第 1 章　网络虚拟环境的前景和挑战

场景 1 在伊拉克的沙漠地带,美军的坦克、飞机和步兵团正在进行一场军事演习,模拟与装备有化学武器的敌人进行交战。坦克进行着激烈的交锋，军队频繁地进行战略战术上的调整，以避免伤亡。整个战场看上去非常真实，但是参战人员并不会真正遭受到身体上的伤害。这是因为所有参与人员其实都身处美国的维吉尼亚州和加利福尼亚州，坐在标准尺寸的坦克和战斗机模拟器内进行虚拟的演习。明天，他们将再进行一次军事演习，演习地点的地形与今天稍有不同，而对手则是一支更具威力的炮兵部队。

场景 2：某个工程小组正在为新一代的商用客机设计机翼。工程师们分别隶属于各地的承建商，他们分别位于西雅图、奥兰多和法兰克福。当这个小组准备好自己的设计方案后，便与圣迭哥的新型引擎研发小组联系。然后，引擎研发小组启动了位于圣迭哥的超级计算机，与位于纽约的另一台超级计算机建立连接之后，便使用其中存储的机翼设计方案进行引擎动力系统的模拟试验。两组工程师同时观测了飞机的模拟飞行，之后断开连接，并对各自负责的部分进行改进。

场景 3：George 想为家人和朋友购买圣诞礼物，但他总没有时间去商场选购。于是，他上网选择了一家虚拟的大型购物中心，那里有各式各样的商铺和特色小店，足够他随意选购。George 看中了一款鞋子，他使用鼠标进行旋转操作以便从各个角度观察鞋子，并向身边的售货员咨询了几个问题，最后他决定买下这双鞋子。George 输入了自己的信用卡号，这样他隔天就能收到这双鞋子。

场景 4 家住达拉斯的 Jeremy 今年 10 岁,他最喜欢做的事就是拨号上网,玩一个名为“疯狂武士”的网络游戏。今天，Jeremy 在游戏中扮演的角色是一名日本武士。开始游戏后，他遇到了一群爬虫和异形生物，其中一种生物实际上是由一个名为 Lucy 的女孩扮演的,她今年 11 岁，也正在伦敦的家中玩这个游戏。Jeremy 和 Lucy 决定联合对抗其他玩家。持续了好几分钟的一场恶战结束之后，他们的角色的能力都升了一级。

以上这些场景描述的都是网络虚拟环境的应用。由此可见，网络虚拟环境已经广泛应用于生活的方方面面,包括教育培训、工程设计、经济贸易、休闲娱乐等。实际上,随着研发人员不断地将网络虚拟环境技术用于节省开支、缩短上市时间、代替人们进行危险活动等方面，网络虚拟环境的应用范围也在迅速扩大。

1.1 什么是网络虚拟环境

网络虚拟环境(net-VE)是一种软件体系,可以支持世界各地的用户进行实时交流。这种系统典型的用户交互方式是：网络虚拟环境以连接在其上的计算机工作站或控制台作为用户接口，用户通过这些接口访问网络虚拟环境的内容。网络虚拟环境通过合成逼真三维图形和立体声，带给用户真实感和沉浸感。网络虚拟环境具有以下 5个特性。

1. 空间共享性。虚拟环境提供给所有参与者身处同地的感觉，例如在同一个房间里、同一幢建筑物或同一个地区内。这个共享空间是一个可供用户相互交流的公共场所，它可能是真实存在的，也可能是虚构的，但都必须保证提供给所有参与者的数据和属性完全一致。例如，所有参与者感觉到的温度和天气必须一致，听到的声音必须一样。虽然这个共享空间并不一定需要具体的图形表示，但是具有强烈逼真效果的网络虚拟环境都提供了共享空间的沉浸式三维图形表示。
2. 表征共享性。每个参与者进入共享空间后，就成为一个虚拟人，也称为用户替身(avatar)。用户替身包括一个具体的图形表示、身体结构模型(例如手臂、腿、触角、触须、关节等)、动作模型(例如关节可能进行各种动作)、身体属性模型(例如身高、体重等) 和其他一些属性。用户替身并不一定按照人体结构进行构建，它可以是动物、植物、机器、外星人或是其他类型的物体。参与者进入虚拟环境的共享空间后，能看到其他替身的分布情况，还能看到新的替身进入场景；当某个参与者离开共享空间后，其他人也都能知道他已经离开。并不是所有的参与者都必须是人为控制的，有些可能是合成体，它们是由事件驱动模拟模型(event-driven simulation model)甚至是基于规则的推理机(rule-based inference engine)来控制的。
3. 时间共享性。在虚拟环境中，参与者能够及时地看到彼此的行为。也就是说，网络虚拟环境应该支持实时交互。
4. 通信方式。支持可视性是高效网络虚拟环境的基本功能。除此之外，大多数网络虚拟环境都还能支持参与者之间利用其他形式进行通信，例如手势、文本或语音通信等。这些通信方式增强了虚拟环境中用户需求的真实感，同时也是工程模拟系统和培训模拟系统中基础性的组件。
5. 共享方式。如果网络虚拟环境具有上面提到的各种特性，那么已经能够有效地支持一个高质量的视频会议系统了。然而，判定一个网络虚拟环境是否真正功能强大，主要依据是用户实际的交互能力。这里的交互不仅是指用户之间的交互，也包括用户与虚拟环境之间的交互。例如，在战争模拟系统或网络游戏中，用户可能会互相射击或者相互碰撞，用户也可以对环境中的物体进行获取、移动、操纵等操作，用户

