

# 支持分布式实时服务的开放系统环境\*

郭乐深\*\* 刘锦德

(电子科技大学计算机学院 成都 610054)

【摘要】 针对一个实际的开放系统环境，提出了实时活动的概念和有关的控制模型；阐述并探讨了模型中各层次的实时可调度公式。为了使实时服务在开放系统中得以具体实现，文中给出了系统的计算模型。

关键词 开放系统；实时；服务质量；实时活动；可调度；表层 QoS；资源 QoS

中图分类号 TP316

目前，计算技术正进入以网络为中心时期，开放系统解决了异质环境中的互操作问题<sup>[1]</sup>，对于基于网络环境下建立的面向种种服务(如科学计算、信息查询和远程合作等)的应用具有卓越的适用性。随着开放系统的发展，支持实时服务的开放系统的设计与建立已成为技术界关注的热点。

## 1 开放系统中间件——OSEware

OSEware(Open System Environment Middleware)是基于 ODP 国际标准开发的开放系统的中间件<sup>[2-5]</sup>。其主要功能是提供在异种机环境中的互操作处理机制，屏蔽网络环境中的异质性，为用户(如 UNIX、NT 或 VMS 等多种节点的网络的)形成一个可实用的开放系统环境。

### 1.1 交易器与服务交易<sup>[6,7]</sup>

在开放系统中服务方分布在网络环境中，中间件 OSEware 通过交易器(Trader)来组织、管理服务方，客户方通过服务交易(Trading)来定位服务方。

服务方首先在交易器中注册自身提供的服务，然后客户方向交易器提出自身的服务请求(包括服务类型、服务属性等)；交易器根据客户方提出的要求，选择、匹配在交易器中已注册的服务方，随后交易器向客户方返回被选中的服务方在交易器中所注册的有关信息，如界面引用、变量值等，通过它可与服务方建立联结，这一过程称为服务交易。

### 1.2 OSEware 的执行框架

无论是客户方与服务方都包括相应进程、代理、OSEware 系统的资源管理与通信进程，其执行框架如图1所示。该执行框架在运行时的具体功能如下：

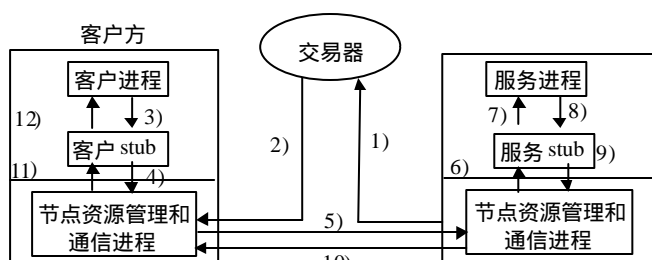


图1 执行框架

- 1) 服务方在交易器中进行服务的注册；
- 2) 客户方通过交易器进行服务的查找，以得到服务方地址；
- 3)、4) 客户发出对服务界面的操作请求，客户方代理进程装配操作参数，发送对服务界面的请求报文；
- 5) 下层资源管理和通信进程根据服务方地址，首先与服务方下层资源管理和通信进程建立联结，双方动态地为这次服务分配资源和通信信道，然后把请求包文发往服务端；

2000年3月1日收稿  
\* 电子部预研基金资助项目  
\*\* 男 30岁 博士生

6)、7) 服务端的下层接到包文，并转发给服务方代理进程，由其提取出客户方请求服务界面操作的输入参数，发给上层服务器处理；

8)、9) 服务器针对客户的请求，执行完服务界面的操作，然后把执行结果转交给服务方代理进程，后者把操作的结果装配成操作的结果报文，并交给下层资源管理与通信进程发往客户方；

10) 客户方的资源管理与通信进程收到包文，把包文转发客户方代理，同时与服务方的资源管理与通信进程一起释放为这次服务所分配的资源与信道；

11)、12) 客户方代理进程收到包文后，从报文中提取服务界面的操作结果返回给客户。

在包含异种机的网络中，各节点上都覆盖上述的 OSEware 中间件来屏蔽异质性，整个网络形成成为开放系统。

## 2 实时系统

实时系统是计算机领域一个重要研究对象。在实时系统中时间是最重要的要素，对于外部事件的响应和任务的执行都必须在限定的时间内完成，所以实时系统的结果正确性不仅在于逻辑结果，同时还取决于结果产生的时间。依据实时系统的结构、对时间严格性要求的不同，实时系统可分为：

