联动北方
LANDING

企业级 IT 运维宝典系列

联动北方 出品

目　录

第 1 章 Tuxdo 概述

Tuxedo 是一个成熟多年的联机事务处理产品，是用于开发、集成、部署和管理大型分布式应用，拥有处理关键业务应用系统问题所需的性能、安全、可扩展性和高可用性，同时又易于安装、部署和管理。

1.1 什么是 Tuxedo 系统

Tuxedo（Transactions for UNIX, Extended for Distributed Operations）是在企业分布式计算环境中，开发和管理三层“客户—服务器”（C/S）关键业务系统的平台软件。它具有空前的交易处理性能、高度的可靠性和无限的可伸缩性，能为企业建立、运行和管理大规模、高性能、分布式的关键业务系统提供一个强大的支撑平台。这个平台具有很好的开放性，它支持各种各样的客户端、数据库、网络、通信方式和主机遗留系统。开发人员能够用它建立跨多个硬件平台、数据库和操作系统的应用系统。

下图展示了企业级 Tuxedo 系统的体系结构：

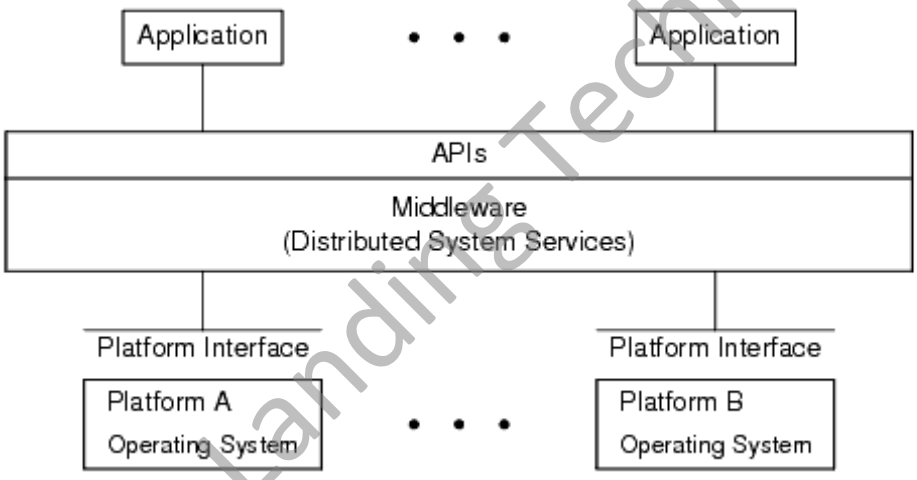

图 1-1

Tuxedo 具有全面而健壮的功能。在企业分布式联机交易系统中，Tuxedo 常常作为一个事务监控器（TP Monitor ，简称 TM）来协调分布式事务；在构建多层 C/S 应用系统中，Tuxedo 经常作为一个中间件的角色部署在客户机和服务器之间，为应用提供服务；在构建企业级应用系统中，Tuxedo 经常以一个应用服务器平台角色出现，为企业应用提供一个部署环境和运行环境。

Tuxedo 支持广泛的操作系统平台，包括 64 位/32 位的 Solaris，Linux，IBM 的 AIX、System i，HP 的 HP-UX、OpenVMS，以及 Microsoft 的 Windows。

1.2 Tuxedo 的历史及发展

Tuxedo 是一个久经考验的，成熟的系统，在 20 多年的历史中不断的发展和增强。

1.2.1 Tuxedo 的产生

Tuxedo 系统在 1983 年由美国贝尔实验室的 AT&T 分部开发，最初被命名为 UNITS（UNIX Transaction System）。开发 UNITS 的目的是便于 AT&T 内部构建基于 UNIX 的业务支撑系统。

在 1989 年 UNITS 项目转移到 AT&T 的 UNIX 实验室 （USL）时，这个 C/S 框架结构已经以“Tuxedo 系统”的名称销售了。1993 年 Tuxedo 系统被转到 Novell 公司。