之间还应当可以互相给予物品。网络虚拟环境设计者甚至支持用户对环境本身进行操作,例如建造碉堡、在黑板上写字,甚至摧毁虚拟空间。

总的来说，网络虚拟环境利用沉浸式图形图像支持多用户之间的交互、信息共享以及用户对物体的操控能力。网络虚拟环境允许多个用户同时出现，这一点是标准的虚拟现实系统或游戏系统无法做到的。它还实现了对象共享，这一特性将其与传统的聊天室区别开来。同时，它的实时交互能力又是传统的网络浏览器或电子邮件所不具备的。如果你的应用程序希望能够在共享空间中显示远程用户，那么网络虚拟环境尤为适用。在这样的应用中，用户要求系统提供的交互真实度达到如同面对面交流的效果，这些都只能依靠网络虚拟环境来实现。

网络虚拟环境具有 4 个基本构件:(1)图形引擎和显示器;(2)控制和通信设备;(3)处理系统;(4)数据网络。这些组件协同工作,为各地用户提供虚拟环境沉浸感。

1.1.1 图形引擎和显示器

图形引擎和显示器构成了网络虚拟环境中用户接口的基础。显示器为用户提供一个进入虚拟环境的三维视窗，图形引擎则生成在显示器上显示的图像。传统上，这样的图形处理能力只有高端的图形工作站才具有，不过近几年来，标准的 PC 也具备了足够的图形处理能力。一些附加式的高速图形处理器不仅价格合理，而且还可使 PC 能够获得和中低端图形工作站相同的渲染能力。此外，标准的 OpenGL 图形接口也支持密集图形应用的可移植性[OpenGL97]。因此,如今不少低端的游戏机,例如 Nintendo 64、Sony PlayStation 以及游乐中心的游戏控制台等，都能满足网络虚拟环境的显示要求。

但是，虽然传统的显示器可以显示高质量的三维图像，但是在提供沉浸感这一方面它们还有一定的局限性。在这类显示设备中，用户很容易受到外在光线以及周围环境的干扰。为了保证更强的沉浸逼真效果，网络虚拟环境系统使用图形图像设备阻隔来自虚拟环境外部的所有可视化干扰，将用户完全封闭在环境的内部。例如，通常可以将一些小型的图形显示器嵌入头盔式显示器(HMD)的护目镜中。这类显示器直接在用户的眼前成像，基本上完全屏蔽了外在光线的干扰[Sutherland68]。HMD 内部具有磁感应器，可以探测用户的头部活动，并把得到的信息提交给附加的处理器进行处理。所以当用户转动头部时，显示画面就会自动改变视点。

CAVE 也是一种沉浸式图形显示器，最初由芝加哥伊利诺斯州立大学开发[CruzNeira+93]。CAVE 本质上是一个五边形立方体,用户位于立方体的中心,图片映射到用户前方、上方、下方以及用户两侧的墙面上，因此用户都拥有 270°的完全视角。当用户在虚拟环境中行走时，画面不断进行更新,并映射到 CAVE 墙面上，这样用户的移动就具有较好的平滑感。CAVE 显示模型已经应用于许多导出接口中，例如多台动态显示器可进行坦克模拟操作，也可用来模拟发布军事命令，它们都提供给参与人员以真实战争环境的沉浸感。

1.1.2 控制和通信设备

虚拟环境必须提供给用户移动、获取和操纵物体以及与其他参与者通信的能力。许多输入设备都能支持用户进行此类操作，其中最为常用的是键盘和鼠标。用户在环境中漫游时，可以使用鼠标改变方向或进行旋转。鼠标也能控制用户的移动速度，并进行一些交互性操作，例如发射子弹或者操控一个共享工程模型。键盘则能支持用户通过键人文本信息进行通信或进行一些比较特殊的操作。

虽然键盘和鼠标是最常用的控制设备，但并不是最有效的。

- 在游戏平台上，通常用操纵杆代替鼠标。
- 一些更为精确的操控任务中会用到数据手套这种控制设备。用户可以将数据手套戴在手上，该手套内部装有传感器，用来监测和捕捉手和手指关节的动作。
- 前文中提到的 HMD 也是一种控制设备。其内部带有磁感应器，用以捕捉用户的观察方向和角度。
- 对于全方位沉浸式环境,例如在 CAVE 中,行为探测器安装在 CAVE 的墙上,可以监测和度量用户真正的身体移动。
- 用户身上可以携带一个监测设备，通过计算动作产生的力的大小来判断用户发生了哪一类行为。

虽然文本通信能简单实现，但是会破坏网络虚拟环境的整体沉浸感，而实现整体沉浸感恰恰是网络虚拟环境的目标之一。因此，在更为成熟的网络虚拟环境中，用户都是通过麦克风进行语音通信的。计算机收到其他参与者的音频信息，与各种数据流进行合成，最终通过一组扬声器播放出来。音频反馈相当复杂，不仅包含参与者的声音，还包含虚拟环境中各种动作所产生的声音效果。

1.1.3 处理系统

众所周知，网络虚拟环境要求极强的处理能力。近几年来，由于处理器价格的下降，研究人员能够不断增加处理器的数量，这样就促进了网络虚拟环境配置的不断升级。处理器单元接收到用户输入设备的事件信息，并利用这些输入信息计算出用户在虚拟环境中的位置变化情况以及环境中其他对象的位置变化情况。处理器还能决定何时以何种方式把这些变化通知给其他参与者。当虚拟环境中其他参与者发来信息，告知他们的位置和行为时，处理器将做类似处理。处理器还会为本地主机自行控制的网络虚拟环境对象进行建模，并最终生成图形显示，使得用户在虚拟环境中能够一直保持一个最新的窗口。

从历史经验来看，我们发现图像的生成需要大量的资源支持。事实上，如果以保证整体沉浸感为前提，那么任何可用的处理器都能从工作周期中划出一部分时间来，以更快的帧速

率渲染更高质量的图形图像。因此，网络虚拟环境设计者所面临的挑战之一是：对于无数的请求任务，如何合理地分配处理器时间，使得所有用户都能正确地显示网络虚拟环境的状态。

