第26卷 第6期
2011年12月

郑州轻工业学院学报(自然科学版)
JOURNAL OF ZHENGZHOU UNIVERSITY OF LIGHT INDUSTRY(Natural Science)

Vol.26 No.6
Dec. 2011

文章编号:1004-1478(2011)05-0024-04

# 普适计算中间件的自适应机制研究

郭延辉

(山东女子学院 信息技术学院, 山东 济南 250300)

**摘要**:针对目前自适应中间件平台未考虑服务质量的问题,提出了一种适合普适计算中间件的自适应机制.在自适应机制中加入QoS属性特征,引入服务合约的概念,并将遗传算法应用于优化问题,较好地实现了自适应效果.

**关键词**:普适计算中间件;自适应机制;服务合约;重配置

**中图分类号**:TP311 **文献标志码**:A

## Research on adaptive strategy for pervasive computing middleware.

GUO Yan-hui

(*School of Infor. Tech., Shandong Women's Univ., Jinan* 250300, *China*)

**Abstract**: To solve the ignorance of service quality of adaptive middleware platform, an kind of adaptive strategy suitable for pervasive computing middleware was put forword. In the strategy QoS feature is added, service contract concept is introduced and genetic algorithm is used to solve optimizoction problem. Thus better adaptive effect has been achieved.

**Key words**: pervasive computing middleware; adaptive strategy; service contracts; reconfiguration

## 0 引言

随着信息技术的不断进步,计算设备逐步朝着微型化、智能化、嵌入式方向发展.各种设备之间的通信和协作成为普适计算研究的新课题,其中解决异构化设备之间差异的普适计算中间件又成为人们研究的热点.目前,中间件平台一般通过使用后期链接和反射原理[1]来实现构件的动态配置和重配置(如OpenORB和UIC).然而,多数方法通过重配置机制只能完成应用程序的功能属性的动态组合,而忽略了应用程序中的QoS属性[2].因此,这些方法基本上不能被用于对QoS要求较高的应用程序,如视频点播、在线会议等.

UAmiddleware是基于TAO平台提出的,扩展了其在上下文感知方面的功能,并支持上下文的QoS属性特征.在该模型中,上下文感知和自适应机制共同完成自我调节功能.自适应机制是UAmiddleware的中枢,它能够根据上下文状况和用户请求进行构件服务的配置和重配置,进而完成自适应调节.本文拟基于已经建立的面向普适计算的自适应中间件模型UAmiddleware[3],采用指明配置信息、环境依赖及QoS特征的服务合约,研究普适计算中间件的自适应机制.

## 1 实现原理

本文将所有的实现都看做服务调用,并将服务

**收稿日期**:2011-04-02

**基金项目**:教育部科学研究重点项目(107106);陕西师范大学优秀科技预研项目(20070218)

**作者简介**:郭延辉(1982—),男,山东省聊城市人,山东女子学院助教,主要研究方向为嵌入式系统与中间件技术.

分为原子构件服务和复合构件服务,其中复合构件服务是由原子构件服务组合而成.整个服务调用过程由服务类型、服务合约、构件服务3部分组成.用户使用的是服务类型,服务规划器按照服务类型生成相应的服务合约,服务合约由服务配置管理器实现其与构件服务的链接,完成整个配置过程.

服务合约支持中间件平台实现构件服务部署,如图1所示.其核心包括服务配置生成器、环境管理器、资源管理器、配置管理器以及重配置管理器.自适应中间件利用资源和上下文信息[4]选择适当的服务配置来保证用户的QoS要求.服务类型和服务合约被部署在源代码旁边.装载后,生成的构件服务放置在仓库中,服务类型和服务合约被发布在代理中.

