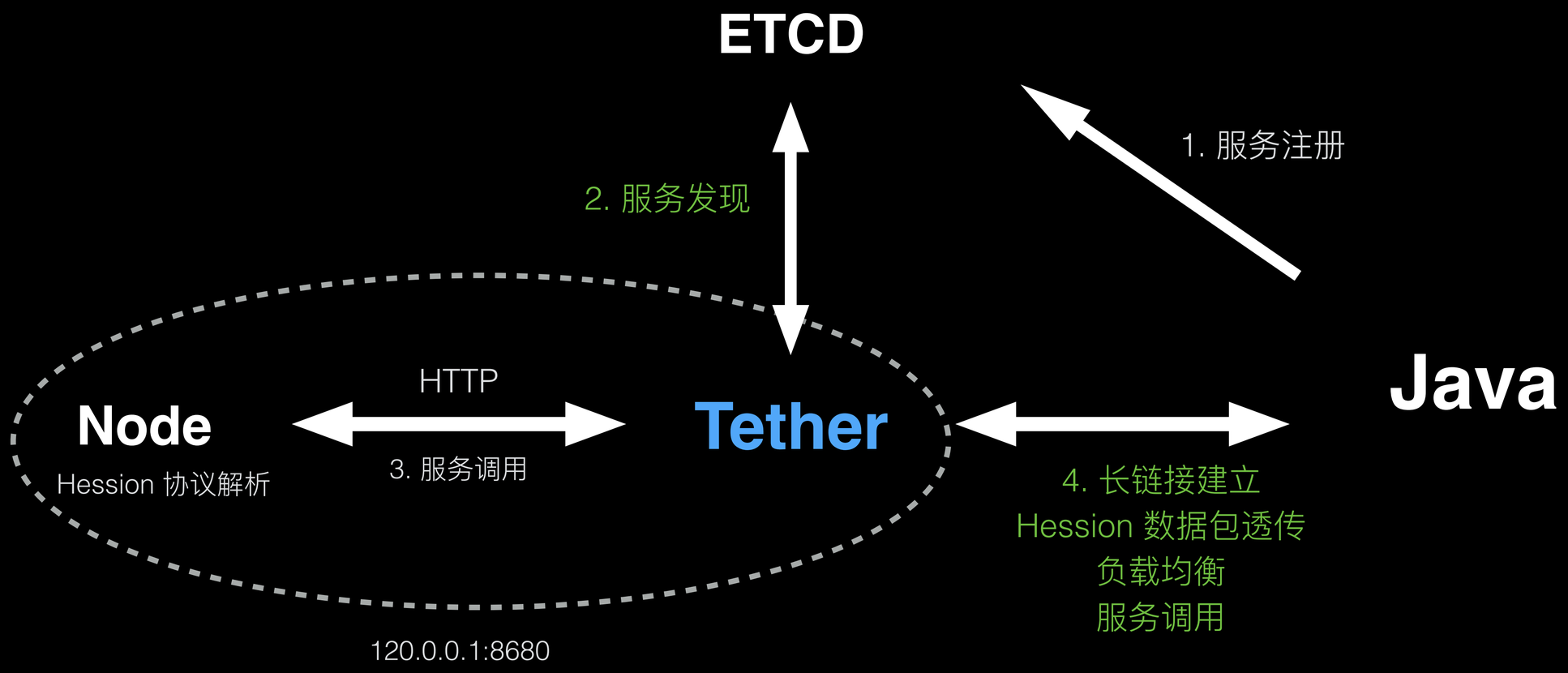
方法名

参数

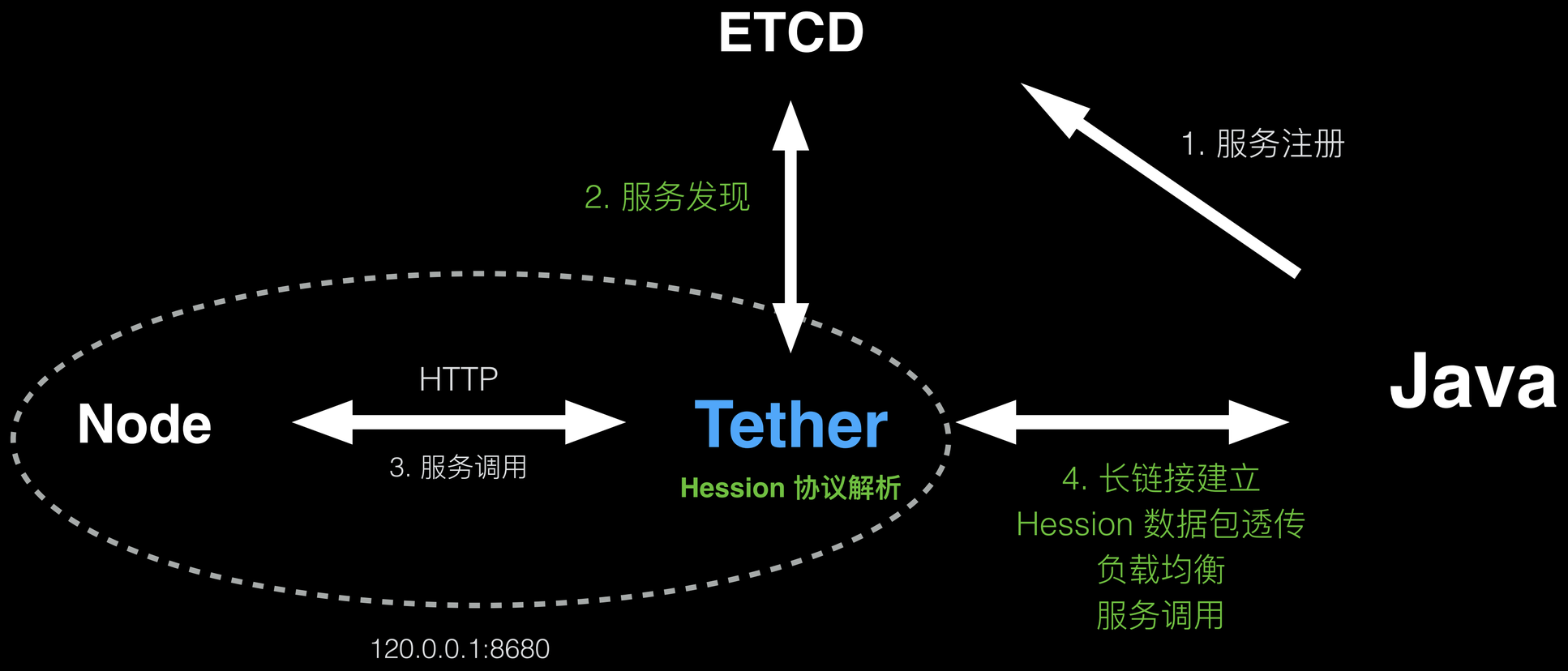
方案2 缺点

- 1. 对前端开发不友好，每个参数都需要**写明参数类型**。
- 2. Node 框架需要负责 **hession 协议解析**，对开发要求比较高；
- 3. 其他语言（PHP）也得实现一遍

