基于CORBA的分布式系统自适应容错模型的研究*)

A CORBA-Based Model for Adaptive Fault Tolerance in Distributed Systems

李琪林　陈　宇　周明天

(电子科技大学计算机科学与工程学院　成都610054)

Abstract Distributed computer systems are widely applied to safety-critical systems, such as air traffic control systems, online paying systems and stock exchange systems. Fault tolerance is a main way of assurance of system reliability. However, distributed systems must always operate in highly dynamic environments, so fault tolerance mechanism should provide more intelligence to adapt itself to in response to the changes in system resource, application demands and user requirements. CORBA is a distributed object computing middleware standard. With support of its core, Object Request Broker (ORB), CORBA provides open and standard communication infrastructure for development and deployment of distributed applications. This paper presents a CORBA-based model for adaptive fault tolerance in distributed systems and describes its structure and implementation schemes on top of a CORBA-compliant object middleware, TongBroker. Finally we give our conclusion.

Keywords Distributed systems, Software fault tolerance, Adaptive fault-tolerance, Middleware, CORBA

一、引言

分布式系统需要可靠性保证，例如在线支付系统对安全性提出了很高的要求。因此，分布式系统必须提供可靠性机制，支持关键业务。容错技术是分布式系统运行过程中可靠性保证的重要手段，冗余资源是实现容错的根本保证。单一的容错策略仅适用于特定的应用和特定的系统，无法适应系统状态的动态变化，支持广泛的分布式应用。系统的容错模型应该能够智能地根据外部运行环境的变化，选择合适的容错策略，以便在保证系统可靠性的前提下提高系统资源利用率[1]。

自适应容错可以在分布式系统中的各个层次中实现。基于操作系统的自适应容错强烈地依赖于特定的操作系统，系统可移植性差；而在应用程序中实现自适应容错又加重了开发者的负担。CORBA是分布对象计算的中间件标准，为分布应用的开发和布署提供标准的服务和协议。基于CORBA实现的自适应容错模型可以充分利用CORBA提供标准服务和协议，简化应用对象间的通信；同时，CORBA先天的平台无关性方便系统的移植；因此能否基于CORBA实现自适应容错，保证系统可靠性的基础上提高系统资源利用率具有重要的理论价值和实际意义。

本文对自适应容错技术作简要叙述，简要介绍了我们自行研制的遵循CORBA规范的对象中间件平台TongBroker的主要特点，详细描述了在Windows NT上，基于TongBroker实现的自适应容错模型的结构，及其组成部分的实现策略，最后给出了结论。

二、自适应容错

容错是指如何保证在出现错误时系统仍能提供正确服务[2]。实现容错的基本手段是冗余技术，目前采用的主要技术包括P-E(Primary-Exception handling)、RB(Recovery Block)、DRB(Distributed Recovery Block)、NVP(N-version Programming)和NSCP(N-Self-Checking Programming)[3]。

分布式系统具有规模大，复杂度高的特点，例如，空中交通管制系统。显然，空中交通管制系统的负载随着空中飞行物体数量的变化而变化。在出现硬件或软件的暂时或永久故障的情况下，为了保证关键任务仍能正常工作，输出正确结果，容错是必不可少的。通常，为保证系统的容错要求，系统的设计往往以系统可能出现的最坏情况为基础。然而，最坏情况在系统的生命周期中所占的比例很小，因此为这种情况而预留的冗余资源在大多数时候被浪费了。此外，大量冗余资源的引入会增加成本和系统复杂度，反而导致系统可靠性的降低。但是，若系统的冗余资源数过少，任务又可能因为无法及时获得足够的冗余资源而无法达到指定容错级别或直接被放弃。另一方面，不同的容错策略在不同的运行环境中，其效果也是不同的。例如，在冗余资源丰富的情况下，任务的容错可以采用多个模块并行的方式实现，如NVP，DRB和NSCP。若系统负载较高，则可采用模块串行运行实现容错，如RB和P-E等。综上所述，在系统运行的不同状态下采用同一种容错策略是不合理的。我们认为，在分布式系统中应该有多种容错策略并存，根据运行环境的变化，为应用选择合适的容错策略，在保证系统可靠性要求的前提下，灵活地配置系统资源，提高资源的利用率和系统的任务吞吐量。这就是所谓的自适应容错。

