计算机网络协议工程

龚正虎 编著

国防科技大学出版社

● 龚正虎 编著

计算机网络协议工程

● 国防科技大学出版社

[湘]新登字009号

内 容 简 介

协议工程是80年代才发展起来的一门非常活跃的新兴学科，它旨在减少协议开发中的错误，提高计算机网络和通讯系统的开发效率，促进标准化的发展。本书首先论述了协议工程的定义、研究内容和研究方法，然后系统地介绍了协议构造、协议模型、协议描述、协议验证、协议综合、协议实现和协议测试的理论和方法。

本书内容丰富，叙述深入浅出，有大量实例和思考题。它不仅可作计算机和数字通讯领域内高年级本科生和研究生的教材，也可作相关专业的科研教学人员的参考书。

计算机网络协议工程
编　著：龚正虎
责任编辑：曹莉华
*
国防科技大学出版社出版发行
（长沙市砚瓦池正街47号）
邮编：410073　电话：(0731)4436564
新华书店总店科技发行所经销
湖南大学印刷厂印装
*
开本：787×1092　1/16　印张：11　字数：254千
1993年12月第1版第2次印刷　印数1001—3000

ISBN 7—81024—275—X
TP·50　定价：12.00元

前　言

给定系统中多个实体相互作用的规则，以及它们与系统的外部环境相互作用的规则总称为协议。协议是计算机通讯、计算机网络、多机系统等分布系统的灵魂。

协议的开发过程包括协议设计与描述、协议验证与分析、协议实现、协议测试等几个主要阶段。分布系统的主要特征是分布性、并发性、异步性、实时性以及信道的不可靠性，这些特征意味着协议本身一定是复杂的。此外，随着计算机通讯与网络技术的发展，分布系统的功能越来越多，所提供的服务越来越丰富，不同系统的互通能力和互操作能力也越来越强，这进一步增强了协议的复杂性。协议复杂性不但意味着协议开发难度大、周期长、而且潜在错误多，而协议开发过程中任何一点错误和缺陷都将给分布系统的稳定性、可靠性、坚固性、安全性、容错性以及异种系统之间互通性和互操性带来巨大的危害。为此，一门新的计算机学科——协议工程(Protocol Engineering)便产生了。

一体化(integrated) 、形式化(formal) 的协议开发过程叫做协议工程，使协议开发一体化、形式化的理论和技术叫做协议工程技术或协议开发技术。协议工程旨在减少协议开发中潜在的错误，提高协议开发效率，促进协议标准化的发展。

协议工程是80年代才发展起来的一门非常活跃的新兴学科，许多理论问题和技术问题尚待人们进行广泛的、深入的研究。本教材编写期间，国内尚无完整教材可参考，国外教材也很少，这样，作者只能对大量的论文资料进行分类整理，综合抽取出协议工程的主要理论技术和方法，形成本教材的内容。作者的意图是，向计算机领域内的高年级本科生、研究生和科研教学工作者介绍这门新兴学科，起抛砖引玉作用。

作者利用访美机会拜访了马里兰大学的D.P.Sidhn教授，他的许多研究成果及其大量论文和讲义为本教材提供了重要参考。教材编写中，作者得到了国防科技大学六０六教研室许多同行的热情帮助，湖南大学王敬觉教授和国防科技大学卢锡城教授对本书的原稿进行了校阅，提出了许多宝贵的修改意见；王绿园教员不辞辛苦进行大量的文字处理，为本书的出版创造了条件，在此作者深表感谢！

龚正虎

1993年6月

目　录

第一章　协议工程概论

§1.1　协议及系统的相互作用

1.1.1 分层嵌套系统模型

计算机网络以及分布计算机系统可抽象成一个分层嵌套的系统模型。第 n 层的全局系统由多个分布的局部系统组成，而这些局部系统由一个或多个信道系统（通讯系统）耦合起来。n 层的各个局部系统以及信道系统本身又可看做（n－1）层的全局系统。只考虑高层信道映射成低层全局系统的分层嵌套的模型如图 1.1 所示。这里，S_1 为 n 层的全局系统，它包含两个局部系统和一个通道系统：S_{11}，S_{12} 和 S_2。S_2 映射成（n－1）层的全局系统，它由 S_{21}，S_{22} 和 S_{23} 三个局部系统和两个 S_3 通道系统组成。上层局部系统是上层全局系统的基本组成部份，它不映射成下层的全局系统。n 层的全局系统构成（n－1）层的外部环境。

