ACE 自适配通信环境中文技术文档

ACE 自适配通信环境（代序）

作者：Douglas C. Schmidt http://www.cs.wustl.edu/~schmidt/
译者：马维达 http://www.flyingdonkey.com/

一、ACE 综述

ACE 自适配通信环境（ADAPTIVE Communication Environment）是可自由使用、开放源码的面向对象（OO）框架（framework），它实现了许多用于并发通信软件的核心模式。ACE 提供了一组丰富的可重用 C++包装外观（wrapper facade）和框架组件，可跨多种平台完成通用的通信软件任务，其中包括：事件多路分离和事件处理器分派、信号处理、服务初始化、进程间通信、共享内存管理、消息路由、分布式服务动态（重）配置、并发执行和同步，等等。

ACE 的目标用户是高性能和实时通信服务和应用的开发者。它简化了使用进程间通信、事件多路分离、显式动态链接和并发的 OO 网络应用和服务的开发。此外，通过服务在运行时与应用的动态链接，ACE 使系统的配置和重配置得以自动化。

ACE 正在进行持续的改进。Riverace 公司（http://www.riverace.com）采用开放源码商业模式对 ACE 进行商业支持。此外，ACE 开发组的许多成员目前正在进行 The ACE ORB（TAO，http://www.cs.wustl.edu/~schmidt/TAO.html）的开发工作。

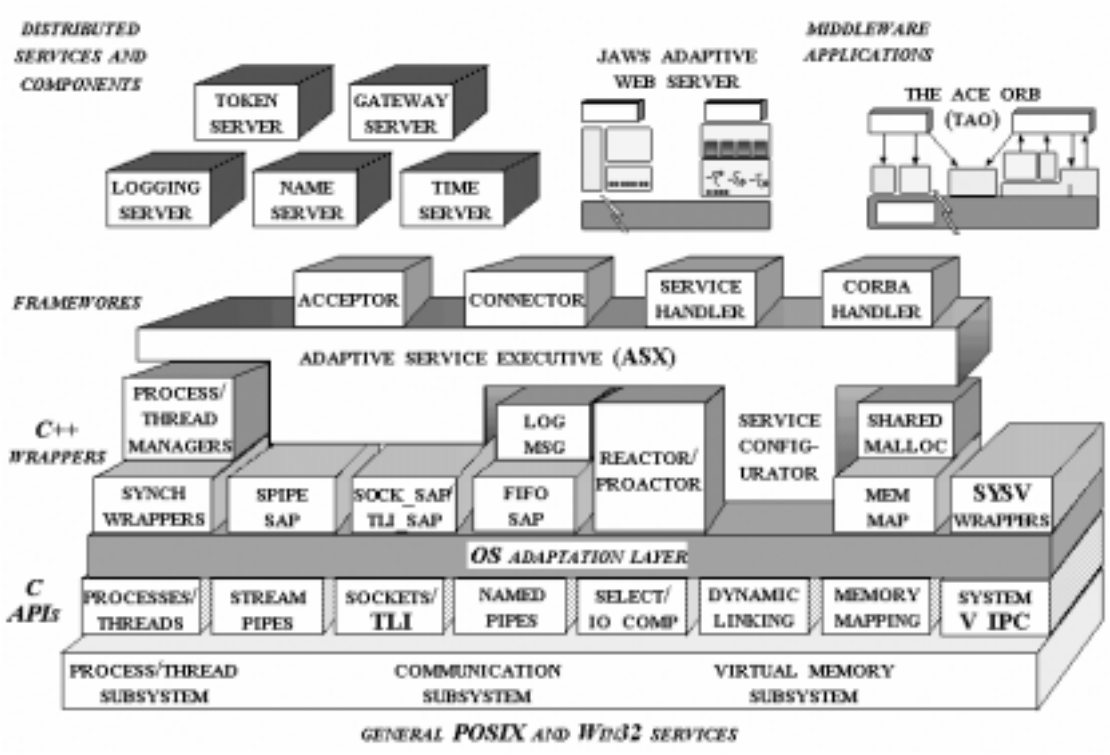
二、使用 ACE 的好处

诸多使用 ACE 的好处包括：

- **增强可移植性**：在 ACE 组件的帮助下，很容易在一种 OS 平台上编写并发网络应用，然后快速地将它们移植到各种其他的 OS 平台上。而且，因为 ACE 是开放源码的自由软件，你无需担心被锁定在特定的操作系统平台或编译器上。
- **更好的软件质量：**ACE 的设计使用了许多可提高软件质量的关键模式，这些质量因素包括通信软件灵活性、可扩展性、重用性和模块性。
- **更高的效率和可预测性：**ACE 经仔细设计，支持广泛的应用服务质量（QoS）需求，包括延迟敏感应用的低响应等待时间、高带宽应用的高性能，以及实时应用的可预测性。
- **更容易转换到标准的高级中间件：**TAO 使用了 ACE 提供的可重用组件和模式。它是 CORBA 的开发源码、遵循标准的实现，并为高性能和实时系统作了优化。为此，ACE 和 TAO 被设计为能良好地协同工作，以提供全面的中间件解决方案。

