计算机科学1991№.2

分布式软件的可靠性及容错

鄢 勇　金灿明　(华中理工大学)

摘　要

本文从可靠性的角度出发，着重分析了分布式系统中各层次软件可靠性所处理的一些问题及其相应的解决方法，并指出了一些有待进一步研究的问题。

随着分布式应用的不断发展，人们对分布式系统的研究与探索更为深化，解决分布式系统中各种问题的方法也层出不穷，而且对分布式系统的研究逐渐由面向机器转向了面向应用层。人们开始注意到如何为用户提供一个透明、一致的用户视图，如何有效地利用资源为用户提供高可靠性的、可扩展的、具有良好功能的结构与服务。

分布式系统由于其物理处理机地理上的分散导致了服务与控制的分散，且通迅时延的任意性使全局现行状态视图成为不可知的，这样人们只能用局部消息，通过分散控制而完成全局效应。分布式系统状态可分为如下三类：

·可知状态：对于两结点A，B和A上的状态S，Δt为S状态的最小改变时间间隔，若从B发消息到A再返回B的信息时延$t' < \Delta t$，则说S对B可知。

·可见状态：局部状态对于同结点上的对象都是可见状态。

·不可知状态：不属于以上两种状态的状态称为不可知状态。

正因为存在这种不可知状态，使得高可靠性的分散控制难于实现。分布式系统的另一重要特点是要求可扩展性，它是一种在不对系统作重大改变的基础上适应长期改动和短期改动的能力。其中短期改动指变化的工作负载、子网流通和主机或子网出错及其它；而长期改动指系统要求说明作较大的改动。

系统透明性和可扩展性都要求分布式软件具有较好的可靠性，使分布式系统能完成系统规格说明所要求的功能。这种可靠性是一种多维活动，它所涉及的层次有：进程结构化、资源结构化、进程通迅、通迅通道的控制及分散控制等，因此是系统功能的决定因素之一。在本文，我们着重探讨一下在分布式系统各个层次的可靠性及其一些解决方法。

一、可靠性

1. 一些重要的概念

分布式系统中的事件可分成如下两类：

(1)期待事件：系统设计要求要完成的功能。

(2)不期待事件：设计要求说明不期待发生的事件，例如出错事件。该类事件根据其预测性又分成可预测事件和不可预测事件

定义1　可靠性是预防和容纳不期待事件所达到的程度。

定义2　完全可靠：在假定部件不会产生永久性故障的情况下，对于部件R，其功能规格说明为ρ，对满足ρ(A)的任一操作A，若：1)要么R完成ρ(A)，2)要么对系统不产生任何影响，则说部件R完全可靠，其中：ρ(A)指A是ρ所容许的操作，R既可指整个系统，也可指系统的某个层次中的分量。

改进可靠性的方法主要有以下两种：

(1)错误防止：利用使软件正确的方法来防止错误。

(2)容错：利用错误探测及冗余信息方法去处理不期待事件。

在分布式系统中，可靠性的改善主要靠数据、控制冗余和系统在地理上的分布及其系统具有相互窥视主机和通迅处理机的状态的能力来达到。一个部件A的可靠性大小可

随部件冗余个数增多而增大，但整个系统的可靠性却不与其冗余个数成正比。在分布式系统中，各个部件的可靠性之间是相互制约的。

2．可靠性是一种多维活动

系统可靠性和容错的研究最初起源于硬件领域，硬件可靠性只是实现一个可靠系统的基础，而软件可靠性则直接为系统可靠性的实现提供了保障。由于软件可靠性与硬件可靠性的处理对象具有一定的共性，因此这两个领域所用的可靠性技术彼此相关。再者，由于软件涉及一个更为广泛的应用领域，所以它处理的问题比硬件系统要处理的问题远为复杂，且更难于解决。与可靠性密切相关的问题有以下几类。

(1)错误封闭 这是一种限制错误蔓延到系统某一区域或某一过程的方法。它主要涉及系统的结构问题。最好的方法是采用好的模块化方法。而在分布式系统中最为常见的是具有资格证的对象模型方法。另外，系统一致性的检查也能加强错误封闭。