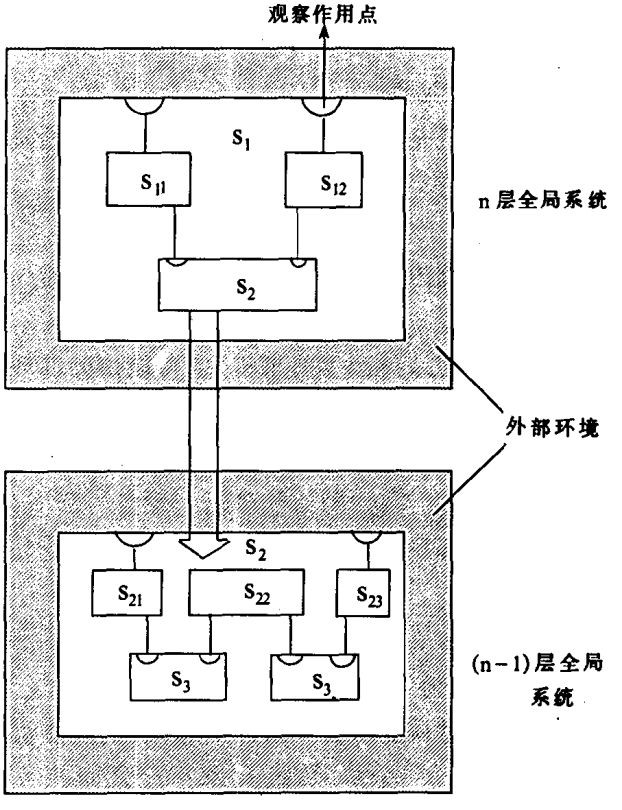

图 1.1　分层嵌套系统模型

- 局部系统

局部系统可以是：

计算机网络中的主机

计算机内的进程或任务

程序模块

计算机外部设备

计算机部件

OSI 模型中的实体，或协议机

- 通道系统

通道系统可以是：

通讯网络（广域网，局域网）

逻辑网络（电子邮政系统，分布文件系统）

共享存贮区

计算机总线

通讯介质

- 全局系统

由通道系统耦合起来的各个局部系统在协议支持下协同工作而形成全局系统，全局系统的行为是局部系统和通道系统全体行为的总和。

- 外部环境

n 层全局系统构成（n－1）层的外部环境，全局系统的行为可在观察作用点看到，外部

环境和全局系统的相互作用(实际是外部环境和某些局部系统的相互作用)通过观察作用点来实现,因此观察作用点是全局系统和外部环境的接口。

1.1.2 系统的相互作用

在n层全局系统内存在多种系统之间的相互作用(交互作用),它们是:

- 局部系统和通道系统之间相互作用

局部系统通过观察作用点和通道系统直接耦合,这些观察作用点归属通道系统,相互作用的手段是事件。

- 局部系统之间的相互作用

局部系统通过交换有确切定义的报文间接耦合,这些报文是通过通道系统传递的。

- 全局系统和外部环境之间的相互作用, 局部系统和外部环境之间的相互作用

某些局部系统通过观察作用点直接和外部环境发生作用,这种相互作用也是通过事件来完成的。

在n层内,系统相互作用按照确定规则进行,这些系统作用规则的总和就构成n层协议,当然n层协议包括三部份,分别对应于上述三种系统之间的相互作用。n层全局系统行为完全受n层协议制约。

1.1.3 事件与活动

局部系统是系统相互作用的主角,它的活动(action)由事件驱动。对局部系统来说,存在两类事件:

- 内部事件

局部系统内部产生的事件,例如时钟超时。

- 外部事件

局部系统外部产生的事件,它可能由外部环境引起的(如收到一个服务请求),可能由通道系统产生(如收到一个报文、通道系统故障等), 可能由其它原因 引起(如网络管理员的干预)。

除特殊事件(如网络管理员干预外),所有外部事件通过观察作用点来传递。每个事件将引起局部系统的一次活动,这些活动执行协议所规定的任务,因此,局部系统也称之为协议机。