在 1996 年，BEA 和 Novell 公司达成了排它协议来继续研发和出售不同平台下的 Tuxedo 系统，包括 Windows 和 UNIX 系统。2008 年 Oracle 公司收购了 BEA，Tuxedo 也转归 Oracle 旗下。

1.2.2 Tuxedo 的发展

1.2.2.1 版本 1.0 到 7.1

从 1983 年的 1.0 到 2000 年的 7.1 版本，Tuxedo 系统经过无数次的改进和扩展，目的就是为了使客户端和服务器端的通信模式更加多样化。Tuxedo 系统作为事实上的标准，演变为开放式（open standard ）的在线交易处理（OLTP）解决方案。Tuxedo 的 4.1 版本增加了 ATMI 接口以及对事务的支持。Tuxedo 对事务的支持直接导致了 XA 接口规范的产生。在 Tuxedo 的 5.1 版本出现了域（Domain）组件，它能够实现 Tuxedo 系统中多个应用程序之间的动态链。Tuxedo7.1 版本推出了安全插件架构，这为集成第三方安全系统提供了接口。

1.2.2.2 版本 8.0

Tuxedo 系统的 8.0 版本发布于 2001 年，它的总体性能比其它版本有所加强，Tuxedo 8.0 的最大特点是引进了对 CORBA 的支持。在 CORBA 域中实现多线程，统一编程和负载均衡机制。Tuxedo 8.0 可以通过 WTC(WebLogic Tuxedo Connector)部件实现与 WebLogic 的互联。

1.2.2.3 版本 8.1

Tuxedo8.1 版本发布于 2003 年，这个版本对 WTC 做了进一步的加强，集成了 XML C++ 解析器，以便更好的支持 XML 数据。Tuxedo 可以和 WebLogic 7.1 或者更高版本的 Domain 集成，进行单点安全管理。还在本地化方面做了提高，除支持英语外，还支持了日语。网络通信方面，在没有改变任何接口的同时提高了域网关的性能。

1.2.2.4 版本 9.0

Tuxedo 的这个版本发布于 2004 年，这个版本主要是在 web service 方面做了进一步的加强，提供了 XML schema 和 FML 之间的双向转换功能，同时还提供了一个用于保存 Tuxedo 服务元数据的存储库（repository）。存储库的主要作用就是在应用开发阶段保存 Tuxedo 服务定义元素等信息，以方便开发人员进行交互式查询。Tuxedo 9.0 还在域网关的性能改进、超时控制、域连接策略、CORBA iiop client 故障转移等方面都做了很大的改进。在安全方面增加了 Cert-C PKI 插件以保护数据的安全，还增加了对 Kerberos 的支持。在 Microsoft .net 工作站客户机编程方面，为开发人员提供了一组 API 和开发工具。9.0 最大的特点是 JOLT 和 SNMP 代理开始和 Tuxedo 打包在一起销售，而不再是作为单独的产品部件。

1.2.2.5 版本 9.1

Tuxedo 9.1 版本是发布在 2006 年，它作为在原来的版本基础上的一个升级版，又增加了一些新的特性， 首先是在数据库方面，提供了对 Oracle RAC 的支持。在.NET Workstation 客户机方面，提供了一组实用的工具来帮助程序员快速的开发基于.net framework 的客户机应用程序。在管理方面，允许通过远程桌面启动、访问、和关闭 Tuxedo 服务。

Tuxedo 9.1 版本发布的同时，还发布了 SALT（Service Architecture Leveraging Tuxedo），实现了 Tuxedo 应用与 SOA 的无缝集成。目前 SALT 已支持 Tuxedo 服务与 Web 服务的双向访问、数据转换、全局事务、安全控制、可靠消息等多种功能。同时 SALT 还支持 SCA（Service Component Architecture）服务组件架构，提供 SCA 容器，这样客户可以在强大的 Tuxedo 平台上构建或重用基于 SCA 规范的 SOA 应用。

