HZ BOOKS
华章科技

从构架模式、编程示例和源代码三个维度系统深入解读ACE的架构设计和实现原理，帮助提升ACE开发人员的水平，及其对ACE定制和扩展的能力

深刻揭示软件框架设计的普适原则和设计哲学，为框架和平台软件设计者提供绝佳指导。

包含对10余种设计模式的解析，以及大量网络编程、动态库编程、异步编程的方法和技巧。

揭秘 系列丛书 UNLEASH

ACE Internals

ACE技术内幕

深入解析ACE架构设计与实现原理

潘荣 著

机械工业出版社
China Machine Press

揭秘 系列丛书 UNLEASH

ACE Internals

ACE技术内幕

深入解析ACE架构设计与实现原理

潘荣 著

机械工业出版社
China Machine Press

本书从构架模式、编程示例和源代码3个维度系统地对经典网络框架ACE（Adaptive Communication Environment）的架构设计和实现原理进行了深入分析，它能解决4个方面的问题：第一，帮助框架设计者领略软件框架设计的普适原则和思想，进而设计出自己的软件框架；第二，帮助ACE应用开发人员加深对ACE框架的理解，提升开发水平，更好地去定制和扩展ACE框架，以及解决C++网络通信中的难题；第三，帮助C++开发人员加深C++语言功底，书中有大量对C++源代码的分析，包括网络编程、动态库编程和异步编程等，还涉及10余个经典的设计模式的解析；第四，增强平台开发人员和软件架构师的技术修养，ACE的设计和实现都极其优秀，它的实现源码和架构思想非常值得去学习和研究。

全书一共7章，详细分析了ACE的Reactor、Service Configurator、Task、Acceptor_Connector、Proactor和Streams等6个框架的架构设计与实现原理。每个框架的分析分为3部分：第一，框架的设计分析，每个框架（除Task框架）都有一个构架模式与之对应，构架模式阐述了框架的设计原理，给出了框架的总体结构，是学习框架的理论基础；第二，框架的应用分析，每个框架都有一个应用实例与之对应，应用实例既帮助读者了解框架的使用方法，又为读者提供了一个可以调试的应用程序，便于读者使用调试器探索框架的内部秘密；第三，框架的实现分析，这是本书的重点，对框架的实现原理进行了详细的分析，并且对重点的类和流程给出了UML类图和UML顺序图，可以让读者在短时间内掌握框架的实现技术。

图书在版编目（CIP）数据

ACE技术内幕：深入解析ACE架构设计与实现原理/潘荣著. —北京：机械工业出版社，2012.6

ISBN 978-7-111-38824-1

Ⅰ.A… Ⅱ.潘… Ⅲ.C语言-程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字（2012）第129028号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）
责任编辑：孙海亮
藁城市京瑞印刷有限公司印刷
2012年7月第1版第1次印刷
186mm×240mm · 21.75印张
标准书号：ISBN 978-7-111-38824-1
定价：69.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991，88361066
购书热线：（010）68326294，88379649，68995259
投稿热线：（010）88379604
读者信箱：hzjsj@hzbook.com

前　言

为什么写作本书

软件框架的设计始终代表着业界最高的设计水准，《设计模式》一书指出：如果说应用程序难以设计，那么工具箱就更难了，而框架则是最难的。

尽管业界有一些大型的、常用的软件框架，如OMG的CORBA框架、SUN的EJB框架和微软的DCOM框架，但是很多一线的开发人员在日常的开发活动中并不和它们打交道，也很少接触其他软件框架。在这种情况下，我们如何来学习软件框架的设计思想，如何让这些思想来帮助我们设计和架构自己的软件框架呢？

ACE（Adaptive Communication Environment）是一个开源的、面向对象的网络框架，它实现了很多用于并发通信软件的核心模式，是一个非常好的软件框架学习平台。学习ACE所需要的特殊的专业知识非常少，读者只要熟悉C++和网络编程即可。有了本书可以让你的框架学习事半功倍。