1.1.4 协议及协议的描述

基于上述系统相互作用的概念,我们给协议下如下定义。

定义: 对于n层全局系统,局部系统之间相互作用的规则以及它们与外部环境和通道系统相互作用的规则的总和就是n层协议。

协议必须以某种方式(文字的、图形的)描述,协议的描述至少应包括以下内容:

1. 局部系统之间交换的报文的确切定义。
2. 各个局部系统在一个事件产生时进行什么样活动。
3. 各个局部系统怎样通过观察作用点使用通道系统提供的服务。

4. 各个局部系统怎样通过观察作用点向外部环境提供服务。

除第 1 条的内容之外，对各个局部系统来说，其它内容可能不相同。极端情况下，如果 n 层内有 m 个功能各不相同的局部系统，那么就必须给出 m 个协议实体描述文本。一般情况下，所有局部系统的功能是相同的，在主从工作方式下，局部系统也只分为两类(主系统，从系统)。

1.1.5 OSI 模型

国际标准化组织(ISO)定义的开放系统互连参考模型(OSI－RM)是一个被人们广泛接受的计算机模型[1]，该模型简称 OSI 模型。目前，ISO 所制定颁布的协议均以此模型为基础，这些协议统称为 OSI 协议。OSI 模型在功能上将计算机网络分成七层：

- 应用层，Application layer，简称 A 层
- 表示层，Presentation layer，简称 P 层
- 会晤层，Session layer，简称 S 层
- 传输层，Transport layer，简称 T 层
- 网络层，Network layer，简称 N 层
- 数据链路层，Data Link layer，简称 DL 层
- 物理层，Physical Layer，简称 PL 层

OSI 模型是一个分层嵌套的系统模型，相邻层之间关系完全可以用图 1.1 形式表示。图 1.2 描述 T 层、N 层之间关系，它是图 1.1 的一个实例。图中，TPM 为传输层协议机，NPM 为网络层协议机，DL_1，DL_2，DL_3 为三个数据链路层信道。N 层内有四个协议机，中间两个不直接向 T 层提供服务。TSAP 和 NSAP 就是观察作用点。根据 OSI 模型，我们给出如下一些常用名词术语的解释。

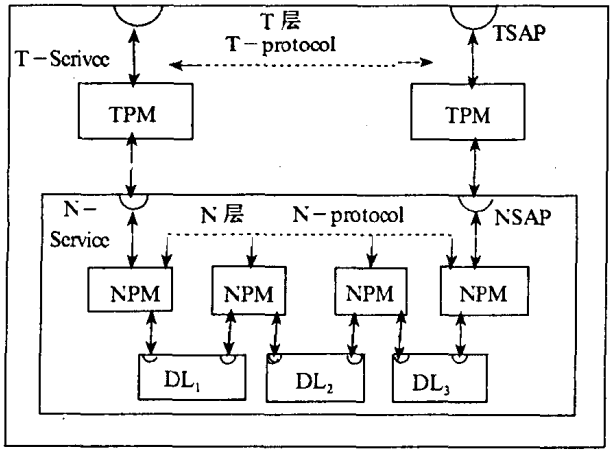

图 1.2 OSI 模型的嵌套结构

- 协议实体(protocol entity)

协议实体又简称实体，一个实体就是一个局部系统。n 层实体记作(n)－entity。OSI 模型各层实体分别记作 A－entity，P－entity，S－entity，T－entity，N－entity，DL－entity，PL－entity.

- 协议机(protocol machine)

协议机是协议实体的代名词，缩写成 PM. n层协议机记作(n)PM，OSI 模型各层协议机记作 APM，PPM，SPM，TPM，NPM，DLPM，PLPM.

- 服务(service)

n 层服务是 n 层全局系统行为的一种体现。n 层服务向外部环境提供服务，外部环境使用 n 层服务，并可通过 n 层服务观察 n 层全局系统的行为。n 层服务记作(n)－service，OSI 模型各层服务分别记作 A－service，P－service，S－service，T－service，N－service，

PL－service.

• 服务提供者(service provider)

n层服务的提供者就是n层的局部系统(协议机)。

• 服务使用者(service user)

n层服务的使用者是(n+1)层的局部系统,但不一定是协议机,例如,应用层服务的使用者不一定是协议机。n层服务的使用者简称为n层用户。

• 服务访问点(SAP)

服务访问点(Service Access Point)是服务使用者和服务提供者的界面(观察作用点)。n层服务访问点记作(n)SAP,OSI模型各层的服务访问点分别记作ASAP,PSAP,SSAP, TSAP, NSAP, DLSAP, PLSAP.

