

5、Skywalking的埋点-插件化架构设计



rock_fish 关注

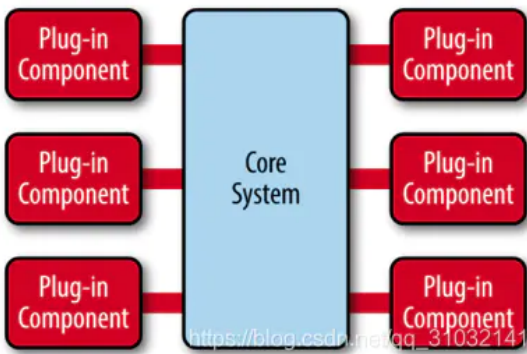
2021.05.22 16:45:01 字数 1,662 阅读 245

插件化架构设计

插件化架构另一种叫法是微内核架构，是一种面向功能进行拆分的可扩展性架构,通常用于存在多个版本、需要下载安装或激活才能使用的客户端应用；包含两类组件：核心系统（core system）和插件模块（plug-in modules）

- 核心系统 Core System 功能比较稳定，不会因为业务功能扩展而不断修改,通常负责和具体业务功能无关的通用功能，例如模块加载等
- 插件模块负责实现具体的业务逻辑，可以根据业务功能的需要不断地扩展，将变化部分封装在插件里面，从而达到快速灵活扩展的目的，而又不影响整体系统的稳定

如下图所示:



Skywalking的插件化实现

Skywalking中每个组件的埋点实现就是一个插件，针对有差异的版本也可能一个组件有多个实现，分别应对不同的版本。

组件加载管理的核心流程

Skywalking是对这些组件埋点插件的加载管理的大致流程如下：

1. 初始化配置，并明确插件所在的路径。
2. 在给定的路径下查找并解析 skywalking-plugin.def 插件文件。
3. 通过独立的类加载器 `AgentClassLoader` 加载插件类。
4. 通过 `PluginFinder` 对插件进行管理分类。
5. 使用 Byte Buddy 字节码增强库创建 `AgentBuilder`，其会根据已加载的插件动态增强目标类，插入埋点逻辑。
6. 使用 JDK的SPI 加载并启动 `BootService` 服务。`BootService` 就理解为通常我们编写的Service服务，就是功能的内聚封装。
7. 添加一个 JVM 钩子，在 JVM 退出时关闭所有 `BootService` 服务。

下边通过关键技术点的拆解，进一步理解上述的流程。



伦敦租房



rock_fish

总资产6

关注

21、Skywalking的埋点-Agent动态采样控制

阅读 219

22、skywalking的Trace数据协议

阅读 12

skywalking的日常维护1:
com.netflix.zuul.exception.ZuulExcept
阅读 41

推荐阅读

Java 操作 Linux 服务器 上传文件并执行脚本
阅读 774

JAVA进阶之Agent
阅读 451

Spring实现AOP的三种方式，你掌握了几种？
阅读 675

AOP
阅读 193

skywalking监控微服务
阅读 1,050

伦敦租房



写下你的评论...

评论0

赞

...



1. 解析agent/config/agent.config，获取配置，填充到Config中
2. 遍历环境变量（即 System.getProperties() 集合），查找"skywalking." 开头的配置 更新 Config中的值
3. 解析 Java Agent 的参数，更新Config中的值

2. 自定义ClassLoader

SkyWalking 的Agent 加载插件时使用到一个自定义的ClassLoader：AgentClassLoader，有了这个AgentClassLoader,就不会在应用的 Classpath 中引入 SkyWalking 的插件 jar 包，来达到对插件的管理让应用无依赖、无感知的目的。