1.2.2.6 版本 10.0

Tuxedo 的 10.0 版本发布在 2007 年，这个系统最大的特点是增加了 TSAM（Tuxedo System and Application Monitor）应用管理监控平台。它由 Manager 和 Agent 构成。可以为 Tuxedo 应用程序提供全方位的性能监控和管理服务。根据事件产生告警，并进行性能调优，以满足高级服务需求。

Tuxedo 在安全方面在这个版本中的得到了很大的加强，首先是在安全验证方面，提供了通用的服务器 GAUTHSVR 来统一集成外部的 LADP 服务器，。此前，Tuxedo 仅支持 LAUTHSVR 来与 WebLogic 的内嵌 LDAP 服务器进行集成验证，而对于其它 LDAP 服务器则只能通过编码来实现。在认证加密方面，10.0 增加了 ATMI 应用对 SSL 的支持，而之前只有 CORBA 才支持 SSL。

在与第三方系统集成方面，10.0 版本提供了 MQ adapter 来与 Websphere MQserices 进行双向的基于事务的连接和消息交换服务。Tuxedo 10.0 在其它方面的更新还包括提供 tpkill 命令来杀死未响应的进程，以免公告板共享内存区的数据结构遭到破坏，而不是像以前那样通过 kill 来发送 SIGKILL 信号。同时还提供了对大文件的支持，所有的 UNIX 平台上均支持大于 2G 的文件。

1.2.2.7 Tuxedo 10gR3

Tuxedo 10gR3 版本发布于 2009 年。它在网络通信协议方面增加了对 IPv6 的支持。IPv6 是由 IETF 设计的下一代协议，用来取代当前的 IPv4 网络协议。IPv6 中最明显的改进是 IP 地址长度从 IPv4 的 32 位加长到 128 位，它也改进了诸如路由和网络自动配置的许多地方。

这个版本增加了新的 API，支持在服务进程中创建用户 context，之前用户只能在系统 context（Tuxedo 服务被调用时自动创建的 context）中发送/接收服务请求和定义全局事务。

为帮助 Tuxedo 管理员监视客户端的运行合法性，Tuxedo 从这个版本开始提供了访问日志，可以记录访问的最高客户量、当前的用户访问量和指定的用户。

这个版本还进行了如下改进：

- CLOPT 长度：在 UBBCONFIG 中 Tuxedo ATMI SERVER 的 CLOPT 长度从原来的 256 增加到 1024.
- FML/FML32 字段名长度： FML/FML32 字段名长度从 30 增加至 254.
- tlisten 密码加密： tlisten 的密码经过系统加密后再保存到 tlisten.pw 文件。此前这是明文保存的。

- 动态 DMIB 更新：允许重新配置远程域网关侦听地址,而无需重启本地域。
- 域网关永久不连接策略：增加新的连接策略 PERSISTENT_DISCONNECT，表示既不主动连接其它域，也不接受其它域的连接。

1.2.2.8 Tuxedo 11g

Tuxedo 11g 发布于 2010 年。在这个版本中新增的客户端-服务器端亲和性的功能，使得 Tuxedo 客户端可以指定其一段时间内所有交易请求的路由范围，可以是严格的范围，也可以是倾向的范围。

Tuxedo 11g 支持在一个服务器组中定义多个资源管理器（RM），这样每个应用服务器都可以在一个全局事务中使用多个资源管理器。

这个版本还支持了在 Tuxedo VIEW 数据结构中再嵌套 VIEW。

先前 Tuxedo 版本支持为指定服务设定 AUTOTRAN，使得该服务被调用时如果尚不在全局事务中，则自动开始全局事务。这个版本中增加了域级别 AUTOTRAN 的配置。