1) 强实时系统：对时间要求最严格，如果任务的执行超过了规定的时间，会造成重大的经济损失或人员伤亡。如嵌埋在武器系统的实时系统、飞机的实时系统等，其时间的要求通常在微秒或毫秒级；

2) 中等实时系统：对时间要求较严格，如果任务的执行超过了规定的时间，输出的结果作废。如网络会议系统、VOD 系统等，其时间的要求通常在毫秒级；

3) 弱实时系统：对时间要求并不十分严格，如果任务的执行超过了规定的时间，通常会影响到输出结果的效果。如银行服务系统、飞机订票系统等，其时间的要求通常在秒或分级。

## 3 开放系统中实时服务的分析

### 3.1 实时活动及其 QoS

在开放系统环境中，实时活动(以下简称活动)在宏观上是开放系统内系统层的一次实时服务运行，在微观上是运行涉及的各节点中各类资源实时运行的总和。

QoS 是用来衡量开放系统所提供实时服务的工作质量。由于不同的系统对实时服务的要求各异，因而对其服务质量的期望值也有所不同，QoS 则作为一个参数被用来具体量化这个数值。QoS 参数可分为确定实时类或统计实时类，在火炮或航天飞机等强实时系统中应使用确定实时类，而在会议电视、VOD 等实时服务中则可使用统计实时类。

当把实时活动作为系统层中的一次实时服务运行时，其 QoS 参数应包括周期  $T$ 、开始时刻  $B$ 、执行时间  $E$  和截止时间  $D$  四个方面。为此建议将实时活动的 QoS 以一个四元组  $T, B, E, D$  来表示。成功的实时活动应要求  $T \geq D > E$ ，这时的 QoS 称为顶层或表层 QoS。当把实时活动作为运行涉及的节点内各类资源实时运行的总和时，其服务质量应计及到该活动涉及节点的资源的 QoS 的全体，这时的 QoS 称为低层或资源 QoS，所涉及的资源种类包括进程(或线程)、缓冲区、I/O 带宽和网络带宽等四个方面。对于进程(或线程)的 QoS 应包括进程的周期、开始时间、执行时间、截止时间；对于缓冲区资源 QoS 应包括这次活动所使用的缓冲区大小；对于 I/O 资源 QoS 应包括 I/O 的周期、开始时间、I/O 的数据量和截止时间；对于网络资源 QoS 应包括网络传输的周期、开始时间、截止时间和传输的数据量，上述实时活动在通常形式下的图形如图2所示。

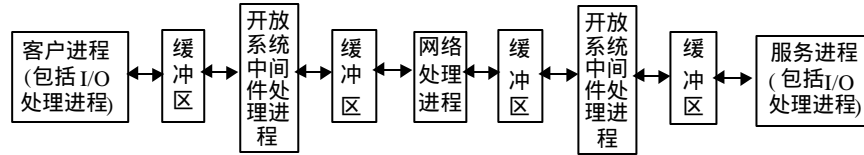


图2 实时活动形式

### 3.2 实时开放系统中层次式控制模型

为了应付在开放系统中具体实现实时服务的复杂性，我们对上述的实时活动的管理引入了分层的实时控制策略：

1) 系统层实时控制：在这一层上实时活动是指一次实时服务运行的整体，是受关注的基本单位。实时控制策略是针对开放系统环境中各个节点所产生的实时活动的集合，依据表层 QoS，通过保证各节点系统级别上实时活动的实时性来实现系统层次的实时。系统层实时控制考虑的层次高，不涉及有关节点的细节(例如不同实时操作系统的调度不同、I/O 处理的方法不同等)，且考虑的面宽，关注开放系统内负载平衡等。例如客户 A 向开放系统提出建立某个实时活动，假设存在  $n$  个服务器可以提供相同的服务，开放系统则首先检查实时活动的客户方是否满足实时要求，并且根据一定的准则从  $n$  服务器中选择一个可以保证该实时活动所要求表层 QoS 的服务器  $i(1 \leq i \leq n)$ ，则客户方可以与服务器  $i$  建立实时活动。

2) 节点层实时控制：在这一层上实时活动是某节点内各类资源按照一定的资源 QoS 执行的总和，这时资源是当前受关注的基本单位。实时控制策略为针对运行在每个节点上资源活动的集合，依据实时活动所映射到该节点内部的各资源 QoS，通过各种保证措施来达到这一节点层次上的实时。由于实时活动涉及的节点资源包括进程(或线程)、I/O、缓冲区和网络，所以这层的实时控制涉及进程实时、I/O 实时、缓冲区实时和网络实时四个方面。