虽然本书还无法和参考资料中大师们的著作相比，但是其对你学习软件框架的架构设计与实现原理同样会有非常大的帮助。侯捷大师说过这样一句话："源码之前，了无秘密。"本书通过框架代码、框架使用示例、UML类图、UML顺序图详细分析了ACE的Reactor、Service Configurator、Task、Acceptor_Connector、Proactor和Streams等6个框架。这些图例不仅可以帮助你从源代码的角度学习框架的架构设计与实现原理，实现与软件框架的一次亲密接触，还可以真正减少你学习所需的时间和精力。

书中每一个框架的架构原理均来自《Pattern-Oriented Software Architecture—A System of Patterns》和《Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects, Volume 2》，想深入学习架构原理的读者可以在阅读时参考以上两本书。

读者对象

本书主要适合以下读者：

- 希望深入理解软件框架设计和实现的读者。本书从构架模式和源代码两个方面，分析了ACE的6个框架的设计和实现原理。这些内容非常适合这部分读者学习和掌握软件

框架的设计思想，进而设计和实现自己的软件框架。

- ACE应用开发人员。对这部分读者来说，本书的内容有助于加深对ACE网络框架的理解，提高自己的开发水平。本书可以作为他们定制和扩展ACE网络框架的宝贵参考资料。
- C/C++编程人员。书中有大量对C++源代码的分析，包括网络编程、动态库编程和异步编程，还涉及十多个Gof经典设计模式，是C/C++编程人员非常好的进阶书籍。
- 平台开发人员和架构师。ACE的设计和实现都极其优秀，它的实现源码和架构思想非常值得这类读者去学习和研究。

如何阅读本书

学习本书应当具备的基础知识如下（ACE是一个跨平台的网络框架，这里不区分Windows、Linux、UNIX等各种不同平台）。

- 熟悉Socket编程，至少了解一种平台的常用的Socket API。
- 熟悉多线程编程，至少了解一种平台常用的多线程编程接口。
- 可选知识有：了解一种平台的动态库知识（用于Service Configurator框架）和异步编程知识（用于Proactor框架）。如果需要这些知识也可以在学习本书的过程中查阅相关资料。

本书共分7章，其中第1章介绍了ACE的Socket封装器，其余6章详细分析了ACE的Reactor、Service Configurator、Task、Acceptor_Connector、Proactor和Streams共6个框架。每个框架的分析又分为3部分：

第一部分是框架的设计分析。每一个框架（除Task框架）都有一个构架模式与之对应，构架模式阐述了框架的设计原理，给出了框架的总体结构，是学习框架的理论基础。

第二部分是框架的应用分析。每个框架都有一个应用实例与之对应，应用实例既帮助读者了解框架的实际使用方法，又为读者提供了一个可以调试的应用程序，便于读者使用调试器探索框架的内部秘密。

第三部分是框架的实现分析，也是本书的重点。本书对框架的实现进行了详细分析，并且对重点的类和流程给出了UML类图和UML顺序图，可以让读者在短时间内掌握框架的实现技术。

ACE的6个框架在实现上是相互独立的，在使用时又可以相互调用。如果读者是第一次接触ACE的框架，建议从本书的第1章开始学习。如果读者已经学习了ACE的部分框架，那么可以根据需要选择相关的章节补充学习。

勘误和支持

由于本人水平有限，书中难免有错误之处，恳请您在阅读时给予批评和指正。您可以访问我的博客http://blog.csdn.net/panrong_nust和我交流，也可以直接将您的宝贵意见发送到