(2)错误探测 指识别错误存在的动作，最常用的技术有一致性和合理性检查、监视时钟、数据和功能复用及逆向检查(reversal checks)。错误探测机制的设计依赖于潜伏期的要求(即从出错到发现为止的时间间隔)。

(3)错误掩蔽 错误掩蔽技术的目的在于隐藏出错所造成的影响，因为在有的情况下错误可被纠正或忽略。这类技术常用的方法是错误纠正码、复用和多数选择。

(4)重试 由于有的错误是暂时的，且对系统不造成实质性损坏，对这类错误往往只须将以前的动作重复一遍即可消除。

(5)诊断 一些现有的错误探测方法往往只能描述一个错误出现，而对出现错误的特性不给出描述。诊断方法则是一种较为全面的错误探测方法，它不仅确定出现的错误的类型，而且确定错误所在地点。这种错误探测机制是与应用相关的，通用机制往往难于实现。

(6)重构 由于硬／软部件的出错往往会导致系统结构的改变，如分布式系统中物理处理机的出错会导致网络拓朴的改变。这就要求系统具有一种能动态重新组织其系统部件的能力。

(7)恢复 它是消除出错事件所造成影响的一种途径。常用的方法有do and undo协议、具有回撤的检测点方法(chechpoints with rollback)和向前错误恢复方法。恢复技术与错误潜伏期和错误封闭有着密切的关系，一般说来，探测错误所花时间愈长，要恢复的内容就愈多，而错误封闭尽可能阻止错误蔓迟。在现有的容错操作系统中，错误封闭与恢复往往是核心方法之一，另外，事务概念将恢复与并发控制紧密地联系在一起。

(8)重启动 重启动指在出错时重新执行所有的或一部分正在执行的运算，以便消除错误。重启动依据对象的性质作相应处理：对幂等运算只要重新执行一遍即可，而对非幂等运算则要利用重启动日志才行，往往在复杂的系统中，重启动与恢复是并行进行的。

(9)修复 修复方法是采用某种技术纠正出错的原因，修复方法与修复对象有密切联系。

(10)重组合 将修复的部件或被置换掉的部件重新放入系统。这种组合可动态完成，也可在中断系统服务的情况下静态完成，这取决于系统的要求，而在分布式系统中动态完成常常是所需的。

二、分布式软件的可靠性

在本节，我们将从基本物理系统、通讯子网、分布式操作系统和分布式应用这四个方面考虑分布式软件的可靠性。

1．基本的物理系统

一个可靠的分布式系统首先必须满足以

下二个条件：

(1)实现可靠的存贮器

根据完全可靠的定义可知，一个完全可靠的存贮器具有以下特征：

(a)存贮器上现有的内容不会因不期待事件的发生而丢失。

(b)存贮器上的操作完全可靠。

对于完全可靠存贮器的实现模型，Lampson提出了一个较为可行的稳态存贮器[4]方法。这种稳态存贮器中实现(a)的方法是利用二个错误不相关的存贮单元共同模拟一个存贮单元；而实现(b)的方法则是利用逐层排错的过滤法提供一个完全可靠的运算，但稳态存贮器没有处理不可预测事件。这种处理只能利用语义知识来进行，具体可行的策略有待进一步研究。

(2)实现完全可靠的处理机

对于提交给处理机的任务A，设系统规格说明为ρ，一个完全可靠的处理机具备以下条件：(a)当$\rho(A)$成立时，完成A。

(b)否则，处理机根本不执行A，对系统不产生影响。

现有的解决方法是设置系统检测点，周期性地检测存贮在可靠存贮器上的处理机状态，并能从以前的存贮器状态开始重新启动。另外，还必须实现完全可靠的监控器，以控制处理机的动作。

2. 通讯子网

通讯网络的目的在于主机之间的消息交换。为了支持系统的可靠性，这种转换必须是完全可靠的，即在网络中传输的数据能正确到达目的站。下面，针对ISO模型对网络可靠性进行分析。