三、ACE 的结构和功能

下图显示了 ACE 中的关键组件以及它们的层次关系：



图中的结构和各层的组成部分描述如下。

四、ACE OS 适配层

该层直接驻留在用 C 写成的本地 OS API 之上。它提供轻型的类 POSIX OS 适配层，将 ACE 中的其他层及组件和以下与 OS API 相关联的平台专有特性屏蔽开来：

- **并发和同步**：ACE 的适配层封装了用于多线程、多进程和同步的 OS API。
- **进程间通信（IPC）和共享内存**：ACE 的适配层封装了用于本地和远地 IPC、以及共享内存的 OS API。
- **事件多路分离机制**：ACE 的适配层封装了用于对基于 I/O、定时器、信号和同步的事件进行同步和异步多路分离的 OS API。
- **显式动态链接**：ACE 的适配层封装了用于显式动态链接的 OS API。显式动态链接允许在安装时或运行时对应用服务进行配置。
- **文件系统机制**：ACE 的适配层封装了用于操作文件和目录的 OS 文件系统 API。

ACE OS 适配层的可移植性使得 ACE 可运行在许多操作系统上。ACE 已在广泛的 OS 平台上被移植和测试，包括 Win32（也就是，在 Intel 和 Alpha 平台，使用 MSVC++、Borland C++ Builder 和 IBM Visual Age 的 WinNT 3.5.x、4.x、2000、Win95/98 和 WinCE）、Mac OS X、大多数版本的 UNIX（例如，SPARC 和 Intel 上的 Solaris 1.x 和 2.x、SGI IRIX 5.x 和 6.x、DG/UX、HP-UX 9.x、10.x 和 11.x、DEC/Compaq UNIX 3.x 和 4.x、AIX 3.x 和 4.x、UnixWare、SCO，以及可自由使用的 UNIX 实现，比如 Debian Linux 2.x、RedHat Linux 5.2、6.x 和 7.x、FreeBSD 和 NetBSD）、实时操作系统（比如，LynxOS、VxWorks、Chorus ClassiX 4.0、QnX Neutrino、RTEMS 和 PSoS）、MVS OpenEdition 和 CRAY UNICOS。

由于 ACE 的 OS 适配层所提供的抽象，所有这些平台使用同一棵代码树。这样的设计极大地增强了

ACE 的可移植性和可维护性。此外，还有 Java 版本的 ACE 可用（http://www.cs.wustl.edu/~eea1/JACE.html）。

五、OS 接口的 C++包装外观

可以直接在 ACE OS 适配层之上编写高度可移植的 C++应用。但是，大多数 ACE 开发者使用的是上图中所示的 C++包装外观层。通过提供类型安全的 C++接口（这些接口封装并增强本地的 OS 并发、通信、内存管理、事件多路分离、动态链接和文件系统 API），ACE 包装外观简化了应用的开发。应用可以通过有选择地继承、聚合和/或实例化下面的组件来组合和使用这些包装：

- **并发和同步组件：**ACE 对像互斥体和信号量这样的本地 OS 多线程和多进程机制进行抽象，以创建高级的 OO 并发抽象，像主动对象（Active Object）和多态期货（Polymorphic Future）。
- **IPC 和文件系统组件：**ACE C++包装对本地和/或远地 IPC 机制进行封装，比如 socket、TLI、UNIX FIFO 和 STREAM 管道，以及 Win32 命名管道。此外，ACE C++包装还封装了 OS 文件系统 API。
- **内存管理组件：**ACE 内存管理组件为管理进程间共享内存和进程内堆内存的动态分配和释放提供了灵活和可扩展的抽象。

ACE C++包装提供了许多与 ACE OS 适配层一样的特性。但是，这些特性是采用 C++类和对象、而不是独立的 C 函数来构造的。这样的 OO 包装有助于减少正确地学习和使用 ACE 所需的努力。

例如，C++的使用提高了应用的健壮性，因为 C++包装是强类型的。所以，编译器可在编译时、而不是运行时检测类型系统违例。相反，不到运行时，不可能检测像 socket 或文件系统 I/O 这样的 C 一级 OS API 的类型系统违例。