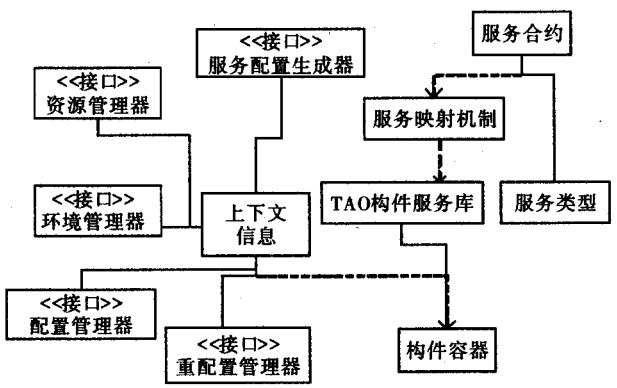


图1　自适应机制的部署

## 2　自适应机制的配置

### 2.1　自适应机制的配置过程

当应用程序用到中间件层的服务时,会将服务类型和用户的QoS请求以服务请求的形式发送到自适应中间件层,最终到达服务规划器.服务规划器根据环境上下文依赖及目前的资源上下文状态生成满足用户QoS要求的配置信息.

服务配置类是从服务合约的信息中分析可供选择的配置.服务规划器让服务配置按照请求的服务类型、服务合约类型等进行自配置.这样,一个服务配置类可以对应任何一个应用程序的配置.

如果是一个服务组合,服务配置要分析服务类型的插座和剖面端口之间的连接.服务配置将产生下一层的服务类型和服务合约,配置图如图2所示.

通过检测指定的上下文依赖,而非来自资源和上下文管理器的共享数据模型,那些目前环境不能执行的服务配置将会被过滤掉.此外,服务合约生

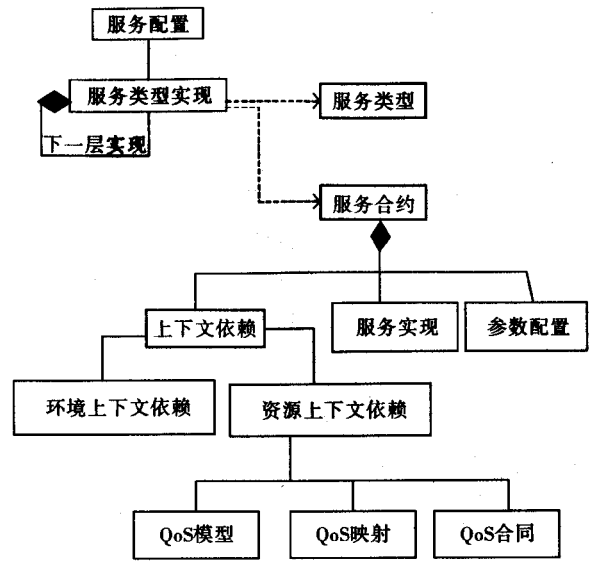


图2　配置图

成器通过预知的端到端的QoS来比较各种服务配置,所选的服务配置会被转发到配置管理器.服务配置对象和服务合约最后为配置管理器提供将要产生的构件清单和本地绑定.

中间件、服务配置及服务合约之间的交互过程如图3所示.其中,1表示请求服务;2表示分析服务可选择的实现,如服务配置;3表示返回当前环境中的资源和上下文信息;4表示过滤、比较、选择服务配置;5表示将服务配置转发给配置管理器;6表示服务配置提供将要产生的构件清单与各个端口之间的连接.

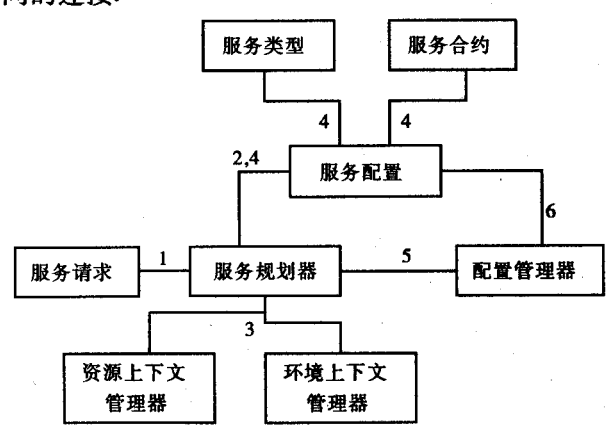


图3　自适应机制的配置

### 2.2　基于遗传算法的构件服务选择

在图3的第4步中,首先通过目前的环境上下文和资源上下文,过滤掉那些不满足条件的构件服务,再通过遗传算法解决服务组合的优化问题.