1.1.4 数据网络

网络虚拟环境的参与者通过网络系统进行信息交换。例如，当用户在虚拟环境中移动时，他必须通过网络发送更新信息，这样其他用户的显示器才能够正确地显示他的位置。同样，如果一个用户得到了环境中的某个物品，应当通知其他用户该物体已被占有。网络系统还应当对网络虚拟环境中的共享状态进行同步操作,包括天气和烟雾、时间、地形等,并能支持用户之间进行文本、音频及视频通信。

多年以来，网络虚拟环境只能在大学、大型的军事基地和商业机构内使用，这是因为这些地方有以太网或令牌网等高速的局域网。由于网络容量有限，而且网络虚拟环境系统还无法对网络容量进行合理化管理，因此网络虚拟环境能同时支持的用户量非常有限，通常不超过 12 个。同样，因为网络容量的局限性，网络虚拟环境无法应用于因特网。为了能同时支持来自于多地区的用户，网络虚拟环境必须建立在一个高带宽的私有网络上，比如国防仿真因特网(DSI)[①]。

近 5 年来，全球的网络系统发生了翻天覆地的变化。局域网的容量已经扩大了 100 倍，以太网的带宽从过去的 10 Mbps 增加到如今[②]的 1 Gbps。因此,利用局域网作为通信手段,现在基本的网络虚拟环境通常都能支持数以百计的用户同时进入环境中，而更为先进的网络虚拟环境因其更加合理地对可用的网络容量进行管理，因此能够同时支持数千名用户。

通信技术的发展也使得网络虚拟环境摆脱了私有网络的限制。调制解调器的速度自 1993 年以来已经增长了 4 倍,从过去的 14.4 Kbps 到如今的 56 Kbps,这使得用户可以在家中直接与因特网服务提供商建立连接，加入到网络虚拟环境中。这些事实都说明了在教育和电子商务等应用领域，网络虚拟环境还有巨大的发展潜力。

如今，因特网也正逐渐成为网络虚拟环境系统的可用平台。随着因特网用户的快速增长，广域网容量也急剧扩大。同时，网络虚拟环境也开始与网络浏览器相结合，例如用户利用虚拟现实建模语言 (Virtual Reality Modeling Language, VRML) 可以从万维网上下载交互式的三维模型[Hartman/Wernecke96]；Living Worlds 标准增加了多用户对这些 VRML 模型的获取能力。此外，也出现了网络虚拟环境专用的网络协议——虚拟现实传输协议（Virtual Reality Transport Protocol, VRTP）就是针对于网络虚拟环境的可用网络协议。现在，标准的网络浏览

① DSI(Defense Simulation Internet)是由美国国防部高级研究计划署（DARPA）建立的私有、租用线路型网络,它由 1.5 Mbps的 T-1 型网络连接到 BBN 路由器上,使用 ST-II 协议。如今它已经广泛应用于国防仿真试验中。

② 文中的‘如今’是指原书的写作时间,即 1999 年。——译者注

器已经成为网络虚拟环境的执行引擎，因此因特网将会逐渐成为网络虚拟环境系统最常用的网络平台。

1.2 网络虚拟环境的设计和开发所面临的挑战

网络虚拟环境一直被认为是难以正确和有效地实现的。由于实现网络虚拟环境通常是将多个传统类型的软件组合成为一个单独的应用程序，因此增加了它的复杂性。通常，我们认为网络虚拟环境是下列分支的组合。

- 分布式系统。它们必须处理网络资源管理、数据丢失、网络故障以及并行性带来的所有问题。
- 图形应用。它们必须具有足够快的帧速率，使得画面能够平滑地实时显示。同时，还要为渲染及其他任务合理地分配 CPU。
- 交互式应用。它们必须支持用户数据的实时输入。对用户来说，虽然他们分布于多个远程主机上，但是所感受到的虚拟环境应该如同在本地机器上一样。

由于网络虚拟环境系统必须提供大量已存在的应用服务，这使其变得更加复杂。

- 典型的网络虚拟环境必须与数据库系统相结合，这样才能在虚拟环境中存储持久性信息。比如，这些数据库存储的数据应当包括：关于环境的地势海拔等的详细信息，环境中的建筑物及其他静态物体的地理位置信息，初始化网络虚拟环境所需的配置信息。
- 网络虚拟环境应当支持用户认证，并能与商业及其他事务系统进行交互。
- 为了支持工程系统的可重建性，网络虚拟环境系统必须能实时地将事件记录到持久性存储器上。实际上，这个工作相当复杂，因为系统中的任一主机都不清楚网络虚拟环境的整体状态。

在网络虚拟环境中，组成元素的交互方式比较复杂，因此设计者必须把应用程序看成一个统一的系统。不过必须记住一个永恒的原则：对网络虚拟环境中的某个方面进行优化后，必将会对其他方面产生负作用。因此，网络虚拟环境系统的研发必须对工程领域中的某些方面进行折中平衡。这一节描述对网络虚拟环境具有影响的一些因素，并对受这些因素影响的一些设计问题进行说明。本书的其他部分将更为详细地介绍这些因素，并说明如何在网络虚拟环境的设计中解决这些问题。

1.2.1 网络带宽

网络虚拟环境依靠数据网络交换虚拟环境的当前状态信息。当某个用户希望得到另一

个用户详细的行为信息时,必须通过网络进行信息传送。如果希望获取更多的细节,则需要传送更多的信息。随着越来越多的用户进入网络虚拟环境，虚拟环境所产生的信息量也在不断增长。