控制模型的层次结构如图3所示。

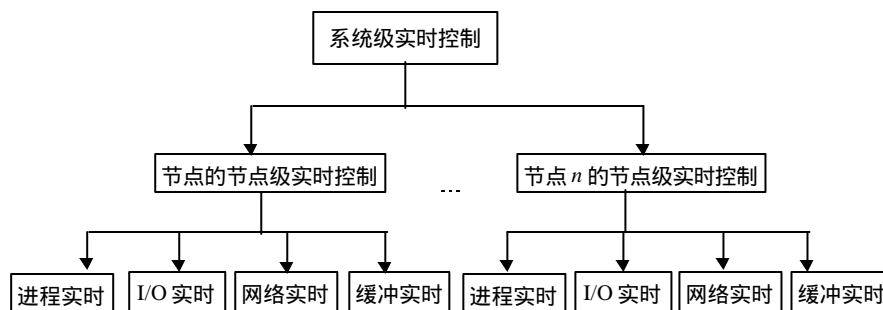


图3 实时控制模型的层次

### 3.3 系统层实时

定义 1 假定开放系统网络环境中，任一周期性实时活动  $S$  在时刻  $t_1$  被激活，到时刻  $t_2$  执行结束，且规定该活动的表层 QoS= $(T, B, E, D)$ ，则如果  $t_2 - t_1 > D$ ，则称该实时活动  $S$  发生了一次超时故障。

定义 2 在开放系统网络环境中，任意节点  $i(1 \leq i \leq m)$  中存在  $k_i$  个周期性实时活动  $S = \{A_{i1}, A_{i2}, \dots, A_{iki}\}$ ，活动  $A_{ij}(1 \leq j \leq k_i)$  的表层 QoS= $(T_{ij}, B_{ij}, E_{ij}, D_{ij})$ ，如果对于任意的活动  $A_{ij}$  的截止时间在任何时候都能得到满足，则称系统为可调度，即实现了系统层实时。

在实时开放系统中，系统层可调度性受到节点的个数、活动数量和活动对应表层 QoS 的限制。假设开放系统中存在  $m$  个节点，每个节点上的活动个数为  $k_i$ ，则系统可调度的条件为

$$\begin{pmatrix} (E_{11}/T_{11}) + (E_{12}/T_{12}) + \dots + (E_{1k_1}/T_{1k_1}) \leq 1 \\ (E_{21}/T_{21}) + (E_{22}/T_{12}) + \dots + (E_{2k_2}/T_{2k_2}) \leq 1 \\ \dots \\ (E_{m1}/T_{m1}) + (E_{m2}/T_{m2}) + \dots + (E_{mk_m}/T_{mk_m}) \leq 1 \end{pmatrix} \quad (1)$$

定理 1 假设一个服务器节点提供唯一服务  $A$  (如 VOD 服务), 其表层 QoS=( $T, B, E, D$ ), 同时设客户方节点满足实时性, 该服务器可提供服务  $A$  的实时活动数为  $w$ , 则

$$w = \left\lfloor \frac{T}{E} \right\rfloor$$

### 3.4 节点层实时

定义 3 在开放系统网络环境中的任一个节点, 如果该节点中任何资源活动的截止时间在任什么时候都能得到满足, 则称该节点是可调度的, 即实现了节点层实时。

定义 4 假设在开放系统环境中节点  $\text{Node}_i$ , 存在  $r$  个周期性实时活动  $S=\{A_1, A_2, \dots, A_r\}$ , 每一个实时活动所涉及资源总和  $\text{Resource}_i=(\text{Pro}_i, \text{Buf}_i, \text{Dev}_i, \text{Net}_i)$ ,  $i=1, 2, \dots, r$ 。