ACE 采用了许多技术来降低或消除额外的性能开销。例如，ACE 大量地使用 C++内联来消除额外的方法调用开销 这样的开销可能由 OS 适配层和 C++包装所提供的额外的类型安全和抽象层次带来。此外，对于关键性能的包装，比如 socket 和文件 I/O 的 send/recv 方法，ACE 会避免使用虚函数。

六、框架

ACE 还包含一个高级的网络编程框架，集成并增强了较低层次的 C++包装外观。该框架支持将并发分布式服务动态配置进应用。ACE 的框架部分包含以下组件：

- **事件多路分离组件：**ACE Reactor(反应器)和 Proactor（前摄器）是可扩展的面向对象多路分离器，它们分派应用专有的处理器，以响应多种类型的基于 I/O、定时器、信号和同步的事件。
- **服务初始化组件：**ACE Acceptor（接受器）和 Connector（连接器）组件分别使主动和被动的初始化任务与初始化一旦完成后通信服务所执行的应用专有的任务去耦合。
- **服务配置组件：**ACE Service Configurator（服务配置器）支持应用的配置，这些应用的服务可在安装时和/或运行时被动态装配。
- **分层的流组件：**ACE Stream 组件简化了像用户级协议栈这样的由分层服务组成的通信软件应用的开发。
- **ORB 适配器组件：**通过 ORB 适配器，ACE 可以与单线程和多线程 CORBA 实现进行无缝集成。

ACE 框架组件便利了通信软件的开发，它们无需修改、重编译、重链接，或频繁地重启运行中的应用，就可被更新和扩展。在 ACE 中，这样的灵活性是通过结合以下要素来获得的：（1）C++语言特性，比如模板、继承和动态绑定，（2）设计模式，比如抽象工厂、策略和服务配置器，以及（3）OS 机制，

比如显式动态链接和多线程。

七、分布式服务和组件

除了 OS 适配层、C++包装外观和框架组件，ACE 还提供了包装成自包含组件的标准分布式服务库。尽管这些服务组件并不是 ACE 框架库的严格组成部分，它们在 ACE 中扮演了两种角色：

1. **分解出可重用分布式应用的“积木”：**这些服务组件提供通用的分布式应用任务的可重用实现，比如名字服务、事件路由、日志、时间同步和网络锁定。
2. **演示常用的 ACE 组件的用例：**这些分布式服务还演示了怎样用像 Reactor、Service Configurator、Acceptor 和 Connector、Active Object，以及 IPC 包装这样的 ACE 组件来有效地开发灵活、高效和可靠的通信软件。

八、高级分布式计算中间件组件

即使使用像 ACE 这样的通信框架，开发健壮、可扩展和高效的通信应用仍富有挑战性。特别是，开发者必须掌握许多复杂的 OS 和通信的概念，比如：

- 网络寻址和服务标识。
- 表示转换，比如加密、压缩和在异种终端系统间的字节序转换。
- 进程和线程的创建和同步。
- 本地和远地进程间通信（IPC）机制的系统调用和库例程。

通过采用像 CORBA、DCOM 或 Java RMI 这样的高级分布式计算中间件，有可能降低开发通信应用的复杂性。高级分布式计算中间件驻留在客户端和服务器之间，可使分布式应用开发的许多麻烦而易错的方面自动完成，包括：

- 认证、授权和数据安全。
- 服务定位和绑定。
- 服务注册和启用。
- 事件多路分离和分派。
- 在像 TCP 这样的面向字节流的通信协议之上实现消息帧。
- 涉及网络字节序和参数整编（marshaling）的表示转换问题。

为给通信软件的开发者提供这些特性，在 ACE 中绑定了下面的高级中间件应用：

1. **The ACE ORB（TAO）：**TAO 是使用 ACE 提供的框架组件和模式构建的 CORBA 实时实现，包含有网络接口、OS、通信协议和 CORBA 中间件组件及特性。TAO 基于标准的 OMG CORBA 参考模型，并进行了增强的设计，以克服传统的用于高性能和实时应用的 ORB 的缺点。TAO 像 ACE 一样，也是可自由使用的开放源码软件。
2. **JAWS：**JAWS 是高性能、自适配的 Web 服务器，使用 ACE 提供的框架组件和模式构建。JAWS 被构造成“框架的框架”。JAWS 的总体框架含有以下组件和框架：事件多路分派器、并发策略、I/O 策略、协议管道、协议处理器和缓存虚拟文件系统。每个框架都被构造成一组协作对象，通过组合和扩展 ACE 中的组件来实现。JAWS 也是可自由使用的开放源码软件。

九、主页

ACE 的主页为：http://www.cs.wustl.edu/~schmidt/ACE.html，在这里可获得最新版本的 ACE 以及其他相关资源。中文文档见 http://www.flyingdonkey.com/ace/。