1.3 Tuxedo 支持的平台

Tuxedo 支持现在市场上主流的操作系统平台。

下面列表提供了 Oracle Tuxedo 11g Release 1 (11.1.1.2.0)支持的平台详细信息：

生产商	操作系统	版本
HP	HP-UX	HP-UX 11i v2 (32-bit) on Itanium 64-bit HP-UX 11i v2 (64-bit) on Itanium
HP	HP OpenVMS	HP OpenVMS HP OpenVMS V8.3-1H1 (64-bit) on IA64
IBM	AIX	IBM AIX 5.3 (32-bit) on IBM PowerPC IBM AIX 5.3 (64-bit) on IBM PowerPC IBM AIX 6.1 (32-bit) on IBM PowerPC IBM AIX 6.1 (64-bit) on IBM PowerPC
IBM	IBM i	IBM i 6.1 on IBM PowerPC
Microsoft	Windows	Microsoft Windows 2008 Server (32-bit) on x86 with MS Visual Studio 2008 Professional Edition Microsoft Windows 2008 Server (64-bit) on x86-64 with MS Visual Studio 2008 Professional Edition Microsoft Windows 7 on x86 with Visual Studio 2008 (Client Only) Microsoft Windows XP on x86 with Visual Studio 2008 (Client Only)
Novell	Linux	Novell SUSE Linux Enterprise Server 10 (32-bit) on x86 Novell SUSE Linux Enterprise Server 10 (64-bit) on x86-64 Novell SUSE Linux Enterprise Server 10 (64-bit) on s390x Novell SUSE Linux Enterprise Server 11 (32-bit) on x86
Oracle	Linux	Linux Oracle Enterprise Linux 5.0 (32-bit) Oracle Enterprise Linux 5.0 (64-bit)
Red Hat	Linux	Red Hat Linux Enterprise AS 5 (32-bit) on x86 Red Hat Linux Enterprise AS 5 (64-bit) on s390x
Sun	Solaris	Sun Microsystems Solaris 10 (32-bit) on SPARC

Microsystems	Sun Microsystems Solaris 10 (64-bit) on x86-64 Sun Microsystems Solaris 10 (64-bit) on SPARC

表 1-1

1.4 Tuxedo 的技术架构

企业级的应用大致经历了三个阶段：以大型机为核心的“主机/终端”模式，以文件服务为核心的“文件服务器”模式和以数据服务为核心的“客户机/服务器”模式。

“主机/终端”模式属于单层结构，它的典型代表是 IBM OS/390 系统，所有数据、应用逻辑、用户界面都存放在主机上，终端机不具有持久存储和计算能力。执行应用程序时，终端将用户界面下载到本地内存中，处理能力完全依赖主机的吞吐量和通信网络的带宽。

“文件服务器”模式属于两层结构，它的主要代表是 Novell Netware 网络操作系统，所有文件和数据都存存储在服务器上，客户机是功能完善的计算机系统，负责提供用户界面并处理业务逻辑。

“客户机/服务器”模式可以分为以数据库管理系统（DBMS）为核心的两层结构和以中间件（application server）为核心的多层结构。基于 Tuxedo 中间件构建的应用系统就具有多层可管理的客户机/服务器框架结构。

1.4.1 客户端/ 服务器模式

在客户机/服务器模式下，企业应用被分割成若干个相对独立的功能模块，其中一部分模块管理着企业的系统资源并为其它的模块提供服务，称为“服务器”；另一部分模块为用户提供输入输出界面，向服务器提交请求，称为“客户机”。