但是，有限的网络容量资源使得网络虚拟环境设计者必须制定合理的资源分配机制。例如，当用户通过调制解调器与网络虚拟环境建立连接后，由于调制解调器只能提供最低限度的网络容量，因此用户不能期望实时地获得网络虚拟环境中所有其他参与者的详细信息。

1.2.2 异构性

现实中的网络虚拟环境并不要求所有用户都连接到相同的设备集上。例如，有些用户可能将带有鼠标键盘的图形工作站与电话线连接起来拨号上网，而其他用户可能会用完全沉浸式的头盔式显示器和数据手套，它们都连接到一个与以太网相连的多处理器机器上。要实现这些功能，就需要异构访问端口。然而，虽然异构访问端口作用巨大，但也面临着许多问题。

首先，网络虚拟环境设计者必须决定是否隐藏用户之间网络容量和网络速度的差异。不同用户可能会通过具有不同网络容量的网络系统来与网络虚拟环境建立连接，这就造成网络的异构性。因此，有些用户可能会比其他用户接收到更多的信息。将系统简化至“最小公分母”状态可以隐藏这些异构性。因为在这种状态下，网络需求量就等于网络连接的最低容量。虽然网络最低容量能保证所有用户得到的信息是相同的，但它却意味着有一个“网络容量低”的用户出现，这将导致所有用户对虚拟环境真实感体验程度的降低。另一种改进方法是利用所有可用的网络资源。但是，如果设计者选择了这一方法，即使用户得到了不同程度的信息，但在交互时也要遵从“公平竞赛”的原则。这一点在游戏以及一些交互式培训应用系统中尤为重要，因为一旦缺乏公平性，就会减少游戏的乐趣，也有可能使得培训出现问题。

其次，在图形显示、可计算以及音频容量等方面也会出现异构性。例如，一些用户的图形工作站可能每秒能渲染几百万个多边形，并能清晰地描绘出纹理，而其他用户可能只有低端的 PC 显示器，每秒只能渲染几百个多边形，而且无法绘出纹理。某些机器可以播放环境中音频信息，而其他机器可能根本无法播放音频。因此，网络虚拟环境设计者必须再一次决定到底是使用最小资源标准[①] 以保证参与者之间的公平性，还是显式地给出所有的区别然后再解决由此带来的公平性问题。

用户接口属性之间的交互方式通常都比较复杂，甚至可能是违反常理的。例如，在军事演练系统中，有些用户可能在图形显示和音频处理上有局限性，但事实上与装备更优的用户

①即将性能最低的网络连接作为整个虚拟环境中的标准，以使得所有用户能公平参与。

相比，他却可能在作战过程中占据优势。由于只有有限的图像资源，因此配置较低的用户在对网络虚拟环境中的参与人员进行建模和渲染时，就会消耗掉所有的处理周期，主机无法再分配足够的处理周期来渲染场景中的植被和可能出现的伪装。但这却带来了好处：由于没有额外的图像干扰原本已经很有限的可见度，用户就可能进行一些在其他情况下做不到的行为，例如他们能够射击原本看不见的敌人。因此，如果设计者希望对异构系统进行统一管理，那么就必须合理地控制可用资源的分配，以避免在网络虚拟环境中发生不切实际的交互状况。

1.2.3 分布式交互

分布式交互是网络虚拟环境系统中最基本的一种性质。用户可以实时地看到彼此的行为，并且实时做出反应。在某些系统中，工程模型可以动态响应其他机器上的系统组件发生的动作。为了加强这一效果，网络虚拟环境系统必须让用户觉得整个虚拟环境就在本地机器上，他们的行为都能及时在环境中反映出来，同时隐藏由应用程序本质上的分布性所引起的错觉。

由于需要在网络虚拟环境中通过消息进行信息交换，因此维持分布式系统的透明度（即让用户觉得系统就在自己的主机上）是比较困难的。举例说明，在用户进行信息交换时，网络会产生明显的时延，也就是说发送出信息和信息抵达目的地这两个时刻之间存在明显的时间差。而且，由于网络类型的不同，以及源站和目的站位置的不同，在传送不同的消息时会产生不同的时延。因此，在各用户尽力保持一致的实时网络虚拟环境时，要面临这样一个问题——所有来自于远程用户的信息，在其到达时已经过时了。

当多个用户或组件直接进行交互时，控制网络时延就相对困难一些。例如，网络虚拟环境必须支持在参与者之间进行精确的冲突检测，保持一致的协议和解决方法。进行精确的冲突检测是比较困难的，因为在任何时刻，没有用户能准确知道其他用户这一时刻精确的位置信息。正如我们所看到的，网络时延使得得到的一切信息都过时了。因此，某个用户很有可能会利用这些已经过时的信息推算出将发生一个冲突，而实际上在网络传输耽搁的这段时间内，发送方已经发生了移动并且避免了冲突。同时，由于每个用户遭受到网络带来的时延各不相同，因此对于是否会发生冲突这一问题可能会推算出完全不一样的结论。即使用户就是否发生冲突达成了一致，他们还必须在许多其他问题上达成共识，比如冲突发生的准确地点、物理意义上产生的力的大小、对参与者造成的影响等。如果冲突涉及两个以上的环境中的物体，那么其复杂度将呈指数增长。

在网络虚拟环境中，用户直接进行交互十分常见。比如在游戏中，判断子弹是否击中目标时就需要进行冲突检测，甚至于一个看上去很简单的握手动作也涉及冲突检测。一个工程模拟系统必须持续计算车辆行使过程中与道路之间的交互情况，或者需要计算飞机机翼

之间的风向与风速等。这些交互过程中包括有摩擦力、热甚至音量信息。网络虚拟环境本质上的分布性还使得音频传输更为复杂，这是因为接收方必须选择正确的方法来消除音频中由于虚拟距离和动态环境带来的影响。