panrong.nust@gmail.com。我将及时修改书中的错误并在我的博客中发布，同时也会尽最大努力为您提供支持和帮助，期待能够得到您的反馈。

本书的源代码来自ACE官方网站：http://www.cse.wustl.edu/~schmidt/ACE.html，选择的版本是ACE-6.0.0。

本书中的示例代码请登录华章网站（http://www.hzbook.com）进行下载。

致谢

感谢ACE的开发和设计者们，是你们奉献了一款优秀的开源软件框架！

感谢机械工业出版社华章公司的策划编辑杨福川老师，有了你的热心帮助才使本书得以出版！感谢姜影老师孜孜不倦地帮我修改稿件！

感谢我的父母，感谢你们将我培养成人。感谢我的小妹和妹夫。

感谢我的爱人，谢谢你的支持和鼓励。

感谢潘天骄和潘锦程两位小朋友，祝你们童年快乐！

感谢我的老师、朋友，谢谢你们的关心和帮助！

谨以此书献给加班在一线的最普通的程序员们，祝福你们！

潘 荣

目　录

第1章 概 述

本章先介绍模式与框架的基本概念及它们之间的关系。然后介绍ACE框架的基本组成及本书的内容组织形式和约定。最后通过一个示例分析ACE的Socket封装器（Socket Wrapper Facade）。Socket编程技术是整个网络编程的基础，而ACE的Socket封装器则是ACE对Socket编程接口面向对象的封装，它为上层提供了统一的、面向对象的接口，这是我们学习ACE网络框架的基础。

1.1 模式与框架

1.1.1 模式

自Gof（《设计模式》一书的作者四人组）对设计模式进行归纳、整理和分类，并出版了《设计模式》一书后，设计模式在软件开发的各个领域都大放光彩。在此基础上，各路“英雄豪杰”又对设计模式进行了更深入的研究，一时间“百花齐放、百家争鸣”。软件的分析与设计在面向过程、面向对象的方法的基础之上又发展出了面向模式的分析与设计方法。更有专家预言未来的软件开发是面向组件、面向框架的。面向组件、面向框架的软件开发可以带来更高的稳定性与更好的重用性。

在《设计模式》一书中，作者引用了模式的经典定义：每一个模式都描述了一个在我们周围不断重复发生的问题，以及该问题的解决方案的核心，这样就能一次又一次地使用该方案而不必做重复努力。

在《Pattern-Oriented Software Architecture—A System of Patterns》一书中，Buschmann等人将软件模式分为3类。

- 构架模式：笔者认为，可行的软件架构必然遵循一定的构架原则。这些构架原则被称为构架模式，它描述了软件系统最基本的组织结构。构架模式提供一组预定义的子系统，并描述了这些子系统的职责，以及它们之间内部关系的组织原则。
- 设计模式：构架模式中描述的子系统及它们之间的关系，通常由一些更小的构架单元组成。这些小的构架单元用设计模式来描述。这类模式等价于《设计模式》中的设计模式，主要用于描述和设计系统中的子系统、模块或组件，以及它们之间的关系。
- 惯用法：描述了如何利用给定语言的特点来实现子系统、模块或组件中的特殊方面，以及它们之间的关系。惯用法是一种针对编程语言的底层模式。

在《Analysis Patterns: Reusable Object Models》一书中，作者提出了分析模式是反映商

业过程的概念模式，而不是具体的软件实现。这些分析模式关注的是人们对业务的思考和认识，而不是计算机系统的设计方法。

通过上述分析可以看到，模式贯穿了软件开发流程的几个关键阶段。

- ❑ 系统分析——分析模式：着重于如何关注需求表面的问题所蕴含的实质。
- ❑ 系统设计——构架模式：规定了一个系统的结构特征及子系统的体系结构。
- ❑ 模块设计实现——设计模式：描述了通信组件的一种通用的、可重用的结构，用于解决特定环境下的一般设计问题。
- ❑ 实现——惯用法：特定编程语言的底层模式。

1.1.2 框架

用框架来翻译framework是非常贴切和生动的。框架在汉语词典中的含义为：①建筑工程中，由梁、柱等联结而成的结构；②比喻事物的组织、结构。显然含义②是一种抽象的定义。我们从含义①出发，“由梁、柱等联结而成的结构”，说明框架还不是一个完整的建筑，或者说框架以外的部分是可以改动的，还可以对很多地方做进一步设计。举个例子：如果门、窗的位置还没有确定，那么可以在框架中设计门和窗的位置；如果门和窗的位置已经在框架中固定，那么可以设计门和窗的款式。总之还有很多可以发挥的地方。至于发挥空间大小由具体的框架完成的粒度的粗细决定。粒度粗一点，发挥空间就大一点，要做的工作也会多一点；粒度细一点，发挥空间就小一点，可以有更多时间用于其他方面的设计。不管怎样，已经完成了的梁和柱等主体结构不能再改变，除非重新造一个框架（但这与使用框架的原意相悖）。同时设计必须符合框架已有的约束，不能把梁锯了开个窗，也不能把柱推了开个门。如果这样做，框架就会失去稳定性，设计也就变得不靠谱。