AgentClassLoader扫描并加载指定Agent目录下的plugins和 activations 目录中的插件jar包。

3. 解析插件定义

当自定义类加载器扫描到插件jar，除了要加载类之外，还有最重要的一个目的，要明确的知道这个插件jar要增强组件中的哪些类以及如何增强，而承载这些信息的类被skywalking约定在每个插件jar包的resource目录下skywalking-plugin.def 文件中，其中其中每一行都是一个插件类的定义，如tomcat-7.x-8.x-plugin 插件中 skywalking-plugin.def 文件的内容如下：

```
1 | tomcat-7.x/8.x=org.apache.skywalking.apm.plugin.tomcat78x.define
2 | .TomcatInstrumentation
3 |
4 | tomcat-7.x/8.x=org.apache.skywalking.apm.plugin.tomcat78x.define
5 | .ApplicationDispatcherInstrumentation
```

插件类中指定了待增强的类、方法，以及增强的逻辑,体现在 `AbstractClassEnhancePluginDefine` 类以及其内部的几个方法上：

- `enhanceClass()` 方法：返回的 `ClassMatch`，用于匹配当前插件要增强的目标类。
- `define()`：插件类增强逻辑的入口，采用了模板方法的设计模式，实现中会调用下面的 `enhance()` 方法和 `witnessClass()` 方法。
- `enhance()`：真正执行增强逻辑的地方。
- `witnessClass()`：一个开源组件可能有多个版本，通过该方法识别组件的不同版本，防止对不兼容的版本进行增强。
- `getConstructorsInterceptPoints()`：构造方法的切入点，并指定增强的逻辑实现
- `getInstanceMethodsInterceptPoints()`：实例方法切入点，并指定增强的逻辑实现
- `getStaticMethodsInterceptPoints()`：静态方法切入点，并指定增强的逻辑实现

4. PluginFinder

JavaAgent 启动时挂在的运行机制是当一个类被首次加载的时候，会给我们机会进行字节码增强处理，即回调 `transform` 方法，在这个方法中传入被加载，待增强类，sw需找到跟这个类匹配的所有的插件类，这个工作就由PluginFinder来完成。简单来说就是根据传入的新加载的类，查找与其匹配的执行增强处理的 `AbstractClassEnhancePluginDefine` 集合。

5. AgentBuilder

AgentBuilder 是 Byte Buddy 库专门用来支持 Java Agent 的一个 API，其作用是方便使用者配置增强哪些类，忽略哪些package，对于要增强的类，回传给指定的 `transform` 方法执行增强，在这个 `transform` 方法中会调用 `PluginFinder` 的查找增强类的功能，找到增强类，然后通过其 `define`方法对目标类执行增强。

6. BootService

写下你的评论...

评论0

赞

聚后，封装到不同的BootService中,比如：

- ConfigurationDiscoveryService //检测配置更新
- KafkaJVMMetricsSender//向Kafka发送Metric
- KafkaTraceSegmentServiceClient//通过Kafka发送Trace
- TraceSegmentServiceClient//通过gRPC发送Trace

BootService 是在Skywalking中由 ServiceManager 通过SPI的方式进行管理，在 META-INF.services 中可以看到 BootService 全限定名的文件，文件中列出了当前Jar包中的所有的暴露出去提供服务的 BootService的实现类。

image.png

```
1 org.apache.skywalking.apm.agent.core.remote.TraceSegmentServiceClient
2 org.apache.skywalking.apm.agent.core.context.ContextManager
3 org.apache.skywalking.apm.agent.core.sampling.SamplingService
4 org.apache.skywalking.apm.agent.core.remote.GRPCChannelManager
5 org.apache.skywalking.apm.agent.core.jvm.JVMMetricsSender
6 org.apache.skywalking.apm.agent.core.jvm.JVMService
7 org.apache.skywalking.apm.agent.core.remote.ServiceManagementClient
8 org.apache.skywalking.apm.agent.core.context.ContextManagerExtendService
9 org.apache.skywalking.apm.agent.core.commands.CommandService
10 org.apache.skywalking.apm.agent.core.commands.CommandExecutorService
11 org.apache.skywalking.apm.agent.core.profile.ProfileTaskChannelService
12 org.apache.skywalking.apm.agent.core.profile.ProfileSnapshotSender
13 org.apache.skywalking.apm.agent.core.profile.ProfileTaskExecutionService
14 org.apache.skywalking.apm.agent.core.meter.MeterService
15 org.apache.skywalking.apm.agent.core.meter.MeterSender
16 org.apache.skywalking.apm.agent.core.context.status.StatusCheckService
17 org.apache.skywalking.apm.agent.core.remote.LogReportServiceClient
18 org.apache.skywalking.apm.agent.core.conf.dynamic.ConfigurationDiscoveryService
19 org.apache.skywalking.apm.agent.core.remote.EventReportServiceClient
20 org.apache.skywalking.apm.agent.core.ServiceInstanceGenerator
```