如下图 1-2：

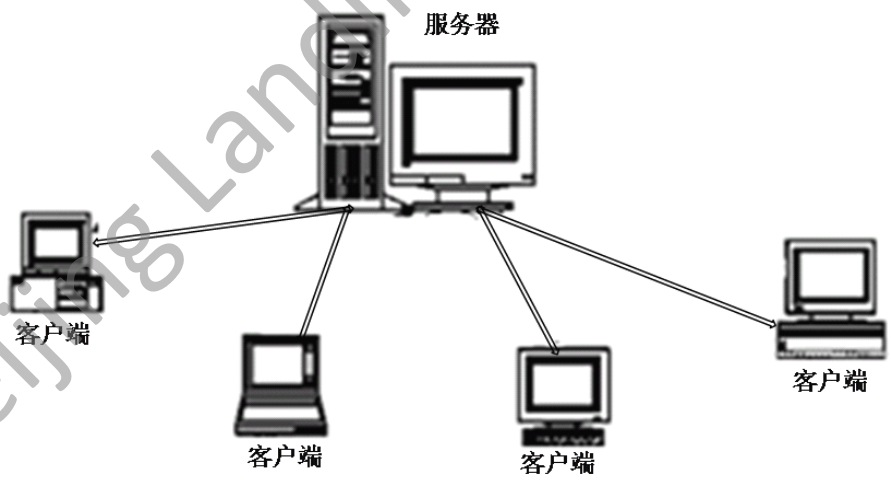

图 1-2

随着企业计算的发展，客户机/服务器模型经历了一个从以 DBMS 为核心的两层结构向以应用服务器为核心的多层结构演变的过程。

1.4.1.1 两层客户机/服务器模型

典型的两层客户机/服务器模型为：服务器端运行关系型数据库，负责提供数据服务，所有应用逻辑和用户界面都安装在客户端，客户机与服务器通过“请求/应答”式通信模型（Request/Response）进行业务逻辑处理，大大减少了网络流量。与单层结构相比，两层

客户机/服务器提高了执行效率，减少了网络通信流量，加强了模块化设计和分布式计算能力，但是随着应用规模的扩大，它的缺点也就逐渐暴露出来：

两层客户机/服务器结构无法处理大并发的用户请求：服务器没有提供连接池管理机制，连接不能在多个客户端共享。也就是每个客户端都要和数据库建立连接。这就消耗了大量的系统资源，当并发的用户数增加到一定的数量的时候，就可能造成死锁或者是系统崩溃。

系统的可移植性差：编程环境往往依赖于操作系统和数据库，客户机编程语言和数据库 SQL 对平台和产品的依赖性都很强，在操作系统或数据库发生改变的情况下， 往往需要重写整个系统。