1.2.4 实时系统设计和资源管理

实时交互定义了网络虚拟环境应用的进程和线程结构体系。在网络虚拟环境中，多个不同任务同时竞争 CPU 资源,而且与其他系统不同的是,网络虚拟环境中基本上所有的任务都有严格的实时限制。因此，对于网络虚拟环境设计者来说，如何满足各种不同任务的实时要求是一个巨大的挑战。

网络虚拟环境应当支持与本地用户进行实时交互。软件设计必须能够及时发觉并快速处理用户的行为（行为可能来自于键盘、鼠标、HMD 等设备）。若对这些行为的反应存在延迟，则会使得用户动作迟缓，并打乱用户的交互过程。例如，图形画面的生成速率一般是固定的,比方说每秒 30 帧，对其进行处理时若有时延可能会造成画面抖动，削弱网络虚拟环境的沉浸感。若没有其他任务，所有可用的 CPU 周期通常都可以用来生成更高质量的图像。但是，网络数据包也应该在可用时尽快进行处理，否则由于数据包的传输时延和处理时延，会在渲染远程用户时引起更高的错误率。最后要注意的是，网络虚拟环境应用中的物理建模和冲突检测必须每秒执行多次。

网络虚拟环境设计者可以应用许多技术对这些任务进行管理。其中一种方法是使用一个单一线程，其使用权被各个任务轮流获取。该线程循环速度必须足够快，以满足所有任务的实时要求。当然，一个应用也可以划分给多个线程进行处理，线程之间通过协调和安排，合理地使用 CPU。除了制定有效的 CPU 使用方案外，多线程网络虚拟环境的实现还必须解决好各个主机上存储的共享数据结构的争用问题。必须使用共享锁来协调状态的更新（响应用户的输入，网络数据包的到达，虚拟环境的建模和仿真）及状态的获取（响应图像的刷新,网络数据包的传送,虚拟环境的建模和仿真)。

1.2.5 故障处理

在分布式系统中，通常任何时候都可能出现若干个在线用户掉线（由于死机或用户自己关机),系统必须能对这类问题进行处理。此外,网络连接也有可能失败,或者出现暂时的或持久的网络连接不可用。这些可能出现的故障会给网络虚拟环境的设计带来很大影响。

网络虚拟环境设计者必须确定何种程度的故障会影响应用程序的执行。通常将故障的处理分为以下 4 类。

1. 系统终止。如果在故障中丢失的资源对于网络虚拟环境的执行来说是至关重要的，那么这个故障可能会导致整个网络虚拟环境终止运行。例如，某个系统使用一个集

中式服务器接收和分发所有的数据，一旦这个服务器出现故障，就很有可能使整个虚拟环境都无法正常运行。虽然集中式处理方式能够很好地维护系统的正确性和真实性，但它可能引起的运行终止通常被视为所有故障中最坏的情况。

2. 系统关闭。有些故障可能不会对网络虚拟环境中已有的用户造成影响，但它将阻碍新用户的加入。例如，当认证服务器发生故障时，新用户就无法再进入模拟系统中。系统中已有的用户已经通过了网络虚拟环境的认证，因此认证服务器出现的故障对其没有任何影响。

3 系统阻塞。有些故障可能只会损害网络虚拟环境提供给用户的服务的质量。例如，存储天气信息的服务器出现故障，意味着用户无法接收到虚拟环境中实时的天气更新信息，但是用户还是能够完成环境中其他任务。同样，如果只是某个用户主机出现了问题，那么其他用户只是看到他从场景中离开了，自己的性能并不会受到影响。虽然如此，这一类故障却可能对虚拟环境造成广泛的影响，造成的问题可大可小。比如，一艘航空母舰的模拟器与系统的连接中断，并从场景中消失，那么所有航空母舰上的飞机将会沉入海底，进而消失。

4. 系统继续。有的故障对网络虚拟环境造成的副效应是微不足道的，例如一些次要的服务器（如日志服务器）出现问题所引起的故障。在这种情况下，即使这些问题可能会影响一些功能，例如必须记录应用程序并重新执行，但是网络虚拟环境的整体运行进程不会受到影响。即使主服务器出现故障，系统也可能继续运行。这通常是因为系统中具有一个“热备份”(hot backup)服务器,该服务器拥有主服务器状态的镜像，支持相同的服务。因此主服务器出现故障后，它能够很快地代替主服务器进行工作。

对于网络虚拟环境来说，系统继续应该是最为理想的故障类型，因为它不会中断网络虚拟环境的执行。为防止以上各类故障的出现，需要花费大量的资金投入，以保证应用的连贯性。使用“热备份”服务器需要额外的硬件设备和网络资源，而且对主服务器的状态做镜像可能会减慢主服务器的运行速度。

通常这些故障不会单独出现，这就给故障处理带来了更大的复杂性。例如，某个网络出现故障，可能会造成若干用户同时失去与网络虚拟环境的连接。类似地，某个用户可能代表网络虚拟环境提供多种服务，那么当它的连接丢失后，其他用户将无法获得它所提供的所有服务。因此，网络虚拟环境设计者必须为用户及网络合理地分配资源和服务，保证并发故障对网络虚拟环境造成的破坏在可接受范围之内。

1.2.6 规模可扩展性

通常，我们可以通过计算同时参与到网络虚拟环境系统中所有实体的数量来衡量系统

的规模（或大小）。所谓网络虚拟环境实体，是指虚拟环境的各个参与者单独模拟的对象。网络虚拟环境实体可以是人为控制或计算机控制的车辆、地形情况（以及相关属性，如石块、树木、建筑物等），甚至可以是逻辑对象，如当前的天气状况，或者是一个对象群体。对于系统的规模没有严格的限制，可以是只有两个实体的简单游戏系统，也可以是拥有成百上千个实体的复杂的工程培训系统。另外两种度量规模的标准是网络虚拟环境参与者的数量和它们之间的物理距离。