Skywalking中可以定义默认的服务并且可以定义新的服务来覆盖默认的服务，通过= BootService 上的 @DefaultImplementor 和 @OverrideImplementor 注解来实现：

1. @DefaultImplementor 注解用于标识 BootService 接口的默认实现。
2. @OverrideImplementor 注解用于覆盖默认 BootService 实现，通过其 value 字段指定要覆盖的默认实现，比如KafkaJVMMetricsSender覆盖JVMMetricsSender

```
1 @OverrideImplementor(JVMMetricsSender.class)
2 public class KafkaJVMMetricsSender extends JVMMetricsSender
```

ServiceManager 将统一初始化 BootServices 集合中的 BootService 实现，同样是在 ServiceManager#boot 方法中，会逐个调用 BootService 实现的 prepare()、startup()、onComplete() 方法

更多精彩内容，就在简书APP



"小礼物走一走，来简书关注我"

赞赏支持

还没有人赞赏，支持一下



rock_fish

总资产6 共写了9.4W字 获得82个赞 共27个粉丝

关注

什么是仓库erp系统



推荐阅读

更多精彩内容>

skywalking-agent初始化流程（一）-插件加载和插件定义

写在开篇前：github地址：<https://github.com/apache/skywalking.git>介绍...



李亚林1990 阅读 961 评论 0 赞 1

```
version: 1.0
on-title: apm-agent
on-version: 6.5.0
sheng
~Vendor: The Apache Software Foundati
~Classes: true
~Title: apm-agent
on-Vendor-Id: org.apache.skywalking
on-Vendor: The Apache Software Foundati
s: org.apache.skywalking.apm.agent.Sky
orm-Cclasses: true
kpmaven 3.6.1
~Version: 6.5
on-URL: http://maven.apache.org
```

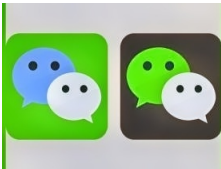
Skywalking系列博客6-手把手教你编写Skywalking插件

前置知识 在正式进入编写环节之前，建议先花一点时间了解下javaagent（这是JDK 5引入的一个玩意儿，最好了...



周立_itmuch 阅读 617 评论 0 赞 1

做一个小程序大概要多少钱



SkyWalking分布式追踪和应用性能监控系统

用SkyWalking做分布式追踪和应用性能监控系统 SkyWalking 是观察性分析平台和应用性能管理系统。提...



iamChel 阅读 379 评论 0 赞 1



写下你的评论...

评论0

赞

简书

首页

下载APP

IT技术

搜索

测试

beta

登录

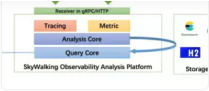
注册

后端老鸟

阅读 570

评论 0

赞 1



【长篇】插件化架构设计

一次让你彻底掌握Android插件化架构设计 插件化简介 宿主host 与 插件（免安装：不需要安装apk，下载...

flynnny

阅读 601

评论 0

赞 13

icloud-apk	DynamicAPK	Small	Droid
正则	携程	wequick	和
件Activity	只支持Activity	只支持Activity	全支
√	×	√	×
√	√	√	×
×	×	×	√
部分	大部分	大部分	几乎
一般	一般	中等	普
无	部署AAPT	Gradle插件	无

写下你的评论...

评论0

赞