系统的灵活性差：虽然系统分成了客户机/服务器两个部分，但他们是紧耦合的，当其中的一方发生改变时候，通常会影响到另一方。

不支持分布式事务：两层客户机/服务器结构没有分布式事务的管理能力，当一笔交易涉及到多个资源管理器时，不能保证交易的一致性。

安全性差：存放应用程序的客户机容易受到黑客的攻击，每个客户机都可以直接访问数据资源，这样会对企业级资源安全构成威胁。

可维护性差：每个客户机上都要安装操作系统和应用软件，这不但造成了巨大的资源浪费，而且系统升级时工作量会非常大。

可管理性差：对于客户机而言，不支持集中式事务管理。对于服务器而言，没有统一的管理工具。这样管理就不是很方便。

可扩展性差：两层结构的可扩展性非常有限，这类系统的容量一般由设计时决定，项目实施后就很慢扩展。

1.4.1.2 三层客户机/服务器模型

把在两层结构中部署的客户机和服务器上的业务逻辑抽取出来，单独放在一个中间层上去处理，这样子就构成了三层结构。

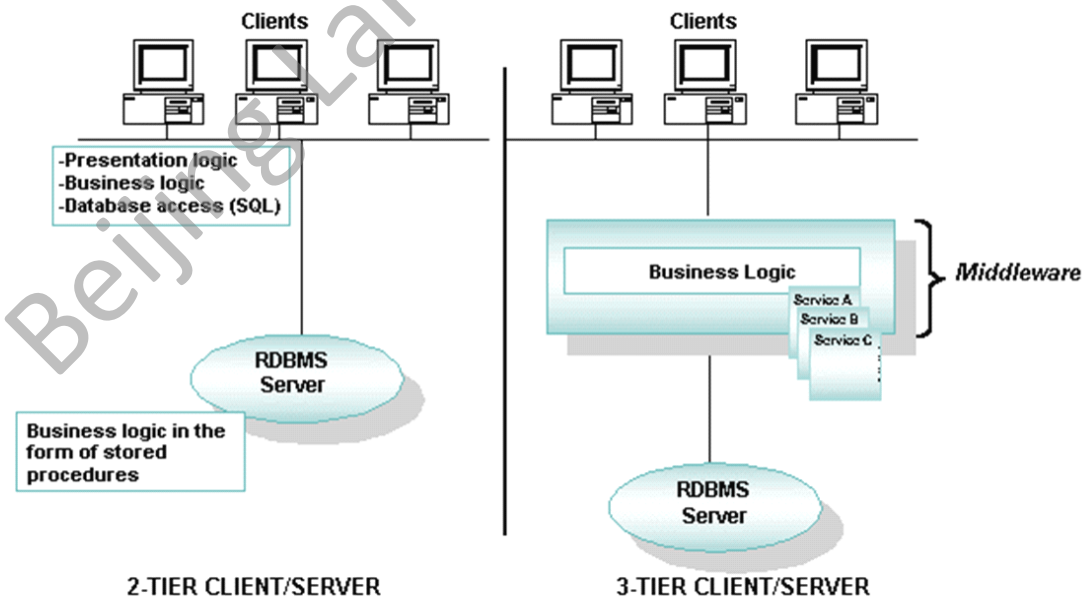

图 1-3

在三层结构中，客户机只处理表示层逻辑(presentation)，即显示用户界面、接收用户输入数据、提交交易请求、显示交易处理结果：应用服务器只处理应用逻辑，即负责维护客户机和服务器之间的通信连接，提供命名、通信、事务、安全、并发、持久性、负载均衡等应用服务，做到最大限度的利用系统资源。后台数据库只提供数据服务，即负责数据存储、查询、复制等服务。

表示层、应用逻辑层和数据层在物理分布上是非常灵活的，它们既可以同时分布在同一台主机上，也可以分布在不同的主机上。根据业务需求，可以对应用服务器层进行扩展，这样子就形成了多层结构。

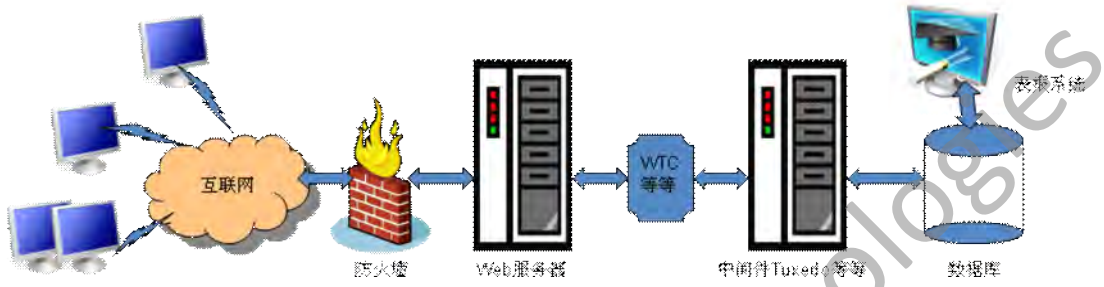

图 1-4

与两层结构相比，三层结构有如下优势：

1、实现同步、异步、嵌套、转发、会话、事件代理、消息通知、消息队列等丰富的通信机制 。在两层模式中只能实现“请求/应答”式通信，而在三层结构中则很容易实现多种通信方式。

2、支持分布式事务：分布式的交易需要分布式事务协调器（DTC）来监控事务。在三层结构中，可以把 DTC 部署在中间层上来协调全局事务 。而在两层结构中，只能使用数据库提供的单点局部事务，很难进行分布式事务的交易。

3、资源可以得到充分的利用：应用服务器可以采用多机、多线程、软件集群、软件容错和负载均衡等技术，能够充分的利用硬件资源 。

4、灵活性好：与两层结构相比较，三层结构提供了最大的灵活性是可扩展性。三层结构中的每一层都是独立的，系统容量可以在项目实施后进行动态调整。

5、高可靠性：中间件应用服务层可以通过执行队列、执行线程和连接池来控制并发请求，通过集群、负载均衡和容错机制来提供可靠性和可用性，真正做到 7X24 小时不间断提供服务。