软件框架就是软件工程中的框架。一般而言，一个软件框架总是面向软件应用的某一领域，它用某种编程语言为该领域的使用者搭建好“梁”和“柱”。而使用者在框架的约束内，通过框架提供的接口，完成一定的应用功能。框架提供了很好的重用性，使开发人员不必每次都从头开始。同时，框架通过良好的设计屏蔽了该领域内最烦琐的细节部分，使不熟悉该领域的开发者也能对该领域进行开发。框架也为开发者提供了一个稳定的基础环境，加快了开发进程。当前最为常用的软件框架主要有OMG的CORBA框架、SUN的EJB框架和微软的DCOM框架等。

业界对框架的定义非常多，但只有放到特定语义环境中，才能准确表达框架的含义。在《设计模式》一书中，作者引用了这样的框架定义：框架是构成一类特定软件可复用设计的一组相互协作的类，该定义略显抽象。在本书中，引用Craig Larman在《Applying UML and Patterns》一书中的定义，因为它和本书的主题最为贴近。在该书中，Craig Larman是这样定义软件框架的：

- ❑ 框架是一个内聚的接口和类的集合，这些接口和类相互协助，提供核心的服务功能和逻辑子系统中不变的部分。
- ❑ 框架包含了抽象的类，这些抽象类定义了框架中用户需要遵循的约束，包含了对象的交互过程和不变量。

- 通常框架的使用者需要定义框架中部分类的子类，才能定制、利用、扩展框架的功能。
- 框架包含有抽象方法和具体方法的抽象类。
- 框架采用Hollywood原则“不要找我们，我们会打电话给你”。通常这意味着用户通过定义框架中既有类的子类，从框架中获取消息。

在本书中，我们把上述定义称为框架五元素，分别用元素1、元素2、元素3、元素4、元素5表示。元素1相对比较抽象，告诉我们，不管应用程序如何变化，框架是不变的部分。另外4个元素是我们的分析重点。

如果你是第一次接触软件框架，可能不是很明白Craig Larman对软件框架的定义。不过没有关系，学习完本书，你就会明白框架和这5个元素之间的关系，甚至还可以用这5个元素来指导软件框架设计。

1.1.3 模式与框架的关系

模式是在软件开发过程中形成的、已经被证明的、能解决某类问题的方法。框架是一个针对某个应用领域的软件半成品。

构架模式为框架的设计提供了设计原则，同时框架的实现又丰富了构架模式。一个新的软件框架的设计和实现可以采用或改善已有的构架模式，也可以根据应用领域的特征设计新的构架模式。前者是对既有构架模式的应用，后者是对构架模式的创新。

设计模式可以为框架带来更高层次的设计复用和代码复用。软件框架的实现通常同时使用了多种设计模式，设计模式有助于获得无须重新设计就可以适用于多种应用的框架体系结构。反之，从框架的设计与实现过程中也可以总结出新的设计模式，经过整理、归纳后可以丰富既有的设计模式。

1.2 ACE框架

ACE（Adaptive Communication Environment）是一个跨平台的用于并发通信的C++框架。ACE提供了丰富的可重用的C++封装器（Wrapper Facade）和框架组件。使用ACE，开发者可以开发出高性能、实时的通信服务和应用。ACE利用进程间通信、事件分离、动态链接和并发技术简化了面向对象的网络开发。ACE的总体结构如图1-1所示（来自ACE官方网站）。

ACE的结构可以分为以下几个部分（参考图1-1）。

（1）ACE操作系统适配层和C++封装器

ACE是一个跨平台的通信框架，支持Windows、UNIX、Linux以及VxWorks等常用操作系统。操作系统适配层直接驻留在操作系统接口之上，主要用于屏蔽各类操作系统的差异，增强ACE的可移植性和可维护性。C++封装器为上层提供统一的面向对象的接口，这些类型安全的C++接口可以大大简化应用程序的开发。这一层主要采用Facade设计模式封装各主要操作系统的常用功能。