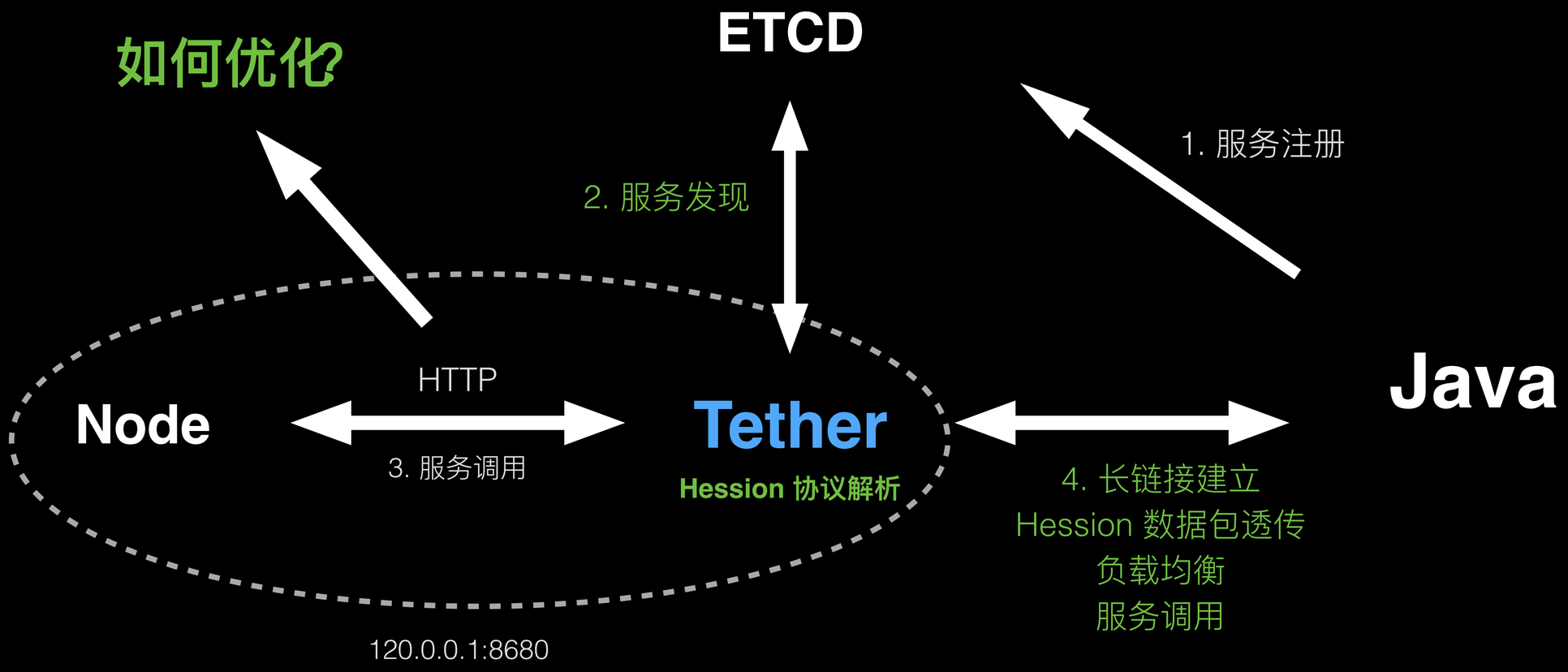
方案 2：Node 直接支持 Java Dubbo 接口的调用



方案3：对方案 2 进行了优化



方案3：对方案 2 进行了优化



Node 调用 Tether 如何优化的?

```
const Agent = require('agentkeepalive');

module.exports = new Agent({
  maxSockets: 100,
  maxFreeSockets: 10,
  timeout: 60000,
  freeSocketKeepAliveTimeout: 30000, // free socket keepalive for 30 seconds
});
```

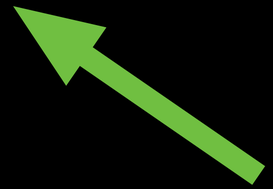
```
const Service = require('../base/BaseService');

class GoodsService extends Service {

  /**
   * 根据商品 alias 获取商品详情
   * @param  {String} alias 商品 alias
   */
  async getGoodsDetailByAlias(alias) {
    const result = this.invoke(
      'com.youzan.ic.service.GoodsService',
      'getGoodsDetailByAlias',
      [{
        $class: 'java.lang.String',
        $: 'nJDNtz2rl9'
      }]
    );
    return result;
  }

}

module.exports = GoodsService;
```

[alias]

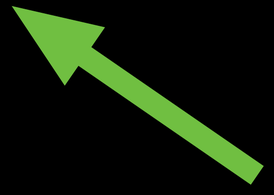
```
const Service = require('../base/BaseService');

class GoodsService extends Service {

  /**
   * 根据商品 alias 获取商品详情
   * @param  {String} alias 商品 alias
   */
  async getGoodsDetailByAlias(alias) {
    const result = this.invoke(
      'com.youzan.ic.service.GoodsService',
      'getGoodsDetailByAlias',
      [alias]
    );
    return result;
  }

}

module.exports = GoodsService;
```

```
curl -X POST \
  http://127.0.0.1:8680/soa/com.youzan.ic.service.GoodsService/
getGoodsDetailByAlias \
  -H 'content-type: application/json' \
  -H 'x-request-protocol: dubbo' \
  -H ‘X-Service-Chain: prj101’ \
  -d '["nJDNtz2rl9"]'
```

方法名



服务名



参数

方案3 优点

- 1. 使用简单（HTTP）、对前端开发友好；
- 2. 多语言接入成本低；
- 3. 后期更方便做协议层的优化；

三、未来需要做的一些事情

- Node 性能监控
- CPU Profile、GC Trace、堆快照、Heap Profile

Q & A