配置的过程是将功能强大的复合构件服务转化为各个原子构件服务.各个原子构件有不同的分

类,功能相同但参数不同的各个原子构件被分配到同一个包中.使用遗传算法就是使复合构件服务组合的过程中,其QoS属性能够最优化.本文只给出适应度函数的计算过程.

本文主要根据每个构件服务的非功能属性进行判断,选择出总体效果最优的服务.每个资源的目标函数$f$可表示为

$$f = \sum_{i=1}^{m} (w_i \times Q_i)$$

其中,$w_i$为相应QoS特征的权值,$0 \leqslant w_i \leqslant 1$,$\sum_{i=1}^{m} w_i = 1$;$Q_i$为子构件服务所需的第$i$个QoS属性的值;$m$为QoS属性的总个数.

不同QoS特征有不同的数量级和单位,所以必须将它们进行标准化,才能保证其在服务中的公平性.同时,有些参数会对服务产生负面影响,其值越大,性能越差;有些参数其值越大,性能越高.这就要对其进行相应的转化.

$$Q'_i = \begin{cases} \dfrac{Q_i^{\max} - Q_i}{Q_i^{\max} - Q_i^{\min}} & Q_i^{\max} - Q_i^{\min} \neq 0 \\ 1 & Q_i^{\max} - Q_i^{\min} = 0 \end{cases} \quad ①$$

$$Q'_i = \begin{cases} \dfrac{Q_i - Q_i^{\min}}{Q_i^{\max} - Q_i^{\min}} & Q_i^{\max} - Q_i^{\min} \neq 0 \\ 1 & Q_i^{\max} - Q_i^{\min} = 0 \end{cases} \quad ②$$

其中,最大值$Q^{\max}$和最小值$Q^{\min}$是资源上下文提供的最大值和最小值.在计算出构件服务的第$i$项QoS属性值$Q_i$后,利用公式①②求出其相对应的新值$Q_i'$.产生正面影响的用公式①处理,产生负面影响的用公式②处理.

最后,构件服务的适应度函数可以表示为

$$f = \sum_{i=1}^{m} (w_i \times Q_i')$$

其中,$\sum_{i=1}^{m} w_i = 1$, $0 \leqslant w_i \leqslant 1$, $0 \leqslant Q_i' \leqslant 1$.

## 3 自适应机制的重配置

自适应中间件除了具有配置功能外,还必须具有重配置功能,这才能满足自适应调解的需要[5].

实例化之后,服务配置和服务合约的功能发生了变化.它们的职责由指导服务部署转化为一个说明当前服务状态的说明文件,用来指明参数配置、环境依赖、资源状态以及用户QoS特征等信息.在重配置的过程中,服务合约就是应用程序配置在元层的一个模型.对于重配置来说,该模型非常有用,因为它不需要涉及构件本身,就可以达到检测的目的.当自适应中间件的监视器检测到环境变化时,服务规划器让服务配置重新检测上下文依赖和资源状态[6],以及用户的QoS需求,并推算变化的结果.如果上下文依赖、资源状态或用户QoS请求与配置有冲突,服务规划器将重新规划服务.如果其他的服务配置更符合用户的要求,则新配置和目前的配置一同发送到配置管理器.配置管理器通过配置中的服务合约获取目前的构件服务,并与新服务配置进行比较,以确定哪些服务被删除或被创建,以及在什么地方设置本地绑定.

图4为重配置过程中与服务合约相关的5个步骤.其中,1表示通过环境上下文和资源上下文监视器,监测环境上下文和资源上下文的变化;2表示服务规划器检查应用程序和上下文依赖与用户QoS的冲突;3表示如果有冲突,分析、过滤、比较、选择可选择的服务配置;4表示比较当前配置和新配置之间的异同;5表示如果新配置更好,为重配置确定构件服务和联系.