与网络虚拟环境规模相关的因素多种多样，包括网络容量、处理器性能、渲染速度、共享服务器的处理速度及吞吐量等。对网络虚拟环境进行扩展将耗资巨大，因为它几乎要求从各个方面对网络虚拟环境进行改进。

在虚拟环境中，由于所有的参与实体都可能进行交互，因此从理论上看参与实体的增加会造成网络虚拟环境的复杂度呈指数增长。网络虚拟环境参与实体的任意组合都可形成一次交互过程，因此所有两两实体间产生的交互数为 2^x，其中 x 为实体数。不过实际上真正的交互次数并不会如我们想像的这么多，这是因为通常网络虚拟环境的某个实体不太可能与其他所有的实体都进行交互。例如，在一个大型的网络虚拟环境中，我们很少看到所有的参与者聚集在一个房间内，即使他们在一个房间里，也不太可能是处在房间的同一块区域内。相反，用户可能分成小组或小群体，分散在整个虚拟空间中的不同地区。虽然这些实体小组的组成人员可能会在网络虚拟环境的执行过程中发生变化，但是在某一时刻只有同一个小组中的用户会发生交互行为。所以小组的存在大大减少了用户的交互次数。

1.2.7 部署和配置

在进行软件部署时，网络虚拟环境设计者也将面临巨大的挑战，他们必须考虑存在潜在用户的可能性。如果网络虚拟环境的客户端软件的规模非常庞大，而且又是一个无法分割的整体，那么它就不适合用户下载。相反，如果客户软件是围绕着一个小规模的核心库设计而成的，那么根据网络虚拟环境执行过程中需求的改变，用户可以方便地进行动态下载。这些不同的包装策略会影响软件整体的设计方案、编程语言的选择、软件可用的支持平台等问题。

如果网络虚拟环境在网络浏览器内部运行，并通过因特网通信，那么部署策略就更加复杂。在这种情况下，网络虚拟环境设计者必须保证：

- 能方便地下载虚拟环境。
- 网络虚拟环境的实现必须与可用的浏览器的安全限制相一致。
- 在不同的浏览器平台上，软件都能正确地运行和显示。网络浏览器最大的缺点之一就是在下载应用程序时无法保证一致性，而标准的三维图形图像标记语言（如 VRML）还没有真正做到在不同的浏览器之间实现可移植性。

良好的网络虚拟环境部署不仅包括软件的发布，它还必须保证环境参与者能够获得系统的配置信息，包括数据传送的网址、天气及地形服务器所处的位置、安全密钥和访问代码等，并能为不同类型的参与者提供不同的图形图像和计算模型。不同的网络虚拟环境具有不同的配置信息，并且该配置信息与网络虚拟环境本身相比，其规模也不容忽略。网络虚拟环境设计者必须对这些配置信息进行收集和整理，使其对所有用户都是可用的。这些信息可以作为一个独立的发布单元，也可以是增量式的可下载单元。

1.3 小结

多年来，网络虚拟环境一直被认为是科幻小说的产物［Stephenson92，Gibson85］。利用电脑空间不仅可以省去旅行需要的大量费用，还可以消除现实世界中许多的物理限制，这些都激发了许多梦想家的想像力。但是迄今为止，网络虚拟环境也还仅仅处于实验阶段。

更快的处理器、更强的图形硬件、更大容量的网络系统支持了网络虚拟环境的发展，使得模拟系统具有更多的虚拟实体，并且能展示更为细节化的实体外观和行为。因此，网络虚拟环境正开始逐步应用于多用户的游戏系统、军事演习、教育培训系统、协作工程系统等领域。这些应用都要求所有用户看到的虚拟世界是一致的，并且用户之间、用户与虚拟空间中的实体之间都可以进行交互，同时隐藏分布性以提供给用户最大限度的真实感。

要建立高效的网络虚拟环境，就必须协调管理众多系统组件之间的交互过程，这些组件包括网络、处理器、图形显示设备、用户输入设备、数据库以及其他外部信息服务器。同时，开发人员必须保证虚拟系统具有良好的异构性、可扩展性和容错性，使其便于配置和部署。然而作为工程性的系统，网络虚拟环境无法同时满足以上全部的要求。因此在具体实现时，设计者必须为这些属性设定优先级，决定首要满足的要求是什么。

本书接下来的各个章节将就网络虚拟环境设计者所面临的各种问题进行详细介绍。当设计者为这些问题设计解决方案时，必须全方位地考察网络虚拟环境的性能。如果网络虚拟环境中某个性质发生了改变，那么其他各个方面的属性几乎肯定会受到影响，虽然造成的影响可能有大有小。因此，要有效地对网络虚拟环境进行开发，就必须清楚地了解系统各个组件之间的交互情况。

网络虚拟环境并不是一个全新的概念。关于这个方面的实验与研究已经持续了几十年。我们将从探究网络虚拟环境系统的发展历程开始，一步一步对网络虚拟环境的设计和开发做更为深入的研究和探讨。当然，只有理解了应用于现有系统中的设计策略，我们才能为将来的网络虚拟环境系统开发出更好的设计方案。

参考文献

[Brutzman+97] Brutzman, D., M. Zyda, K. Watsen, and M. Macedonia. Virtual reality transfer protocol (vrtp) design rationale. In *Proceedings of the IEEE Sixth Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises* (WET ICE '97), 179–186. IEEE Computer Society, Cambridge, MA, June 1997.

[Cruz-Neira+93] Cruz-Neira, C., D. J. Sandin, and T. A. DeFanti. Surround-screen projection-based virtual reality: The design and implementation of the CAVE. In *SIGGRAPH 1993 Conference Proceedings.* ACM SIGGRAPH, Anaheim, July 1993.

