<<STL扩展技术手册卷I>>

图书基本信息

书名：<<STL扩展技术手册卷I>>

13位ISBN编号：9787111242277

10位ISBN编号：7111242270

出版时间：2008-9

出版时间：机械工业出版社

作者：Matthew Wilson

页数：414

前言

难道每门语言都难免日趋复杂，并最终绊倒在复杂性的门槛上吗？
—AdamConnor 难用的话，少用就是了。
—MelanieKrug 事物的两面性 3年前，《ImperfectC++》快要完工时，我跟编辑说起这本《ExtendedSTL》，当时我信心满满地声称它会是一本易读易懂。
且轻薄短小得可以轻松从两个抽象层之间滑过的小册子。
此外我还保证会在半年之内写完。
结果呢？
在写这篇序言的时候，离当初约好的截稿日期已经过去了一年半有余，而且，本来计划好的一本薄薄的。
约十六至二十个章节的小册子现在也膨胀成了两卷本，其中第一卷洋洋洒洒四十三个章节（含“插曲”章节），哦，对了，CD上还有三章呢…但话说回来，当初有一个保证现在仍然可以说是成立的，那就是这是一本对任何有一定C++经验的读者来说都轻松易懂的书。
为什么这本书后来的情况远远超出我当初的预计呢？
并不是因为我是软件工程师——大家都知道软件工程师估计的工作量，乘三之后才是实际需要的时间。
而是因为（我认为）以下四个重要的原因： 1。
STL并不直观，花费可观的智力投资后方能熟练运用。
2。
没错，STL在技术上功臻化境，没错，STL在内聚性方面超凡入圣。
然而，STL前瞻性不够，对于在它那有限的概念定义之外的抽象，它并不能妥善应付。
3。
C++语言本身是不完美的。
4。
C++是一门难学的语言，但你得到的回报是效率，而且同时又并没有牺牲设计。
最近几年C++与时俱进，这一方面意味着C++变得非常强大，但另一方面也暴露出了其晦涩难懂的一面。
如果你试着编写一个具有一定规模并用到模板元编程的模板库，那么一种可能是你将学到许多东西，并掌握了一个强大的工具，但同样也很可能的是你编写出的那堆东西除了C++狂热信徒之外谁也无法理解。
C++的使用精神本来就是扩展。
除了很有限的一些应用是将C++看作“更好的C”来使用的之外，绝大多数C++使用都是围绕着类型定义（类。
枚举。
结构。
联合）来进行的，而且这些自定义类型很大程度上被做成与内建类型界面一致。
也正因为这个原因，C++中的许多内建操作符都是允许重载的。
这样，一个vector，比如说，便可以重载operator[]来模拟内建数组的界面，再举个例子：任何可被拷贝（Copyable）的类型（一般）都定义（重载）了拷贝赋值操作符（operator=）。
如此等等，不一而足。
但由于C++的不完美。
强大。
以及极强的可扩展性，伴随而来的便是JoelSpolsky所说的抽象渗漏法则（theLawofLeakyAbstractions）：“所有非平凡的抽象在某种程度上都是有漏洞的。
”简单来说这句话就意味着要想顺利使用非平凡的抽象就必须对抽象下面的东西有所了解。
这也正是许多C++开发者重新发明轮子的原因之一。

其实这里的原因并不仅仅是众所周知的所谓“非我发明症”（NIH），而是因为我们常常发现自己所用的第三方组件除了百分之八十的功能是自己能理解并使用的之外，剩下的百分之二十往往裹着一团晦涩的黑气，造成后者的原因很多：复杂性。
与既有概念或惯用法的不一致。
低效。
效率。
范围局限性。
设计或实现的不优雅。
糟糕的编码风格等等。
而且，编译技术现阶段的一些实际问题还会极大地加剧这种情况，尤其是遇到模板实例化过程中的错误消息时。

我觉得我之所以有资格写这本书，原因之一就是我花了大量的时间来研究并实现STL相关的库，而不是接受C++标准（1998）所指定的库或其他人写的库。
而我决定写这本书的原因之一则是想将我在以上过程中学到的东西总结出来。
如果你想要编写STL扩展，这本书可以为你提供帮助，而就算你只是想使用其他人写的STL扩展，这本书也同样有用，因为抽象渗漏法则决定了你很可能会不时需要掀开抽象这块幕布往里面瞧一瞧。

UNIX编程的原则　　在《UNIX程序设计艺术》（Addison-Wesley，2004）中，EricRaymond总结了UNIX社群的最佳实践准则，这些准则来自大量不同的经验。
在我们改装STL的宏大计划中，这些准则就像标灯一样为我们指明方向：　　· 清晰原则：清晰比巧妙重要。