• 服务原语(service primitive)

服务原语是服务使用者和服务提供者相互作用的原子行动的描述,所谓原子行动是不可部分执行的行动,要么完全执行,要么不执行。服务原语描述服务提供者和服务使用者一次原子交互作用的名称以及各参数的含义。一条服务原语的执行在服务访问点引起一个事件,这个事件也是一种原子事件(atomic event)。

• 服务规范(Service specification)

服务规范确切定义了服务使用者和服务提供者之间相互作用的规则(例如,服务原语的执行序列等)。它仅包括1.1.2节所描述的局部系统和外部环境之间的相互作用规则。

• 地址(address)

地址就是服务访问点标识(SAP identifier),n层地址标记为(n)－address,OSI模型各层的地址分别记作A－address, P－address, S－address, T－address, N－address, DL－address, PL－address.

• 协议(protocol)

协议是一组n层实体在执行n层功能中相互通讯行为的规则和格式(语法和语义)。它包括1.1.2节中所描述的局部系统之间相互作用以及局部系统和通道系统之间相互作用的规则。n层协议记作(n)－protocol,OSI各层协议分别记作A－protocol,P－protocol, S－protocol, T－protocol, N－protocol, DL－protocol, PL－protocol.

• 协议规范(protocol specification)

协议规范用某种语言确切定义了实体之间通讯规则和交换的报文的格式,以及实体利用低层服务的规则。

• 协议数据单元(PDU)

协议数据单元(Protocol Data Unit)是实体之间交换的报文。n层协议数据单元记作(n)PDU,OSI模型各层的PDU分别记作APDU, PPDU, SPDU, TPDU, NPDU, DLPDU, PLPDU.

• 服务数据单元(SDU)

服务数据单元(Service Data Unit)是服务提供者和服务使用者之间传递的数据单元。n层服务数据单元记作(n)SDU,OSI各层的SDU分别记作ASDU, PSDU, SSDU, TSDU, NSDU, DLSDU, PLSDU.

• 协议控制信息(PCI)

协议控制信息(Protocol Control Information)是实体本身产生的控制信息，它是PDU中一部份。n层协议控制信息记作(n)PCI,OSI各层PSI分别记作 APCI, PPCI, SPCI, TPCI, NPCI, DLPCI, PLPCI.

说明：*1.(n−1)SDU=(n)PDU=(n)SDU+(n)PCI(报文头+报文体)*

图1.3示出PDU,SDU和PCI之间关系。

2.本教程将广泛使用OSI术语。

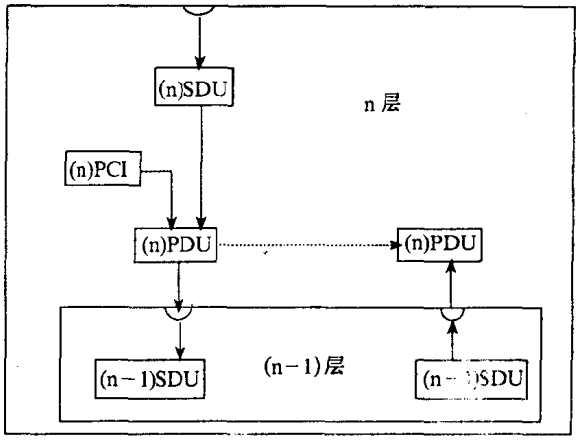

图1.3　SDU,PDU和PCI

§1.2 协议开发过程

协议的开发包括六个过程：

• 协议设计，PDU格式，协议机制，服务原语等设计；
• 协议描述，用某种语言确切地描述协议元素；
• 协议验证与性能分析，对所描述的协议验证其正确性，分析其性能；
• 协议实现，根据描述的协议产生网络硬软件；
• 协议测试，对实现的协议进行测试；
• 协议维护，对网络硬软件进行维护；

n层全局系统是一个分布系统，它的行为完全由n层协议(包括OSI定义的服务规范和协议规范)所支配，分布系统的复杂性就是协议的复杂性。复杂性体现在：

• 分布性　多个局部系统分布在不同机器上；
• 迸发性　各个局部系统迸发工作；
• 异步性　各个局部系统不能同时观察到事件的产生；
• 实时性　局部系统的活动有严格的时序要求和时间要求；
• 通道系统的不稳定性　通道系统本身可能不稳定，产生错误。