(1)**物理层** 该层主要处理数据传输和设施互联的机电特性。常用的可靠性技术是用奇偶和循环冗余或基于海明间距的错误纠正码等纠正码探测和/或恢复传输错误。这些代码与硬件相关，在此不作详细讨论。

(2)**数据链路层** 主要是在两个相邻通讯处理机或主机与处理机之间获得可靠、有效的物理通讯。此层常处理的不期待事件有：帧丢失，帧损坏、帧重复和设施出错。解决上述事件常使用的技术有：监视时钟，多次重发、ACKS和NAKS、帧的顺序化和多重循环纠正码或检验和。

(3)**网络层** 此层主要负责从一个源通讯设备到一个目的通讯设置的消息传输，这种传输可能通过中间设施，与此层可靠性密切相关的问题是：路由选择、存取控制和拥塞控制。

• 路由选择：分布式系统中最为有效、可靠的算法是分布的动态路由算法，这种算法的特征是：利用可知与可见状态消息及部分历史消息作出全局决策，这种算法是一种反映现行状态的近似假想算法。

• 拥塞控制：通讯网络中消息的多少与拥塞程度直接影响到网络的可靠性，拥塞所产生的不期待事件有：消息丢失、消息重复、缓冲区溢出和死锁，常用的处理方法是：缓冲区预分配、放弃消息、发送阻塞消息包或限制、不允许发送的消息数。

• 存取控制：指用于决定在某个时刻哪个设备允许使用通讯介质的一种机制，这种机制的使用主要从安全保密和功能上进行考虑。

(4)**传输层** 传输层为不同的主机之间的用户提供可靠且有效的端点服务，与此层相关的问题是：寻址、确定联接、流控制，数据缓冲、多路选择、同步、错误恢复和网络互联，传输层的可靠性依赖于要求说明和网络层的功能，若网络层支持虚拟电路，则传输层的可靠性只要做到流控制和错误恢复即可，另一方面，若网络层支持数据报，则传输层必须保证子网可靠的物理通讯。

上面我们针对ISO模型对网络软件的低四层进行了分析。与网络可靠性密切相关的另一重要问题是：网络拓扑问题。对任两通讯结点，其通讯路径愈多，可靠性愈高，同时也会导致系统开销增大，所以要提高网络的可靠性，拓扑结构的选择尤为重要，现今所有的网络拓扑中性能较好的要数C·S·Ragh-

avendra等人设计的最佳双环网[10]。而另一方面，为了适应网络出现故障时拓扑的动态变化，网络通讯软件必须具有一定的容错能力。

实现网络可靠而有效的通讯是由一套可靠的协议完成的，网络协议的描述可分成以下四个基本部分：

- 协议规则的描述
- 协议服务的规格说明
- 传输系统的规则说明。传输子系统往往用于描述所考虑的抽象层提供的协议机之间的通讯服务。
- 协议环境的规格说明。

关于协议的正确性验证是分布式领域专门研究的一个领域，其主要目的是寻求一种形式的方法给出协议的描述与证明以及协议的自动生成。

3.分布式操作系统(DOS)

DOS的可靠性涉及较广的范围，它包括进程间通讯、分布控制、进程和资源结构及容错结构的设计、结构一致性检查、数据结构的冗余及同步。在本节从以下几方面进行分析。

(1)DOS模型 在分布式系统中，几个较为常用的模型是Jone的任务队模型、Fieldman的活动模型、Liskov的Gurdians模型和Lampson的顾客服务员模型。在前两种模型中，对相互合作的相关进程给出了一种结构方法，而后两种模型，则利用抽象数据类型的概念来构造资源和服务，为分布式系统提供了一个较好的结构模型。然而，上述四种模型都采用一种结构化的方法来提高系统的可靠性，而没有考虑DOS的容错问题。