在定义 4 中,  $\text{Pro}_i$  为服务于活动  $A_i$  的一组进程的集合,  $\text{Pro}_i=\{\text{pro}_{ij} | \text{pro}_{ij}$  服务于实时活动  $A_i$ , 并且运行在节点  $\text{Node}_i, j=1, 2, \dots, p_i\}$ , 对于任何进程  $\text{pro}_{ij} \in \text{Pro}_i$ , 其 QoS 参数可用四元组定义为  $(t_{\text{pro}_{ij}}, b_{\text{pro}_{ij}}, e_{\text{pro}_{ij}}, d_{\text{pro}_{ij}})$ , 分别代表进程的周期、开始时间、执行时间、截止时间。设该节点上的进程调度时间为  $s_{\text{pro}_i}$ ;  $\text{Buf}_i$  是一组服务于  $A_i$  的缓冲区的集合,  $\text{Buf}_i=\{\text{buf}_{ij} | \text{buf}_{ij}$  服务于实时活动  $A_i, j=1, 2, \dots, q_i\}$ 。并且设该节点上的缓冲区可用值为  $\text{Buf}_{\max}$ ;  $\text{Dev}_i$  为服务于  $A_i$  的 I/O 请求。  $\text{Dev}_i$  可以用四元组定义  $(t_{\text{dev}_i}, b_{\text{dev}_i}, e_{\text{dev}_i}, d_{\text{dev}_i})$ , 分别代表 I/O 的周期、开始时间、I/O 的数据量和截止时间。同时定义该节点上 I/O 的带宽为  $\text{Dev}_{\max}$ , I/O 切换时间为  $s_{\text{dev}_i}$ ;  $\text{Net}_i$  为服务于  $A_i$  的网络进程,  $\text{Net}_i$  用四元组定义  $(t_{\text{net}_i}, b_{\text{net}_i}, d_{\text{net}_i}, c_{\text{net}_i})$ , 分别代表网络传输的周期、开始时间、截止时间和传输的数据量, 同时该节点的网络带宽为  $\text{Net}_{\max}$ 。

在实时计算机系统中, 不同资源的实时调度算法不同, 一般是基于优先级调度算法, 可分为静态算法和动态算法两类, 又可分为剥夺和非剥夺两种方式。其中静态算法包括静态表驱动切换, 单比率单调调度切换, 动态算法包括最早到期优先调度和风车算法等<sup>[8,9]</sup>。

#### 3.4.1 进程可调度

对于实时计算机系统中的进程(或线程)资源的调度的算法可以为上述的静态或动态, 剥夺和非剥夺中的任意组合。静态算法要求事先估计运行的环境, 精确地计算系统的效率和实时性的要求, 在两者之间寻找平衡点, 但由于环境的复杂性, 所以实时系统一般采用的是动态算法, 同时是剥夺方式。假设在节点中存在  $n$  个实时活动, 活动  $A_i$  包括  $l_i$  个进程, 每个进程是周期的, 节点层进程(或线程)可调度要满足以下条件

$$\sum_{i=1}^r \sum_{j=1}^{p_i} \frac{e_{\text{pro}_{ij}} + s_{\text{pro}_i}}{t_{\text{pro}_{ij}}} \leq \sum_{i=1}^n p_i (2^{\sum_{i=1}^r p_i} - 1) \quad (2)$$

#### 3.4.2 缓冲区可调度

缓冲区管理负责活动的缓冲区的分配和释放。对于实时活动, 缓冲区管理应保证活动的缓冲区需求在系统所能提供的范围之内。假设在节点中存在  $n$  个实时活动, 节点层缓冲区可调度的条件为

$$\sum_{i=1}^r \sum_{j=1}^{q_i} \text{buf}_{ij} \leq \text{Buf}_{\max} \quad (3)$$

### 3.4.3 I/O 可调度

I/O 调度不同于进程调度, 不能采用可剥夺方式。这是因为 I/O 调度不能随意进行切换, 例如对于磁盘调度, 磁头刚寻道定位结束, 若进行 I/O 的切换会浪费时间, 必须在 I/O 读完数据后才能切换, 所以 I/O 调度应采用基于优先级的非剥夺方式, 可以是静态或动态算法。假设在节点中存在  $n$  个活动, I/O 带宽为  $Dev_{max}$ , 在单位时间内 I/O 切换的时间为  $n*s_{dev}$ , 那么用于数据传输的时间为  $1-n*s_{dev}$ , 节点层 I/O 可调度的条件为

$$\sum_{i=1}^r \frac{e_{dev_i} + s_{dev_i} \times Dev_{max}}{t_{dev_i}} \leq Dev_{max} \quad (4)$$

### 3.4.4 网络可调度

在具体节点中, 活动的可调度性受到网络带宽的限制。假设节点中存在  $n$  个活动, 网络数据传输速度的带宽为  $Net_{max}$ , 则节点层网络可调度的条件为

$$\sum_{i=1}^r \frac{c_{net_i}}{t_{net_i}} \leq Net_{max} \quad (5)$$

显然, 节点层可调度则必须同时满足式(2)~(5)。

## 4 实时开放系统计算模型

由以上分析可知, 系统层实时是从宏观的角度考虑开放系统的实时特性, 开放系统根据实时活动表层 QoS 来保证系统实时性; 节点层实时从微观的角度来考虑具体节点中服务于实时活动的各种资源的可调度性。显然不同的实时计算机系统(如 RTOS、VxWorks、LynxOS、Windows NT 等)采用各自不同的方法来保证内部的实时性。为了屏蔽系统之间的差异, 实现计算的透明性, 开放系统的计算模型必须提供统一的机制来定义实时活动的表层 QoS 与资源 QoS, 而且在程序语言上要实现这一高级支持机制, 籍此用户能方便地开发具有实时服务的应用。为此, 对于支持实时服务的开放系统要求如下:

1) 为了实现系统层可调度, 开放系统应支持实时活动的表层 QoS 的注册, 该函数的入口参数包括服务的优先级、表层 QoS, 即( $T, B, E, D$ );

2) 为了实现节点层可调度, 开放系统提供支持统一的实时活动的进程(或线程)资源 QoS 注册的服务, 该界面入口参数为进程(或线程)的优先级、进程(或线程)的周期、开始时间、执行时间和截止时间;

3) 开放系统应提供统一的实时活动的缓冲区资源 QoS 注册的服务, 该界面入口参数为 Buffer 的大小;

4) 开放系统应提供支持统一的实时活动的 I/O 资源 QoS 注册的服务, 该界面入口参数为 I/O 的优先级、周期、开始时间、I/O 的数据量和截止时间;

5) 开放系统应提供统一的实时活动的网络资源 QoS 注册的服务, 该界面入口参数为网络的优先级、网络传输的周期、开始时间、截止时间和传输的数据量;

6) 应提供实时活动建立函数 bind, 客户方通过它完成具体实时活动, 其功能为:

(1) 建立活动时进行系统层实时的核查: 根据步骤1)注册的该活动的表层 QoS, 在整个系统范围内找出可以提供该服务且满足式(1)的服务器节点;

(2) 建立活动时进行节点层实时的核查: 根据步骤2)~5)所注册的活动使用的各个资源 QoS, 为客户方与服务方进行节点层实时核查, 看其是否满足式(2)~(5)。

以上两步成功后, 为客户方节点和服务方节点分配相应的资源, 函数成功返回。

## 5 结束语

对于开放系统中实时服务的要求是防止服务的超时故障。本文提出了基于实时活动的概念和两层实时控制的模型，系统层控制根据系统内各节点上活动的表层 QoS 来保证系统实时性，节点层控制根据节点内部活动的资源 QoS 来保证本节点所有活动的实时性，文中还给出了控制各层实时性所需的可调度公式，并从计算的角度阐明了在开放系统中如何支持实时服务。通过建立支持实时应用的开放系统，将使开放式分布处理(ODP)技术进一步的推广和应用。

### 参 考 文 献

- 1 刘锦德. 对于开放系统内涵的澄清. 计算机应用, 1997, 17(6): 1~3
- 2 ITU/ISO. Reference model of open distributed processing-Part 1: overview. ISO/IEC 10746-1, ITU-T Rec X901, 1996
- 3 ITU/ISO. Reference model of open distributed processing-Part 2: foundation. ISO/IEC 10746-2, ITU-T Rec X902,1995
- 4 ITU/ISO. Reference model of open distributed processing-Part 3: architecture. ISO/IEC 10746-3, ITU-T Rec X903,1995
- 5 ITU/ISO. ODP trading function – Part 1:specification. ISO/IEC IS 13235-1, ITU/T Draft Rec X950, 1997
- 6 苏 森, 唐雪飞, 刘锦德. ODP 中的服务交易技术研究. 电子科技大学学报, 1998, 27(6): 638~641
- 7 苏 森, 唐雪飞, 刘锦德. 面向对象的互操作技术. 电子科技大学学报, 1998, 27(1): 90~94
- 8 Stankovic J A, Spuri M, Natale M D, *et al.* Implication of classical scheduling results for real-time system. IEEE Computer, 1995, 28(6): 16~25
- 9 Lehoczky John P, Lui Sha, Ye Ding. The rate monotonic scheduling algorithm:exact characterization and average case behavior. IEEE Computer, 1989: 166~171

## Open System Environment for Distributed Real-time Applications

Guo Leshen    Liu Jinde

(College of Computer Science and Eng., UEST of China Chengdu 610054)

**Abstract** For an existing open system environment, the concept of real-time action and a two-layer management model for real-time are proposed in this paper. Related schedulable strategies of real-time actions for the two layers are discussed in detail. For the sake of making such kind of support into reality, a computer model is presented.

**Key words** open system; real-time, real-time action; schedulable; appearance QoS; resource QoS