6、可移植性好：商业化的中间件系统一般采用可移植性好的语言来开发 ，例如 C/C++和 Java 因此可以广泛的支持当前主流的操作系统。

7、安全性高：三层结构的安全性首先体现在它实现了客户机表示层和数据层的隔离，这样有效的回避了前端业务人员不小心对数据造成的永久性修改。还可以在中间应用服务器上部署软件模块实现对用户登陆进行验证、授权和访问控制。并可以通过 SSL、LLE、PKI 等实施验证和加密 。

8、可管理性强：在三层结构找中，由于业务逻辑都部署在应用服务器上，因此很容易实现集中式的管理。可以通过操作系统来管理和配置进程、线程、内存、IPC 等资源。通过商业的中间系统来管理和监控连接池、执行队列、执行线程。此外还可以在中间系统上安装 SNMP Agent ，由专业的网管软件来管理和监控 。

Tuxedo 是支撑三层结构的强大的中间件。它具备上述所有实用特性，并且性能高，稳定性好。它不仅部署灵活，而且针对不同的部署进行了相应的优化。

1.4.1.3 本地客户端

本地客户端是指与 Tuxedo 服务器在同一台机器上,不用通过网络就可以访问到 Tuxedo 服务的客户端。客户端(进程)和服务器端(进程)通过 IPC(进程间通讯)进行通讯。当然本地客户端也可访问其它的服务器，这样的话就成了远程客户端了。由于共享内存是实现进程间通讯的最快方式，而本地客户端正是通过查询共享内存中公告板（BB）的信息，将请求发送到指定服务进程的消息队列，因此提高了请求服务的速度。

1.4.1.4 远程客户端

WORSTATION CLIENT(远程客户端):是指要通过网络才可以访问到 Tuxedo 服务器的客户端。远程客户端与 Tuxedo 服务器不在同一台机器上，它通过网络连接到到 Tuxedo 的代理进程（WSL/WSH），再由代理进程执行服务调用，并将服务响应返回给客户端。这样实现了客户端的灵活部署，和与服务器端的隔离。

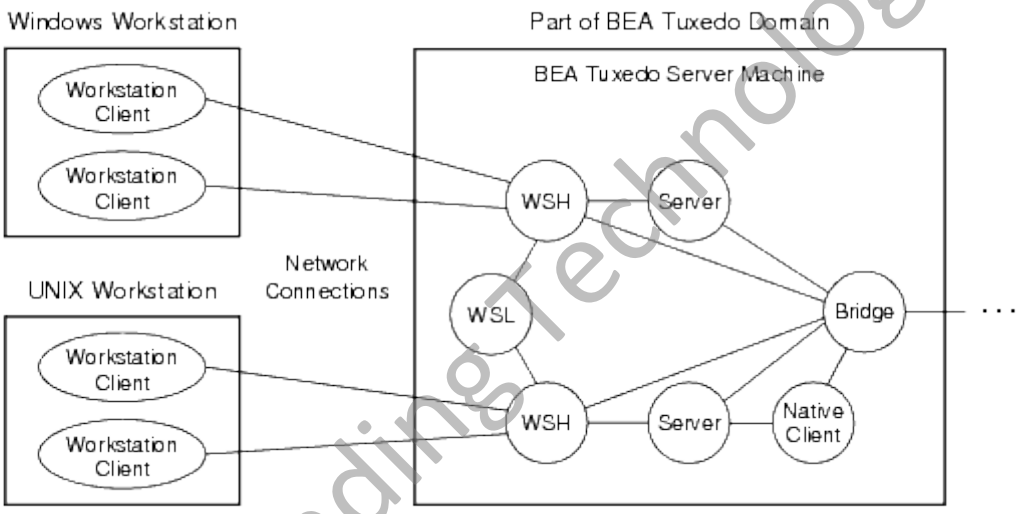

图 1-5