自从Reed[8] Mosk[9]相继利用事务作为一种新的容错分布式OS的模型以来，人们开始注意寻求一种新的结构，将容错有效地与分布式控制结合在一起。

事务模型最初用于数据库领域，主要处理数据读、写操作的一致性问题[5]，其管理的运算只有读、写，且数据对象单一。但事务概念的四个特性方便了事务引入DOS，同时，要使其作为一般目的分布式系统模型还必须作进一步的扩充，使得它既能支持应用中出现的各种抽象类型，又不影响系统的并行性。

支持数据抽象的系统必须允许定义具有相应类型和确定运算集的任意对象类型，必须允许通过组合现有的对象类型去构造新的对象类型，且结果类型对它们的用户是原类型。这样，事务不再是对记录的一组读、写运算，而成了对对象的一个层次类型的运算，假如层次中的某些运算本身是由事务实现的，这样，就出现了嵌套事务的概念。在嵌套事务中，由于可序列化特性的限制，系统的并行性受到较大的限制。为此，萌发了嵌套原子的概念。Lampson和其他一些研究者应用嵌套原子去构造可靠DOS的核心部分，以支持整个系统。然而，无论是嵌套事务还是嵌套原子，其容错结构都是一种树状层次结构，且容错的控制由上而下进行，虽说这种方法给出了较好的系统模块性，但对系统的效率也产生了较明显的影响。

为此，[11]提出了一基于数据驱动的分布式事务计算模型，该模型在保持系统一致性的前提下，将分布控制与容错有机地结合在一起，使系统性能得以大大改善。

(2)与容错机构紧密相关的问题：恢复与同步 恢复技术主要使出错进程恢复到某一前状态，然后重新开始执行。由于恢复对象具有不同的特性，所以统一恢复机制的设计很困难。在分布式系统中运算依其特性可分为如下三类：

(a)不保护运算：这类运算不需撤出和重作，它随着执行实体的撤除而消失，如对临时文件的运算和中间消息的传递等。

(b)保护运算：这类运算不仅能且必须对其撤出和执行，如对数据库的一些传统运算。

(c)实际运算：这类运算一旦完成就不能撤出和重执行，对这类运算的处理是恢复机制首先要考虑的问题，例如支票的分发及航空收费。

在Moss[5]与Reed[13]的容错机制中恢复采用了Shodow副本法，这种方法存在一些弊病，系统需保存过多的副本数，且在容错结构树中，每个结点动作的最后实现全由根结点集中完成，这样，大大影响了系统效率。

目前所使用的恢复方法还有：检测点与记录(checkpoints and logging)法、redo and undo协议、undo and redo协议、和反向运算(reversal operation)等。

上述各种恢复机制都存在一定的问题，选择何种有效的恢复策略有待进一步研究。

容错机制中的同步策略目的在于协调容错单元的恢复与重启动，在Moss与Reed模型中，分别采用了时间戳和封锁方法，前一种方法在全局范围内实施了一个全序，影响系统的"平等"特性，而后一种则大大影响了系统的并行性。为此，[12]提出了一基于动态优先级的同步机制，该机制切实地反映了分布进程的运算特性，使系统并行性有所提高。

在以往的容错系统中，具有这样一个共同特性：在全系统范围内实施同一恢复与同步机构，但由于容错系统处理的运算没有统一的容错特性，这样就出现试图用统一的方法去处理不统一对象的矛盾，这种矛盾是带来现有容错机制低效的原因之一。为此，能否设计一种机制将不同对象的恢复和同步与对象本身结合起来呢？

(3)进程通讯 可靠的进程通讯首先必须处理下列一些问题：

(a)逻辑信息丢失。

(b)接收到的消息与发送消息不一致。

(c)消息可能从不同的进程到达目的消息缓冲。

(d)发送者可能快速地发送一串消息。

在Lampson的原子模型中[4]，利用分层方法构造了一个可靠的原子通讯，对上述问题进行了处理，但要支持有效的IPC还必须考虑选择何种消息结构？为了实现可靠、有效的IPC，系统应需要哪些通讯原语？为了适应系统容错，原语应具有哪些特征？