自适应容错的主要目的是提高分布式系统的可靠性、性能和在恶劣环境下可生存能力。因此，自适应容错应该具有以下几个主要特征：

高效　自适应容错机制应该能够提高系统的可靠性，同时其运行开销必须保持在可以接受的范围内。

高的资源利用率　自适应容错机制应该通过提高资源的利用率降低系统对冗余资源的需求，并能够保证在部分资源暂时无法使用的情况下，满足关键任务的可靠性要求。

通用性　自适应容错机制必须能够应付不同类型的系统错误，如外部设备错误、处理器错误、系统软件和应用软件错误，同时也应该能够处理暂时错误和永久错误。

可配置　自适应容错机制必须能够接受通过参数形式传

*)本文获四川省重点科技计划项目基金资助。李琪林　博士研究生，主要研究方向：分布式系统，对象中间件技术。陈　宇　博士研究生，主要研究方向：实时操作系统，软件容错技术。周明天　教授，博士生导师，主要研究方向：网络，对象中间件技术，应用服务器。

递的应用程序对于容错的特殊要求，并基于这些参数进行容错策略的选择。

多样性 自适应容错机制应该包含多种不同的容错策略，以响应运行环境的变化。

界面友好 自适应容错机制作为分布式系统的一个模块，应该具有简单、清晰的编程接口，以减轻应用程序设计者的负担。

目前，国外多家研究机构和实验室已就自适应容错展开相关的研究并且在确定自适应容错的概念框架和自适应容错的潜在应用方面取得一定的成果[4~6]。在二十世纪九十年代的前半期，对于自适应容错的研究主要围绕其基本概念和实现机制进行。Kim 和 Lawrence 对自适应容错在复杂的实时分布式应用进行了分析，并提出了 action-level 容错的概念[7]，Bondavalli 等在分布式实时操作系统 Spring 上提出了自适应容错模型 FERT[8]。基于以上对自适应容错原理和实现机制的研究，建立了一些具有自适应容错功能的实验系统，并逐步走向实用，其中包括 Electra[9]、Proteus[10]、ROAFS[11] 和 Chameleon[12]等。但是，分布式自适应容错的有效实现模型仍有待于进一步研究。而且，绝大多数这类系统的原型实现均基于非标准的软硬件平台，系统的可移植性差。因此很有必要基于开放标准的平台研究自适应容错的模型和实现策略。为此，本文提出了基于 CORBA 的分布式系统自适应容错模型及其实现策略，并在对象中间件平台 TongBroker 上予以实现。

三、TongBroker

TongBroker 是我们自行研制的新一代面向对象的中间件，符合分布对象标准规范 CORBA2.x。TongBroker 系统主要组成部分包括：ORB 核心、IDL 编译器、扩展设施、CORBA 服务和系统管理。ORB 核心即对象请求代理，它将底层不同类型的通信设施进行集成，提供了分布对象计算的基础设施。IDL 编译器目前支持 IDL 语言到 C++ 的映射。利用编译器，可以生成 ORB 中的客户存根和服务器骨架代码，与应用代码最后构成完整的应用。扩展设施是一些方便应用或增强系统性能的工具如构件应用服务器、应用程序管理器等。CORBA 服务包括名字服务、事件服务和对象交易服务。分别用于对象定位、支持事件驱动模式和保证交易完整性。系统管理提供对各部分进行控制管理的工具包括集中管理控制台、管理代理和 SNMP 代理等。

