在中间件中支持用户自定义连接子

# 作者介绍

## 曹东刚

北京大学电子工程与计算机科学学院副教授，主要研究方向包括软件系统、分布式技算、云计算和大数据处理。曾经参加过PKUAS、UniAS、Docklet等项目。

## 梅宏

中国科学院院士，现任北京理工大学常务副校长、北京大学高可信软件技术教育部重点实验室主任。主要研究方向包括构软件中间件、软件构件技术、开发方法学、软件复用等。承担了数十项国家级科学项目，两次担任国家973计划项目首席科学家，获国家自然科学二等奖、国家技术发明二等奖、国家科技进步二等奖、教育部自然科学一等奖、教育部技术发明一等奖等科技成果奖励。

## 曹建农

香港理工大学电子计算学院讲座教授、互联网与移动计算实验室主任。主要研究方向包括并行和分布式计算、无线网络和移动计算、大数据和云计算、普适计算、容错计算等。

# 术语介绍

## 构件

构件是具有某种功能的可重用的软件模板单元，是软件体系结构的组成元素。按照构件功能进行划分，可分为处理构件、数据构件和连接构件三种。处理构件负责对数据进行加工，数据构件是被加工的信息，连接构件把体系结构的不同部分组合连接起来。

## 连接子

连接子也叫做连接件，是软件体系结构的一部分，主要负责构件之间的交互。常见的连接子包括管道、过程调用、事件广播、BS通信协议、SQL链接等。

# 解决问题

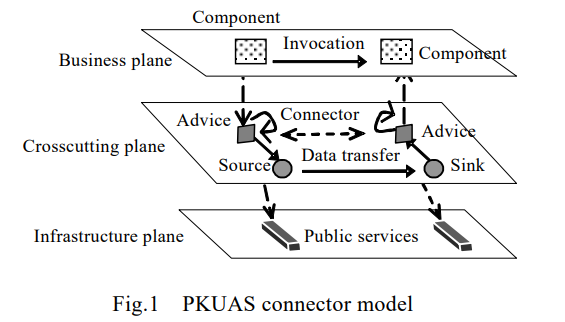
在软件体系结构研究中，连接子是与构件同等重要的一阶实体，但是研究进展并不如人意，目前连接子的研究大多局限于体系结构设计阶段的建模与验证，大多数体系结构描述语言只提供了很简单的连接子模型，连接子的行为也大多仅局限于通信。更为重要的是，在软件生命周期的另一个重要阶段——运行时刻，一直缺少对连接子的显式支持机制。

本文描述了一种在J2EE应用服务器PKUAS中显示支持用户自定义连接子的技术。该技术可以帮助用户在不必修改已有业务逻辑的前提下，方便、灵活地在应用的部署或运行时刻在连接子中通过 Advice 动态添加新的处理逻辑。同时，PKUAS 连接子的发布则借鉴了JINI 的思想，连接子在应用部署时刻由应用服务器动态生成并发布到名字服务中，由客户端在运行时刻透明地下载到本地加载。这种方法体现了良好的关注点分离原则，交互机制灵活，模块化程度高，有利于系统的维护和演化。

# 技术路线

## PKUAS中的连接子模型

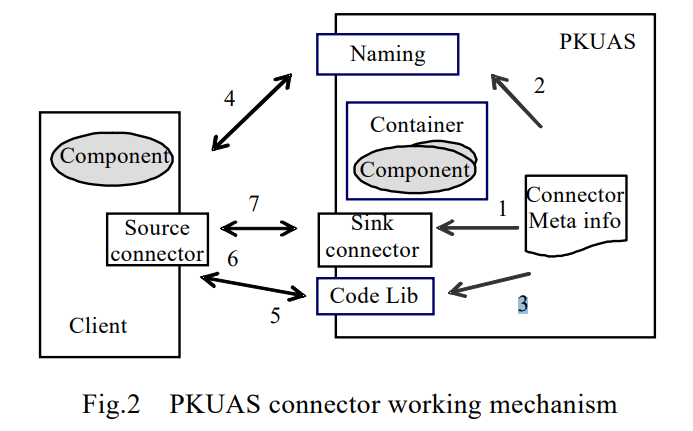
在PKUAS中，使用Advice描述构件之间的交互行为。Advice分为源端和宿端两种类型，二者协同完成一个完整的交互行为。同时，PKUAS中的连接子也分为源端和宿端两种类型。在处理构件之间的业务交互请求时，源端和宿端连接子会调用相关的Advice，动态地将Advice 与构件业务方法编排到一起。