随着计算机网络和分布计算机系统的发展，协议的复杂性越来越高，这就给协议的开发带来许多困难。这表现在：

• 网络软件规模大，开发周期长。

一个功能较完善的计算机网络所包括的各层协议，少则十多个，多则几十个，实现后的程序代码有几十万行，甚至几百万行之多。一个功能较完善的高效可靠的计算机网络软硬件系统的开发周期为5～10年，甚至更长。

• 潜在错误多，排除困难。

由于分布系统的特点，网络软件的调试比较困难，潜在错误多，随机性大，排除困难。

• 协议标准化难于保障

随着计算机网络的发展，不同厂商的网络必须能互联，不同厂商的机器必须能互联。协议的标准化不但要求各厂商使用相同的协议，而且要求实现后的软硬件能协同工作。目前，协议标准化还未得到完全保障。

- 软件移植性差

一种机器上开发的网络软件移植到另外一种机器时，可移植性差。一般情况是，高层协议的可移植性大一些，低层协议的可移植性小一些。

- 软件可维性低

网络软件的更新升版时，软件变化较大，更新升版周期长。

由于上述原因，协议的开发过程急需工程化，以便提高网络软件的生产率，促进标准化的实现，提高网络软件的可靠性和可维护性。

§1.3 协议工程的研究内容

1.3.1 协议工程的定义

一体化的、形式化的协议开发过程叫做协议工程。协议开发中所需要的各种开发、管理、维护工具，以及协议的不同表示工具构成协议工程环境。使协议开发一体化、形式化的理论和技术，以及协议工程系统建造技术统称为协议工程技术或协议开发技术（简称协议技术）。

所谓“一体化”的含义是：协议的设计、验证、实现和测试，在技术上前后衔接，并在同一个开发系统中完成，一体化（integrated）即系统化。以往的协议开发没有做到一体化，技术上前后阶段不衔接。协议设计者凭自己的经验和智慧设计协议，用自然语言描述出来，经过他人审定或模拟之后，就予以公布。另外的人只在感兴趣时才采用某种方法和理论对协议进行论证。有些协议公布后至今也无人对它进行验证。协议验证者往往也没有将验证结果直接反馈给协议设计者。协议实现者往往根据自己的环境和要求修改协议，协议实现之后，他们往往只要求自成系统，不考虑不同协议实现者的产物的互联问题，不考虑自己实现后的协议是否和颁布的协议一致，也不将协议实现中的问题反馈给协议实现者。协议的测试由实现者进行，实际上，这种测试只是一种程序动态调试手段。

所谓“形式化”，它的含义是：用形式描述语言 FDL(Formal Description Languages) 连接协议开发的各个阶段。一种 FDL 都有一种或多种形式描述技术作基础，有严格的语法和语义定义，它抽象于具体的实现，可符号执行，可转换翻译成程序设计语言。协议的形式描述是 FDL 发展的最主要动因。当协议用一种 FDL 描述之后，协议的自动化验证、自动化实现、自动化测试便可在协议工程系统中进行了。

协议工程系统包括许多软件工具，包括许多协议的不同表示（见图 1.4）。在协议工程系统中，协议的表示形式有：

- 非形式描述文本 (Informal Specification)

用自然语言和图表表述的协议，易读易懂，但不严密，有多义性。

- 形式描述文本 (Formal Specification)

用 FDL 描述的协议，严密，无二义性，可符号执行，可转换成程序设计语言程序。

- 与机器无关的源程序代码(Machine－independent Source Code)

这是由形式描述文本翻译过来的程序设计语言(Pascal,C 等)程序。协议本身有一定抽象性，即协议没有指明这个协议在某个机器上怎样实现，正因为协议本身是抽象的，它才适合用 FDL 来描述。这样，形式描述文本翻译后的程序设计语言程序就是与机器无关的代码了。

- 实现代码(Implementation Code)

这是协议实现后的最终代码。一般与机器无关的源程序代码只占最终实现代码的一部份(50% 左右)。协议在一种机器上的实现还包括大量协议文本没有描述的程序，例如缓冲器分配管理，系统输入/输出操作等，这部份程序称作与机器相关代码。