TongBroker 具有分层的体系结构，核心支持构件化的协议模块。核心最下面是通信层，提供 GIOP 消息的通信能力，可以配置选用不同的协议或增加新的协议构件。GIOP 消息层负责进行 GIOP 消息的语法和语义解释，通过动态绑定的接口使用通信层提供的服务。在服务方，服务对象登记在对象适配器中。用户根据业务定义 IDL 接口，进而可以编写客户和服务对象实现，存根和骨架由 IDL 编译器生成。流作为核心模块在分布对象交互的全过程使用。

TongBroker 允许用户方便灵活地开发分布对象应用；提供了从对象、线程管理到负载均衡等多种性能管理手段；同时具有良好的可伸缩性，支持关键任务处理；另外，作为应用集成工具，TongBroker 可以将不同语言、平台、产品和不同体系的各种应用系统无缝地集成在一起。因此，基于 TongBroker 实现分布式自适应容错模型，可以充分利用 TongBroker 提供的标准服务和协议，简化系统设计和实现的复杂性且系统具有良好的可移植性。

四、自适应容错模型

分布式系统自适应容错模型包括自适应管理器、容错策略管理器和系统监控管理器。自适应容错模型利用底层操作系统和中间件提供的服务，进行资源分配，支持应用间通信，监控环境、用户和应用的变化，根据用户和应用的要求，提供相应的容错机制，实现应用的自适应容错。其结构如图1所示。

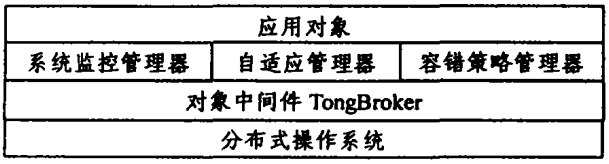

图1 基于 CORBA 的分布式系统自适应容错模型

4.1 基本容错策略分析

理论上，自适应容错可以包容任何容错策略。但是，容错策略实际上可以分为并行、串行和简单错误处理三类。并行容错指同时在多个结点上运行同一任务的多个不同版本，并通过比较决定最终的输出，如 DRB 和 NVP。这一策略可以在较短时间内获得运算结果，但要求同时占有多个结点且结点的协调工作十分复杂，开销较大。串行容错指任务的不同版本在一个结点上串行执行，仅当前一个版本无法通过测试才运行下一个版本，如 RB。在出错情况下，任务的最终响应较并行容错迟缓，但正常情况下，两者几乎一致。同时，运行于单一结点使串行操作具有简单、开销小的特点。简单错误处理通常指异常处理，当任务运行出错时，转而运行简单的错误处理小程序，如记录错误状态等，将系统恢复到任务执行前的正确状态，然后直接返回到父任务，由父任务决定随后的操作。简单错误处理不能获得任务的正确输出，但能够保证系统始终处于正常状态。因此，自适应容错从上述三类容错策略中分别选出一种，作为基本容错策略，其中包括 P-E、RB 和 DRB，由自适应决策机制根据当前系统状态选择合适的容错策略。之所以在多种并行容错策略中选择 DRB 是因为：DRB 是 RB 的扩展，是在 RB 的基础上实现的，选择 DRB 有利于代码的重用。

4.2 相关数据库

为了支持自适应容错，基于 CORBA 的分布式系统自适应容错模型需要维护以下三个数据库：

1）环境数据库：记录分布式环境中远地节点的状态信息。系统监控管理器负责监视环境变化，维护和更新数据库。

2）内部状态数据库：记录本地节点的资源状况。系统监控管理器负责监视环境变化，维护和更新数据库。

3）用户需求数据库：记录用户为应用定义的特殊属性，如应用的重要性。

自适应容错管理器依据用户需求数据库提供的应用属性信息，根据环境和内部状态数据库的系统资源信息，确定应用的容错结构。

4.3 自适应管理器

自适应管理器是自适应容错模型的核心，它根据环境和用户需求变化，为应用选择最合适的容错结构。自适应管理器做出的自适应决策转发给容错策略管理器，由容错策略管理器为应用分配资源以适应满足应用需求。