如图，当构件之间有交互请求时，连接子就会调用相应的Advice，然后根据Advice定义的规则去调用底层的系统接口。

PKUAS连接子和Advice 为系统开发人员提供了一种灵活的、模块化的贯穿特性处理机制。用户可以在不改动构件业务代码的情况下，在应用部署时刻或者运行时刻定义新的Advice，并将其与相应的业务交互行为进行联接，对构件间的交互行为施加新的处理逻辑。

## PKUAS系统设计



在应用部署时刻，PKUAS会解析部署描述文件pkuas.xml，生成每个连接子的元信息，然后:

(1) 生成宿端连接子并和容器绑定。PKUAS 用一个容器管理一个构件类型的所有实例，这些实例共享一个连接子。容器和宿端连接子绑定之后，构件就可以接受外界的服务请求了。

(2) 生成并发布源端连接子对象。源端连接子封装了客户端的构件交互逻辑，PKUAS 首先实例化源端连接子对象并序列化，以二进制的形式发布到名字服务中，供客户下载。

(3) 发布源端连接子代码。PKUAS 连接子的构成是动态的，可能包含若干客户自定义的Advice。而客户程序可能并没有这些Advice代码，因此PKUAS需要将相关的代码(class文件)发布到代码库中，供客户程序下载。

(4) 客户程序在名字服务中查找并下载源端连接子对象。此时，源端连接子对象是以序列化的形式存在，需要实例化为内存中的对象后才可以使用。

(5) 客户程序从代码库中下载实例化源端连接子所需要的Advice代码。

(6) 将序列化的源端连接子对象实例化为内存中的对象。

(7) 连接子准备就绪，此时构件就可以进行交互了。

在应用运行期间，对于新添加Advice，PKUAS只需将其也发布至代码库，并在pkuas.xml文件中将其与相应的交互操作联系起来。当客户端执行相应交互操作时，就会自动去名字服务器中下载新的Advice。

# 验证手段

本文使用一个简单的Time应用来展示PKUAS中自定义连接子的使用方式。该应用的业务逻辑非常简单，就是获取服务器系统时间。不过该应用有一个非功能需求：时间度量。该需求需要记录time请求的传输时间以及在服务器端的执行时间，因此客户端需要获取到time请求在服务端的执行时间。

为实现该功能，用户只需要重新定义两个Advice，宿端Advice将服务器执行时间通过Result捎带给源端Advice，然后修改pkuas.xml文件即可。

# 结论

连接子作为软件体系结构设计阶段的重要概念，应该在运行阶段得到中间件的显式支撑。在PKUAS中，用户通过Advice定义各种非功能约束，这些Advice和构件的业务方法调用通过连接子组织在一起。应用系统开发者和部署者就可以在不必修改应用系统已有的业务逻辑的前提下，方便、灵活地在应用系统的部署或运行时刻通过Advice动态添加新的处理逻辑。连接子的发布方式借鉴了JINI的思想，动态发布并由客户端透明地下载，体现了良好的关注点分离原则，交互机制灵活，模块化程度高，有利于系统的维护和演化。

同时，连接子作为运行时刻的一阶实体，从软件工程的角度为开发人员提供了看待底层中间件基础设施的视点，从而在一个较高的抽象层次提供了模块化的封装贯穿特性的机制。

# 贡献

本文主要有以下贡献：

1. 分析了当今对连接子研究的不足之处
2. 详细描述了PKUAS中的连接子模型与系统的设计思路
3. 通过样例展示了PKUAS中自定义连接子的使用方式

# 借鉴之处

本文所介绍的PKUAS连接子有以下可借鉴之处：

PKUAS中的连接子与Advice机制，将比较底层且不易更改的中间件部分高度模块化，方便了开发人员对中间件模块的开发，同时也降低了开发门槛。

在PKUAS中，通过名字服务器发布连接子的交互逻辑，通过http服务发布连接子所需的Advice源码，实现了在运行时不改动已有源码即可增加新的交互逻辑的功能。这种设计模式体现了良好的关注点分离原则，交互机制灵活，模块化程度高，有利于系统的维护和演化。

# 问题

## 性能损失

PKUAS提出的连接子机制在处理交互请求时需要经过三重处理：实例化连接子、调用Advice、执行底层处理逻辑。与传统的系统相比，这种设计方式不可避免的会带来性能损失，降低系统的响应效率。

## 成本增加

PKUAS系统除了需要提供主要服务的服务器之外，还需要两个服务器用来发布连接子对象和连接子对应的Advice代码，增加了服务器成本。