在可靠的消息通讯中，另一要处理的是"ophan"事件。ophan主要是在运行过程调用中出现故障时发生。为了提高处理机效率，防止"ophan"事件破坏系统环境，并且配合恢复动作的执行，有效地消除ophan是很重要的。关于这一问题的处理Lampson给出了几个相应的解决方法。

(4)系统一致性 在分布式系统中，一个任务往往在几个不同的结点上加以执行，为了避免不必要的冲突和副作用，有效地协调活动之间的动作是必要的，活动之间一致性的保证可以减少错误恢复开销，提高系统的可靠性。

实现系统范围内一致性靠的是作业调度和分散控制，以前对一致性的考虑只限于死锁探测、解除及Byzantine问题的研究，而没有很好地解决以下几个问题：

(a)资源冲突。

(b)进程间的相互影响。

(c)冗余功能调用。

为提高系统一致性，有效地解决上述问题，A·Speetor[7]等人提出了利用语义知识辅助作业调度的方法，作业语义知识由用户交互给出。但他们在给出抽象数据描述时没有给出其相应的恢复及同步特性，不能有效地支持容错，因此，要更好地提高系统可靠性且不影响系统效率，将抽象数据类型的容错描述与抽象数据类型的实现放在一起是一个较明智的方法。

4. 分布式应用

(1)分布式数据库 在分布式数据库中，是通过实现"事务"概念来保持其一致性的，在分布式环境中，由于一个事务可能涉及多个自治结点，所以必须有一种方法解决事务的提交(Commit)和回撤(rollback)，目前DDBMS中常用的方法是二阶段协议2PL，其缺点是限制了系统并行性，为此，有人又提出了三阶段协议3PL。

在数据库中，错误处理协议分为两类：终止协议和恢复协议。终止协议主要在错误探测时进行引用以便保持事务的原子性，而恢复协议主要用于重启动系统进程且为事务重新建立一致性状态。

在分布式环境中，为了提高系统可靠性，数据库可在全系统范围内进行分割，由于多个结点的自治性及任意的通讯时延和出错导致了保持数据库语义一致性的困难，如下几个与分割数据库密切相关的问题有待进一步研究：

(a)分割数据库的探测。

(b)在分割期间，允许的用户事务执行程序。

(c)用于合并或避免合并分割数据库的技巧与方法。

(d)在网络分割和合并时的恢复方法。

处理网络分割问题有两个共同的方法：(1)确定方法，主要用于避免数据不一致性。(2)乐观方法，允许在网络分割时出现暂时的不一致性，然后在合并时给予消除，此思想也可用于系统一致性的维护，以提高系统效率。

另外，可靠的DDBMS还必须提供可靠的分布查询和多复本修改。

(2)分布式语言 以往的顺序程序设计语言，早已开始注意到例外事件的处理，但在分布式语言中近来才引起人们的注意。一个常采用的例外事件处理方法是"超时处理"。在Plilts[4]中，事务关键字的概念被用于通知它期待的下一个消息收到的时间，和用于通知进程关于错误的消息，例如，一个进程仅通过关键字去等待消息，而不需知道消息地址，因此，发送进程的错误，可以通过适当的事务关键字通知给接收进程。

随着事务概念在DOS中的地位增强，有人试图将事务概念引入分布式程序设计语言，将容错特性融入程序结构中去，但要在语言层实现事务概念，DOS必须提供强有力的支持，特别是高层原子性的实现更为困难。

分布式语言具有这样一些特性：并发、通迅、同步、时间相关、和容错。在分布式语言中实现这些特性还有待进一步探讨。

三、总 结

为了获得高可靠性的分布式系统，还有以下几方面问题值得人们进一步研究：

- 分布控制为错误恢复提供了有效的帮助吗？其关系如何？
- 嵌套原子动作适合作高层程序设计的程序抽象吗？
- 嵌套原子动作能被有效地支持吗？在哪一层进行支持？
- 实现可靠进程间通讯的最好方法是什么？
- 怎样为错误恢复和一般可靠性提供有力的理论支持？
- 怎样提供一种方法来估价无错系统所导致的系统开销？
- 如何设计一系统模型，在全系统范围内有机地融入容错技术？

参考文献

[1] B.Liskov and R. Scheiflen, "Guardians and Action, Linguistic Support for Robust Distributed Programs" Proc.Symp. on Principles of Programming Language, Dec. 1982.