- 测试套具(Test Suite)

这是一组关于协议测试步骤和测试数据的文件，它由协议的形式描述文本产生。测试套具是用另外一种语言描述的(形式描述语言或非形式描述语言)。

协议工程系统包括多个子系统，每个子系统由多个软件工具构成。这些子系统包括：

- 转换(Transformation)系统

  该系统将协议的非形式描述文本转换成形式描述文本。该系统要求高智能化的软件去识别和读懂非形式描述文本，因此该系统实际是一个人工系统。

- 验证(Verification, Validation)系统

  对协议进行验证检查，发现并修改潜在的错误。

- 性能分析(Performance Analysis)系统

  对协议性能进行分析评价，修改协议，提高性能。

- 翻译(Translation)系统

  将形式描述文本翻译成源程序代码。

- 实现(Implementation)系统

  它实际是具体操作系统所提供的程序开发环境(编辑、编译、连接、调试、排错等)。实现代码在这个系统中产生。

- 测试(Test)系统

  测试套具在该系统中执行，对实现代码进行测试，分析结果，检测错误，将错误反馈给实现系统，修改实现代码。在某些情况，错误信息还必须反馈给协议实现者。

- 测试套具生成(Test Suite Generation)系统

  根据协议的形式描述文本产生测试程序(描述怎样测试)和测试数据，该系统的部份工作可由人工完成。

图 1.4 所示的是一个完整的协议工程系统，随着协议工程技术的发展，系统中的软件工具将越来越完善，越来越方便，自动化程度也将逐步提高。

协议工程技术可归结为五个方面：

- 协议设计(构造、综合、开发)技术
- 协议模型及形式描述技术

- 协议验证分析技术
- 协议实现技术
- 协议测试技术

本书不去讨论协议工程系统中各个软件工具的细节，而是按上述五个方面去讨论协议工程技术。为此，下面将简述各个方面的发展情况。协议转换技术也应该包括在协议工程技术之中，但本书不加讨论。