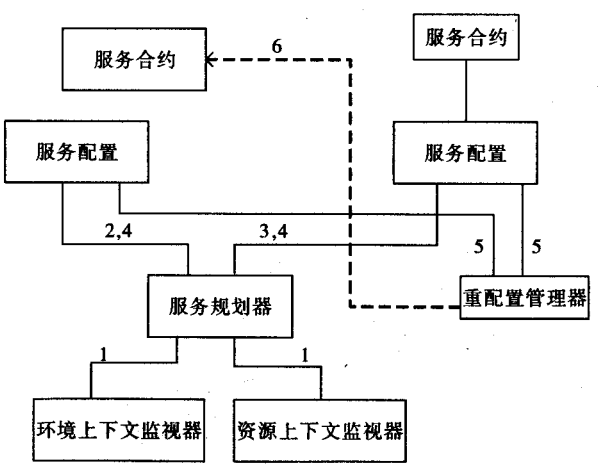


图4 自适应机制的重配置

## 4 自适应实现机制

自适应实现机制是根据上节所选的服务配置进行组合的过程,即服务组合.整个服务的实现过程可分为以下几个步骤:1)定义所需的原子构件服务;2)定义服务组合说明文件;3)指明参数配置文件;4)完成服务配置.自适应实现机制如图5所示.

在实现的过程中,将服务分为原子服务层和抽象服务层(复合构件服务).抽象服务层由原子层或

其子抽象层组成.

图5中,根据不同的上下文信息,服务类型抽象功能1可以有3种不同的服务规划(服务合约),它们分别对应着不同的实现.其中服务合约1和服务合约2直接对应着2个原子服务,而服务合约3又由2个服务类型构成,这2个服务类型之间可以进行数据的交换和传递,共同完成抽象服务1.

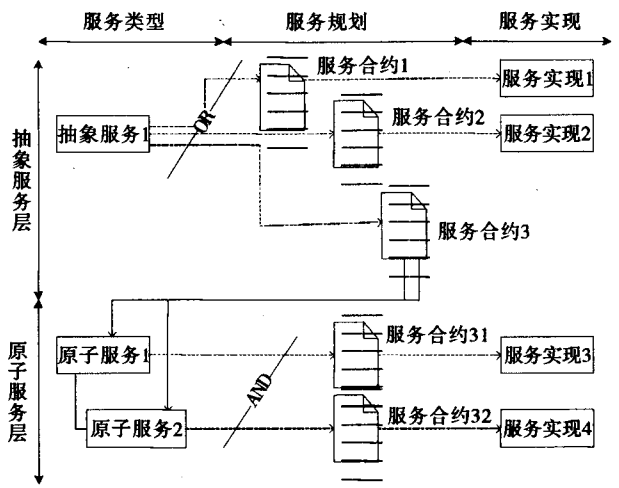


图5　自适应实现机制

## 5　结语

本文主要研究了中间件模型的自适应机制.在自适应机制中加入QoS属性特征,引入了服务合约的概念,解决了自适应机制的部署和重配置问题,并将遗传算法应用于优化组合,较好地实现了自适应效果.

**参考文献:**

[1] 黄皇,王千祥,梅宏,等.基于软件体系结构的反射式中间件研究[J].软件学报,2003,14(11):1819.

[2] Becker C, Handte M, Schiele G, et al. PCOM—a component system for pervasive computing[C]//Proc of the Second IEEE Annual Conf on Pervasive Comp and Com, Florida: Orlando, 2004:67-76.

[3] 郭延辉,李蜀瑜,梁艳华.面向普适计算的自适应中间件研究[J].贵州大学学报:自然科学版,2008,25(6):592.

[4] 郭延辉,李蜀瑜,梁艳华.自适应中间件中的上下文感知研究[J].郑州轻工业学院学报:自然科学版,2008,23(6):41.

[5] Schmidt D C. Middleware for real-time and embedded systems[J]. Comm ACM, 2002, 45(6):43.

[6] Chen H. An intelligent broker architecture for context-aware systems[D]. Baltimore: Univ of Maryland, 2003.

[7] Sheikh K, Wegdam M, Sinderen van M. Middleware support for quality of context in pervasive context-aware systems[C]//Proc of the Fifth Annual IEEE Int Conf on Pervasive Comp and Com Workshops(PerComW07), New York:[s. n.], 2007:461.

[8] Grace P, Blair G S, Samuel S. A reflective framework for discovery and interaction in heterogeneous mobile environments[J]. ACM Sigmobile Mobile Comp and Com Review, 2005, 9(1):2.