[Gibson85] Gibson, W. *Neuromancer.* New York: Ace Books, 1985.

[Hartman/Wernecke96] Hartman, J., and J. Wernecke. *The VRML 2.0 Handbook: Building Moving Worlds on the Web.* Reading, MA: Addison-Wesley, 1996.

[OpenGL97] OpenGL Architecture Review Board. *OpenGL Reference Manual: The Official Reference Document to OpenGL, Version 1.1*, C. Frazier and R. Kempf, eds. Reading, MA: Addison-Wesley, 1997.

[Stephenson92] Stephenson, N., *Snow Crash.* New York: Bantam Spectra, 1992.

[Sutherland68] Sutherland, I. E. A head-mounted three-dimensional display. In *AFIPS Conference Proceedings*, vol. 33, I:757–764, 1968.

[VRML-LivingWorlds] VRML Living Worlds Working Group Web site: *http://www.vrml.org/WorkingGroups/living-worlds/*

第 2 章　网络虚拟环境的起源

本章将全面回顾网络虚拟环境的历史，讨论在网络虚拟环境发展初期扮演重要角色的几个系统。这包括军事领域的两个开发已久的系统（SIMNET 和 DIS），网络游戏（SGI 公司的 Dogfight，id 公司的 Doom 等），以及应用于学术研究的系统（NPSNET，PARADISE，DIVE，BrickNet，MRToolkit 等）。

本章的目的不是对这些系统进行深入描述，而是旨在分析这些系统的关键特性，也就是网络软件架构（NSA）。网络软件架构指的是在网络虚拟环境系统中一些不可避免的问题，即在有限的网络带宽和处理器周期下，系统使用哪些网络协议以及使用什么软件架构来支持这些协议。使用这个术语是为了强调同时解决这两个问题的重要性，而不是仅仅注重于定义一个软件架构而忽视了可用的带宽，或者只关注定义使用的协议而忽视了处理器周期的限制。

2.1　美国国防部与网络虚拟环境

美国国防部是网络虚拟环境系统最大的开发机构，它研究的网络虚拟环境主要用于模拟系统。国防部在这个领域早期的贡献主要有两个方面：从历史的角度看，它是最早使用 SIMNET 系统[①] 开发网络虚拟环境的机构之一；从技术的角度来看，它是最早对大规模网络虚拟环境进行研究的机构。

2.1.1　SIMNET

SIMNET（simulator networking，仿真网络）是一个为国防部高级研究计划署开发的分布式军事虚拟环境，最早的开发者有 BBN 公司、Perceptronics 公司和 Delta Graphics 公司[Pope89]。SIMNET 从 1983 年开始开发，并于 1990 年 3 月底交付美国军方使用。SIMNET 的目标是开发一个低成本的网络虚拟环境，用于训练小的战斗单元（M1 坦克、AH-64 直升机、战地指挥所等）以团队形式作战。训练这些小的战斗单元不是完全依靠 SIMNET，SIMNET 的预期作用只占 70%，而另外的 30%需要士兵们在坦克中进行实战演习获得。

① 从 1984 年开始 Amaze 被认为是第一个现代的网络游戏／虚拟环境系统[Berglund/Cheriton85]。

SIMNET 的两个主要技术挑战是:(1)怎样制造高质量低成本的仿真器；(2)怎样把仿真器联网，从而形成一致的仿真战场[Miller/Thorpe95]。为了开展对这些技术挑战的研究， SIMNET建造了 11 个站点的试验台，每个站点有 50 到 100 个仿真器。SIMNET 可以从网络上的任何地方进入，用一个仿真器作为虚拟环境的入口。一旦进入了该虚拟的环境，这个参与者就可以与在虚拟战场上的其他参与者交互。在虚拟环境里的活动并不是预先设定好的，而是由参与者在遵循命令链的前提下实时决定。

SIMNET网络软件架构

SIMNET网络软件架构有三个基本组件：

1. 对象事件结构
2. 自主仿真节点概念
3. 一组嵌入式的称为‘航位推测’[Miller/Thorpe95]的预测建模算法的集合

对象事件结构是 SIMNET 网络软件架构中最简单的组件。它把世界建模成对象的集合，对象之间的交互构成了事件的集合。对象是可以通过网络进行交互的车辆和武器系统。一个单一的对象,比如一辆坦克,在 SIMNET 中通常由一个单一的机器管理。 SIMNET 中的事件是由网络传送的消息,指示环境或对象状态的改变。比如,某个对象(坦克)向右方转动是一个事件，把它作为消息发送出去，而另一个事件可能是坦克已经开火。

在 SIMNET 中，对象事件结构不同寻常的设计决策之一是:SIMNET 中基本地形及其结构与对象集合是分离的。也就是说，对于一个在 SIMNET 的地形中要被摧毁的桥梁，必须单独视为一个对象，并且它的状态要不间断地传递到网络上。

在 SIMNET 中,自主仿真节点的意思是:在网络上的各个参与者、车辆和武器系统负责把消息和数据包放到网络上，以便准确地报告它们当前所处的状态。除了发送状态更新信息和事件信息以外，这些自主节点并不与数据包接收器进行交互。数据包接收器负责接收这样的状态变化信息，并且根据更新消息适当改变它们自己（数据包接收器）的局部世界模型。不采用集中式服务器使得在系统中一个节点的故障不会造成整个仿真的停止。此外，这种无集中服务器的结构也允许参与者随时加入和离开仿真应用。