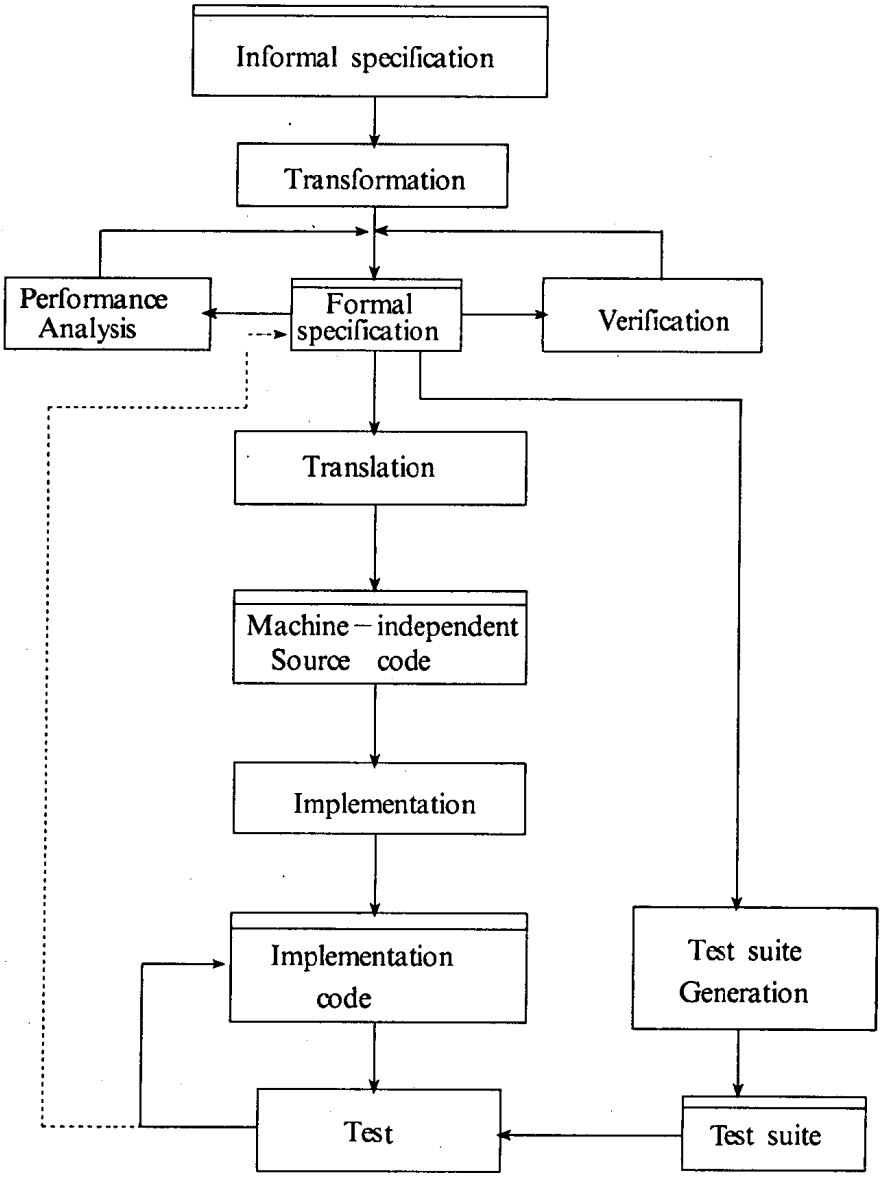

图 1.4 协议工程系统

1.3.2 协议设计技术

"协议设计"这个名词有不同含义，我们可以说，整个协议开发过程就是协议设计过程，就是说，一个协议只有经过实现、运行、测试，证明是可用的、正确的，才算完成了设计任务(图 1.4 的全部过程)；我们也可以说协议经过严格验证和性能分析之后就已经完成了设计任务(图 1.4 的前半部过程)；我们还可以说，非形式描述文本提出后(经过非形式的验证分析)就已完成了设计任务。无论是哪种含义的协议设计，都涉及两个共同的技术问题：环境分析方法和协议设计方法。

一个分布系统按功能划分可抽象成一个分层嵌套的系统模型(图 1.1 和图 1.2)。n 层全局系统就是 n 层协议的设计环境，而 n 层环境的分析包括四个内容：

- 从总体结构上看，n 层实体实现哪些功能？总体结构对它提出什么样要求？
- 它向(n+1)层实体提供什么样服务，或者说，(n+1)层实体向它提出什么要求？
- 对 n 层来说，(n−1)层的全局系统映射成通道系统，那么这个通道系统有什么特性？
- n 层包括多少实体，它们采取什么工作模式？

环境分析是设计的基础，好的设计方法是提高协议设计质量和效率的保证。虽然程序设计方法的一些概念(如结构化，模块化)可延伸到协议设计，但是协议设计有许多独特方法，如生成规则(production rule)等。

协议是一种宝贵的知识，协议设计者凭自己的智慧和经验已设计了许多高性能的协议，这些协议的积累为设计更复杂、性能更高的协议提供了基础。从这个角度来说，人工智能技术(专家系统)在协议设计中将有很好的应用前景。

协议设计技术一直未得到人们充分重视，参加研究的人员较少，主要原因是：协议设计者不能得到有力的协议设计(综合)工具，另外，一般的分布系统研究人员往往采用流行的成熟协议而不设计自己的协议，这样协议设计者就变成少数标准制定者了。这情况和程序设计大不一样，程序设计方法为众多程序设计者采用，因而得到广泛重视。

然而，无论是协议设计者还是协议使用者，学习和研究协议设计技术仍然是非常必要的，本书第二章和第六章将继续讨论这个问题。

1.3.3 协议模型及形式描述技术

形式描述技术涉及两个研究课题：第一，用什么样数学模型或逻辑模型表示 n 层局部系统，通道系统，全局系统以及它们之间的相互作用，以便获得抽象的协议模型；第二，设计基于一种或多种数学逻辑模型的形式描述语言(FDL)。

1.2 节指出，n 层全局系统的主要特点是分布性、迸发性、异步性、实时性、以及通道的不稳定性，那么用来表示 n 层全局系统的模型必须充分反映这些特性。目前，人们已将有限状态机 FSM(Finite State Machine)、Petri 网 PN(PetriNet)、时态逻辑 TL(Temporal Logic)、通讯系统演算 CCS(Calculas of Communication System)、形式文法 FG(Formal Grammar)、过程语言 PL(Procedural Language)等数学模型和逻辑模型用来表示协议模