为帮助自适应管理器确定应用的容错策略，需要为应用增加新的属性——重要性，并根据该属性将应用分为三类：关键应用、重要应用和普通应用。关键应用需要采用 DRB 或 RB

容错结构，重要应用需要采用 RB 或 P-E 容错结构，而普通任务仅要求 P-E 容错结构或无需容错服务。当新的应用请求到达后，自适应管理器根据应用的重要性、环境数据库和内部状态数据库的系统资源信息，确定应用的容错结构，并将决策结果传递给容错策略管理器。容错策略管理器依据容错决策，为应用构建相应的容错结构。

自适应管理器应具有以下特点：1)为关键应用维持最小的可靠性和性能要求；2)及时反映环境、系统和用户需求的变化；3)自适应决策反映特定应用的自适应参数。基于上述的分析，我们将自适应管理器作为 ORB 的服务实现。它根据环境数据库、内部状态数据库和用户需求数据库，依据系统状态的变化，为应用做出自适应决策。

4.4 系统监控管理器

系统监控管理器维护系统的当前状态信息。它主要负责监控分布式系统中各节点的状态。当远地节点无法正常工作时，系统监控管理器将该信息记录在环境数据库中，供自适应管理器和容错策略管理器使用。当节点从失败中恢复或新的节点加入时，系统监控管理器会更新环境数据库，便于正常节点的快速学习。另外，系统监控管理器从底层的资源管理模块中获取本节点的资源信息，该信息记录在内部状态数据库中，供自适应管理器和容错策略管理器使用。

我们将系统监控管理器作为 ORB 的服务实现。利用 TongBroker 提供的网络管理功能，系统监控管理器定期生成检测信号，检测邻居节点的状态及节点间连接的状态，从邻居节点收集错误怀疑报告，使用收集的信息判断错误源并向系统中所有正常的工作节点发送出错通知。另外，系统监控管理器还定期维护环境数据库和内部状态数据库，为自适应管理器提供决策支持。

4.5 容错策略管理器

容错策略管理器根据自适应管理器的自适应决策，提供不同的容错方案，支持广泛的应用。容错策略管理器主要提供用户透明的自适应容错服务。目前，容错策略管理器支持三类容错方案 P-E、RB 和 DRB。

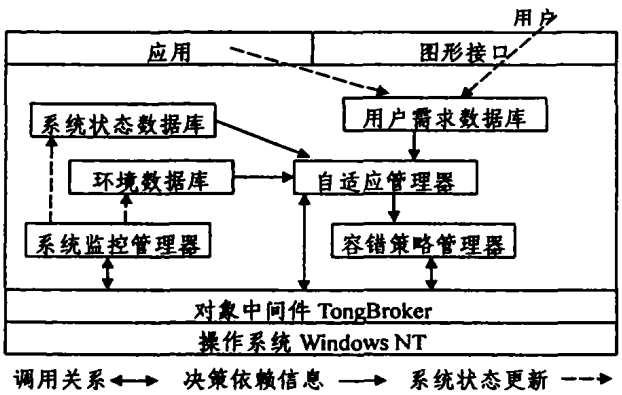

图2 自适应容错模型内部结构的交互关系

容错策略管理器设计的基本思想是容错功能与应用分离，即容错功能对应用透明。容错策略管理器从自适应管理器获得应用的容错服务决策信息。对于关键应用，若本地节点无法获得足够的资源时，容错策略管理器通过 TongBroker 的进程管理功能在就绪队列中选择尚未被调度过的部分普通应用放入资源等待队列，释放这些应用的资源供关键应用使用。若通过释放普通应用占用的资源仍无法满足关键应用的需要，或关键应用选择并行容错策略，容错策略管理器将根据环境数据库中的节点状态信息，选择关键应用的可执行节点。对于重要应用，当在本地节点无法获得足够的资源时，则将其放入决策等待队列。对于普通应用，当在本地节点无法获得足够的资源时，则放弃并返回相应信息。