自主节点的概念是指每个节点要对虚拟环境中一个或多个对象负责。对象负责也就是节点必须将数据包放置在网络上，通知其他节点它所维护的对象的当前状态或者状态的任何改变。把状态变化信息放置在网络上意味着，每当一个节点的对象的变化大到足以需要其他参与者了解这种变化时，这个节点就需要把数据包放在网络上。把状态变化信息放置在网络上也意味着，每一个节点的对象必须提供周期性的心跳信息，通常是 5 秒钟一次,这是为了让其他的参与者知道一个具体的对象还活着，而且还在系统中，因此应该被显示。节点的自主性在 SIMNET 网络软件架构对象中还有一些有趣的现象，一些对象会因为心跳信息

的延迟而暂时消失或重新出现。经常改变状态的对象需要将大量数据包放置到网络上。

SIMNET 网络软件架构的第三个组件是一个定义良好的预测建模算法集合，称为“航位推测”算法。当 SIMNET 的原型首次被构建时，它的实现方式是针对对象状态的每一次改变都会立即有更新数据包发送到网络上。如果参与者数量很少，那么则不必太过担心，但对于一些对象而言，生成数据包的速率和仿真控制程序的速率几乎是一样的（更确切地说，就是帧频）。数据包以这样快的速度产生不但会使网络堵塞，也会使 CPU 负载过重。

为缓解上面提到的大量数据包充斥网络的问题，SIMNET 的研究人员创建了对象（object）和幻影（ghost）的样式。这种样式背后的想法是：每当对象的本地节点确定网上其他节点对该对象的预测状态与其实际状态之间的差别超出一定的阈值时，该对象才把更新数据包放在网上。当这种情形出现时，一个节点就必须把更新数据包放在网络上告知其他节点这个模拟对象当前的状态。这种样式假设系统中其他节点把该对象的幻影复制保存在内存中，并且最后记录的方向、速度和位置足以在一定的误差阈值内预测对象的当前状态。当“航位推测”接收到一个新的关于幻影对象的更新数据包时，就会以新接收到的运动参数重新开始运行。

当接收到关于幻影对象的新的数据包时，根据允许的误差阈值，幻影对象被视为做出了相应细微的挪动或退缩。较大的阈值意味着当对象和它的幻影不同步，推测结果需要被纠正时，我们可以在网络上少传一些数据包，但对象显示的时候会有很大的跳变。航位推测算法同样使得数据包丢失不再是很严重的问题，因为我们认为对象继续在以最后知道的方向和速度在移动。如果对象不是在做无序移动，这一招通常有效。如果对象在做混乱无序的移动，那么在丢失的数据包之后，就必须获得另一个数据包，并且对象不可以离开得太远。

心跳数据包同时提供更新的和额外的位置同步信息。SIMNET 中数据包的发送速率是这样的：在地面上慢速移动的车辆平均每秒发送一个数据包，而空中运输工具则是每秒传送三个数据包。SIMNET 车辆状态数据包提供了完整的位置、速度和方向信息，如表 2.1 所示。

可以看出，数据包中的一些信息是相对固定的。例如，在每个数据包中，总会重复发送一些关于标记和功能方面的信息。所有需要的位置、引擎速度、速度向量、坦克转角，以及射击仰角等车辆状态改变数据都在数据包里。其中的一个新颖之处是：只需一个位字段就可以表示车辆是否保持静止状态。这个字段允许关闭车辆幻影计算，使得接收者节点节省 CPU 周期。除了车辆显示的数据包以外，SIMNET 还给出其他的数据包定义作为它的应用层协议的一部分，这些事件数据包有射击（发射武器），间接射击（发射弹道武器），冲突（车辆与物体发生碰撞冲突）和击中（一个武器打中一个对象）。

一个典型 SIMNET 节点上的软件体系结构包括以下几个主要的模块［Miller/Thorpe95］，如图 2.1 所示，模块间的连线标识了数据的流向：

- 网络接口
- 其他车辆状态表

- 计算机图像生成软件
- 控制和显示接口
- 本地车辆动态信息
- 声音生成

表 2　SIMNET车辆显示包内容

域名	内容	域大小(字节)
车辆 ID	站点	6
	主机	
	车辆	
车辆类型	坦克型	1
	简单型	
	静止的	
	不相关的	
部队 ID		1
装束	对象类型:显著的	8
	对象类型:其他	
坐标系	位置:x,y,z	24
转动矩阵	—	36
显示	—	4
标记	文本域	12
时间戳	—	4
功能	—	32
引擎速度	—	2
静止位和填充位	—	2
车辆显示变量	速度向量	24
	坦克转角	
	射击仰角	

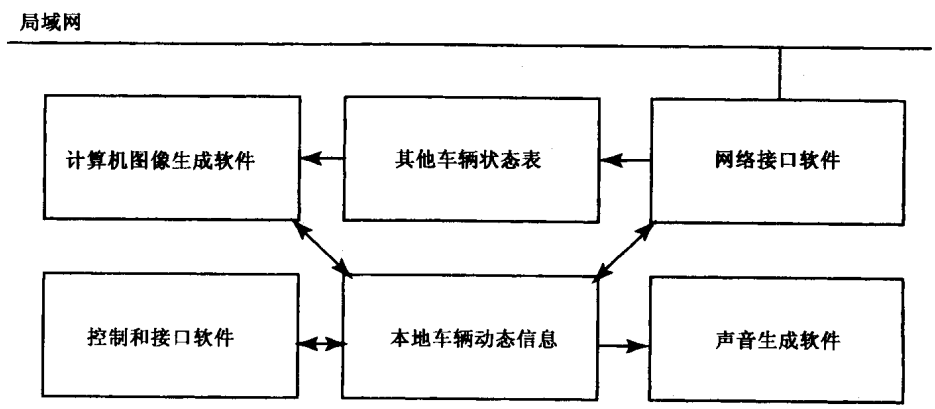

图 2.1　SIMNET主要的软件模块