[2] J.N.Gray, "A Transaction Model" Automata Language and Programming,Springer Science VoL.80, 1980.

[3] J.N. Gray,"The Transaction Concept, Virtues and Limitation" VLDB, Sept. 1981.

[4] B.W. Lampson, "Atomic Transaction" LNCS VoL.105,Distributed Architecture and Implement.

[5] J.E.B. Moss,"Nested Transaction and Reliable Distributed Computing" Second Symposium on Reliability in Distributed Software and Database Sys-

新一代知识系统需求分析*)

金 芝 胡守仁（国防科技大学计算机系）

摘 要

针对第一代知识系统存在的许多明显缺陷，许多研究者相继提出了各自的第二代知识系统的概念，在不同程度上弥补了第一代知识系统的不足，缓解了它们与知识处理需求之间的矛盾，但这些解决方法大多是局部的或不完全的。本文分析了第一代知识系统的缺陷，并在分析第二代知识系统的几个典型结构的基础上，确定了新一代知识系统的设计目标，最后提出了新一代知识系统的一种通用结构。

专家系统的出现标志着人工智能进入了知识处理的时代。从提出并实现第一代基于规则的系统到现在，已经二十多年过去了。现在，这些知识处理系统已在各自的应用领域中显示出它们的价值，并作为一种问题求解方法在当前人工智能应用系统中广泛流传。

十几年应用实践表明，第一代知识系统在某些方面确实能表现出知识系统所具备的独特功能，但还存在许多明显的缺陷。这些缺陷用第一代知识系统的方法无法克服。正是由于这些缺陷，知识系统曾一度走向低谷[10,11]

知识就是力量。人类对复杂问题的求解能力来源于人类所拥有的理论知识、经验知识和常识知识等。人工智能要用计算机模拟人类的各种智能活动，走知识处理的道路势在必行。要将知识处理的研究进一步深入开展下去，对人工智能研究者们来说，开发新一代知识处理系统成为当前一项紧迫而又艰巨的任务。

1、第一代知识系统的缺陷

第一代知识系统的特点是以领域专家的经验知识为基础的问题求解。在一些狭窄的

*) 国家高科技发展计划资助课题

tem.

[6] S.K.Shrivastava and F. Panzieri," The Design of Reliable Remote Procedure Call Mechanism", IEEE Transaction on Computer, July 1982.

[7] A.Spector and P.M.Schwartz, "Transaction, A Construct for Reliable Distributed Computing", ACM Operating System Review, Vol.17, No.2, 1983.

[8] D.B.Lomet,"Process Structuring,Synchronization and Recovery Using Atomic Actions" Proceedings of The ACM Conf. on Lang. Design for Reliable Software, SIGPLAN Notices, Vol.12, No.5, 1977.

[9] 鄢勇，刘键，"并发控制一致性理论的进一步完善"，《计算机学报》1989 第4期.

[10] C.S. Raghavendra, et al., "Optimal Loop Topologies for Distributed System", Proc. 7th Data Commun. Symp., Oct. 1981.

[11] 鄢勇，刘键，"基于数据驱动的分布式事务计算模型"，《计算机学报》1990 第8期.

[12] 鄢勇，刘键，"基于动态优先级的分布式同步机制"《计算机学报》1989第10期.

[13] D.P.Reed,"Implementing Atomic Actions on Decentralized Data",Proc. of The 7th ACM Symp. on Oper. Syst. Principles Dec. 1979.

[14] J.A.Feldman,"High Level Programming for Distributed Computing",Commu. of The ACM 22.6, June 1979.