容错策略管理器作为 ORB 的服务实现，利用 TongBroker 提供的底层通信设施和自适应管理器交互，根据自适应管理器的自适应决策，为应用构建不同的容错服务。基于 CORBA 的分布式系统自适应容错模型各部分的交互如图2所示。

结论 近年来，分布式系统在安全领域内得到越来越多的应用，比如空中交通管制系统等。可靠性已经成为衡量系统优劣的一个十分重要的因素。在系统运行过程中，为提高系统的可靠性，基于资源冗余的容错技术被广泛的应用。传统的分布式系统在应用程序设计时，将功能与容错作为一个整体实现。因而，采取何种容错策略，在系统构成之初就已经确定，并将在该应用的整个生命周期中保持不变。然而，随着大型、高复杂度系统的出现，仅依靠某一种固定容错策略无法适应多变的运行环境和系统负载。自适应容错成为解决这一问题的重要手段。

自适应容错将几种适用于不同运行环境和系统负载情况的容错策略结合起来，根据应用本身的属性和资源状况，动态地为任务选择当前状态下最佳的容错策略，在保证系统可靠性要求的前提下，灵活地配置系统资源，提高资源的利用率和系统的吞吐量。基于开放标准的 CORBA 平台实现的分布式系统自适应容错能充分利用 CORBA 提供的标准服务和平台无关性，简化分布对象的通信和互操作，具有良好的可移植性。我们在跟踪国外技术发展的基础上，提出了基于 CORBA 的分布式系统自适应容错模型和其组成部分的实现策略，并在 Windows NT 和我们自行研制的遵循 CORBA 规范的对象中间件平台 TongBroker 上实现了该系统。实践证明，基于 CORBA 的自适应容错在保证系统可靠性的基础上提高系统资源利用率是可行的，能够推动高可靠的分布式系统的进一步发展。

参考文献

1 Kim K H, Lawrence T. Adaptive Fault-Tolerance in Complex Real-Time Distributed Application. Computer Communication, 1992, 15(4): 243～251
2 Cristian H. Understanding fault-tolerant distributed systems. Communications of the ACM, 1991, 34(2): 56～78
3 Lyu M R. Software Fault Tolerance, Wiley, 1995
4 Gupta, et al. Adaptive Fault Resistant System(AFRS). Rome Laboratory USAF: [Technical Report RL-TR-95-3]
5 Goldberg J. Adaptive Fault Resistant System. SRI International: [Technical Report SRI-CSL-95-02]
6 Bihari B, Schman K. Dynamic Adaptation of Real-Time Software. ACM Transactions on Computer Systems, 1991, 9(May): 143～174
7 Kim K. Action-level fault tolerance, in Advances in Real-Time Systems, S. H. Sang ed., Prentice Hall, 1994. 415～434
8 Bondavalli A, Stankovic J, Strigini L. Adaptive Fault Tolerance for Real-Time Systems, in Responsive Computer Systems: Steps toward Fault-Tolerant Real-Time System, D. S. Fussell and M. Malek ed., Kluwer, 1995. 187～208
9 Landis S, Maffeis S. Building Reliable Distributed Systems With CORBA, in Theory and Practice of Object Systems, Wiley, 1997
10 Sabnis C, Cukier M, et al. Proteus: A Flexible Infrastructure to Implement Adaptive Fault-Tolerance in AquA. In: Proc. of the 7th IFIP IWC in DCCA, 1999. 137～156
11 Shokri E, Crane P. Architecture of ROAFTS/Solaris: A Solaris-Based Middleware for Real-time OO Adaptive Fault Tolerance Support, Proc. COMPSAC98, 1998. 90～98
12 Shokri E, Hecht H, et al. An Approach for Adaptive Fault Tolerance in OO Open Distributed Systems. International Journal of Software Engineering and Knowledge Engineering, 1998, 8(3)
13 Kalbarczyk Z, Iyer R K. et al. Chameleon: a Software infrastructure for Adaptive Fault Tolerance. IEEE Transactions on Parallel and Distributed Systems, 1999, 10(6): 560～579