**Distributed microservice architecture with Docker**

**Microservices and containers: benefits and best practices**

**Practical Use of Microservices in Moving Workloads to the Cloud**

**Achieving Continuous Delivery of Immutable Containerized Microservices with Mesos/Marathon**

**Preservation of low latency service request processing in dockerized microservice architectures**

**Open Issues in Scheduling Microservices in the Cloud**

**About microservices, containers and their underestimated impact on network performance**

**Container and microservice driven design for cloud infrastructure devops**

**Auto-scaling of micro-services using containerization**

**Swarm Based System for Management of Containerized Microservices in a Cloud Consisting of Heterogeneous Servers**

**A Decentralized System for Load Balancing of Containerized Microservices in the Cloud**

**Investigation of Impacts on Network Performance in the Advance of a Microservice Design**

**Performance Engineering for Microservices: Research Challenges and Directions**

**背景**

近年来，微服务架构[1]及容器技术备受关注，在各类文章、演讲、博客中频频亮相，成为业界最热门的话题。在时尚的词汇和热情满满的讨论背后，人们开始严肃的重新思考互联网时代服务的架构以及应用开发、运维的方法。微服务以一种全新的架构设计模式，牵动了互联网应用从设计到运维整个流程方法论的变革。而以Docker为代表的容器技术则为微服务理念提供了匹配的实现机制，进而实质性的改变了新一代应用开发和发布的方式。

**【研究内容】**

（1）云环境下基于容器的微服务组件间关联度的分析预测

目前，基于容器的微服务架构虽然摒弃了以往的单一应用的加载编译耗时长、代码管理复杂、横向扩展难、各模块之间的耦合程度高等问题。但各服务组件间的依赖是依然存在的。故本文通过分析组件容器间通信频繁程度及通信量为依据，结合部署前人为设置关联度，设计出一个组件关联度预测模型。

1. 云环境下微服务组件调度过程优化

当前，容器化加速了微服务的成长，是一种部署服务的的非常有效的方式之一。通过对组件容器的并行调度，以适应微服务环境下云计算资源调度，最终实现云系统的负载均衡和服务的性能提升。

（3）云环境下微服务部署原型系统

在实际的云应用中，云管理者需要一套完整的基于容器的微服务资源部署方案。所以要实现一个高可靠度，可扩展的云环境容器微服务调度系统。 在系统中将服务器集群分为控制节点和普通物理节点。在控制节点中添加模块，对预测和调度过程进行控制。

**、、、、、、、、、、、、、**

基于容器的微服务框架下，进行有效的，快速的，可扩展微服务组件容器的部署。将微服务组件容器进行并行调度，快速实现集群的负载均衡，同时尽可能的提高服务性能。本课题通过对微服务组件间的依赖情况进行分析，并与部署前人为设置关联度相结合，以平均通信时间最短为原则，预测出微服务组件间的关联度，在着重集群整体负载均衡的基础上进行微服务组件容器的部署，实行云资源的负载均衡和服务的性能提升。