- 并发和同步——多进程、多线程和同步功能。
- 进程间通信和共享内存——本机及远程间通信和共享内存功能。

❑ 事件分离——同步和异步I/O、定时器、信号、同步事件的分离功能。
❑ 动态链接——使用动态链接来提供应用程序的配置。
❑ 文件系统——文件和目录服务。

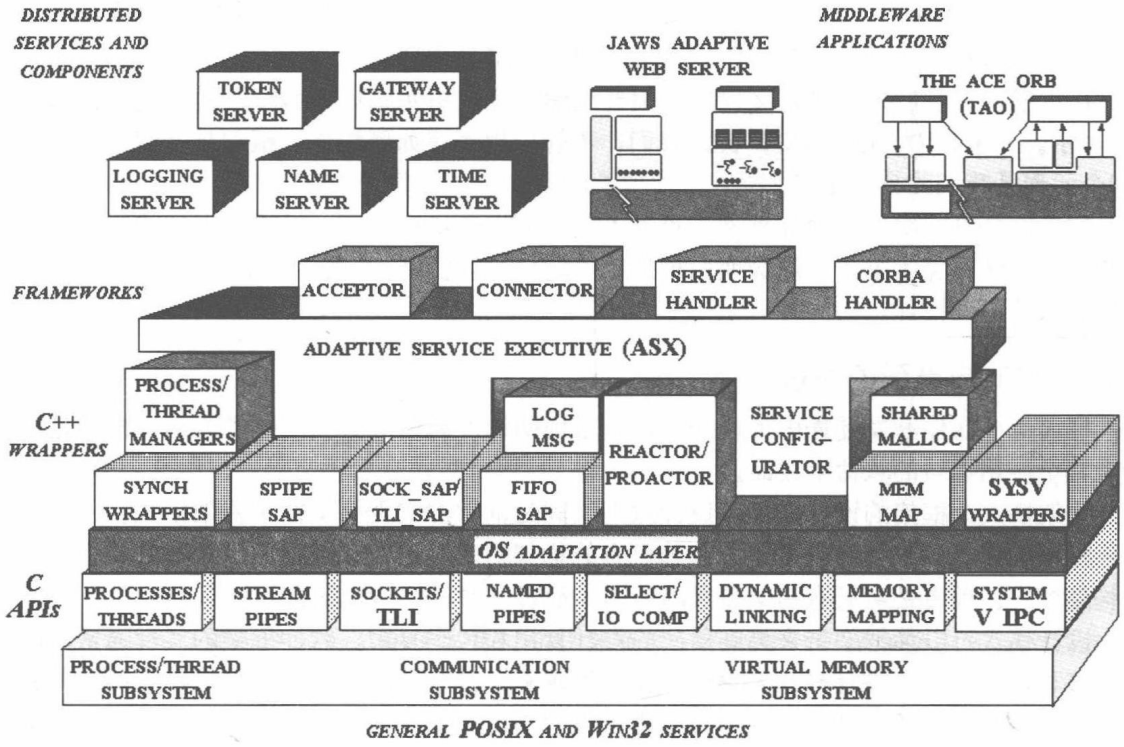

图1-1 ACE结构图

（2）框架

ACE框架是一个在操作系统适配层和C++封装器之上的网络编程框架，可以分为以下几个组件。

❑ 事件分离组件——包括Reactor和Proactor框架，它们为应用程序响应特定I/O、定时器、信号和同步事件提供分离和调度功能。
❑ 服务初始化组件——Acceptor_Connector框架，它将网络通信的初始化和应用程序的处理解耦。
❑ 服务配置组件——Service Configurator框架，它使得应用程序的配置功能可以延缓到执行期。
❑ Task组件——Task框架，主要应用于多线程的并发编程。
❑ 流组件——Streams框架，它为处理流数据的系统提供了结构。
❑ ORB适配器组件——ORB Adapter，ACE使用它可以实现与CORBA的无缝整合。

（3）分布式服务库和组件

除了操作系统适配层、C++封装器和框架组件外，ACE还提供了包装成自包含组件的标