· 组合原则：设计能够互相连接的组件。
· 多样性原则：质疑任何被声称为“真正唯一”的途径。
· 经济原则：程序员时间是昂贵的，在它跟机器时间之间，优先节省前者。
· 可扩展性原则：在未来时态下设计，因为未来比你想像得来得更快。
· 生成原则：避免手动编码，可以的话，编写程序来生成程序。
· 最小意外原则：在接口设计中作出的决策应该始终是那个令人最少感到意外的选择。
· 模块性原则：编写简单的模块，模块与模块间通过干净的接口连接。
· 最大意外原则：如果免不了要失败的话，要弄出最大动静，而且失败得越早越好。
· 优化原则：首先要能工作，然后才能谈得上优化。
· PrincipleofParsimony：Writelargecomponentsonlywhenitisclearbydemonstrationthatnothingelsewilldo。
· 吝啬原则：除非能够明确证明别无它法，否则不要编写大的组件。
· 健壮性原则：透明性和简单性是健壮性的父母。
· 分离原则：策略和机制分离，接口与引擎分离。
· 简单原则：设计应该是简单的，只在必须的时候才增加或暴露复杂性。
· 透明原则：设计的时候应考虑透明性，以方便检查和调试。

<<STL扩展技术手册卷I>>

内容概要

本书以STLSoft为基础，广泛深入地论述了C++标准库STL的相关内容。
全书共三部分43章，包括标准库模板、扩展STL、元素引用类别、DRY SPOT原则、抽象泄漏法则、契约式编程、约束、垫片、不完备结构一致性的发端、资源获取、模板工具、推断式接口适配、Henney假说、适配、遍历进程和模块、环境变量、字符串分词、聚集分散的I/O以及迭代器等内容。
全书通过严谨的表述与丰富的示例，将概念和理论与实际的设计和代码结合起来，从而使读者既能深刻地理解STL的知识，又能熟练地掌握STL运用方法。

作者简介

Matthew Wilson，Synesis软件公司的软件开发顾问，STLSoft和Pantheios库的创建者，《Imperfect C++》（Addison-Wesley）的作者。
他曾是《C/C++ Users Journal》（C/C++用户杂志）的专栏作家，也是一些主要出版物的撰稿者，具有15年以上的C++经验。
他拥有英国曼彻斯特大学

<<STL扩展技术手册卷I>>

书籍目录

章节摘录

第2章.扩展STL：STL与真实世界的碰撞　　好的定律是清楚和简单的，并让使用者在面对具体问题时应对自如。
-RonMcCallum教授　　搞清楚你此刻应当做什么，然后去做。
-BillyConnolly　　前章涵盖了STL的要点，包括容器、迭代器、算法、函数对象、分配器及适配器的核心概念。
可惜扩展STL时，这些概念不是太抽象就是太粗糙。
这一章中我们将针对本卷中关于集合及迭代器适配器的材料，进一步探讨这些概念。
2.1. 术语　　虽然标准库中包含了许多原STL（Stepanov与同事开发的版本）的内容，但却(还)没有全盘照搬。
比方说C++03标准目前只定义了基于树的关联容器，而没有包含原STL中基于hash表的关联容器。
尽管已经有建议要加入此类容器，并且会在下个C++标准发布时被采纳，但眼下的状况就是标准库并非STL的超集。
另一方面，标准库中包含不属于STL但却与STL兼容的组件，即IOStreams。
当把IOStreams看作严肃的I/O程序库，无论从哪方面考量，附着其上的STL兼容接口都对C++有利无害。
STL与标准库有许多重叠的地方，但两者却又是不同的。
在这一卷中，我不去理会任何未被标准库采纳的STL组件，反之亦然。
所以，当我提到“标准组件”的时候，我的意思是指那些既在STL库又在标准库中，且定义与标准库定义一致的组件。
本书主要讨论的是扩展STL的技术，我们还需要一套术语才能进行下去，这些扩展不仅是增加新容器而已。
事实上，大部分扩展提供的是对一个区间内的元素的访问手段，但它们根本不是容器，我称它们为STL集合。
我们在下一个小节中讨论集合概念，及其与容器概念的关系。
2.2. 集合　　STL本身只关注容器。
标准把容器定义为“存储别的对象，[同时]对这些对象负有分配及回收的责任，途径是对象的构造函数，析构函数，插入操作以及删除操作”(C++-03：23.1；1)。
但是有比容器多得多的类型属于对象集合（图2.1）。
在本书所有后续章节，特别是在第二部分中，我将使用在本章中定义的分类。

媒体关注与评论

“《Extended STL》不只是一本关于适应STL并用于日常工作的书，它也是一次冒险旅行，它带你经历软件设计和概念、C++的强力技术，以及真实世界软件开发中的危险，换句话说，它是一本Matthew Wilson风格的书。
如果你对C++的态度是严肃认真的，我认为你应该阅读它。
” —Bjorn Karlsson，主设计师，ReadSoft；〈Beyond the C++ Standard Library： An Introduction to Boost》的作者

编辑推荐

著名的C++专家MatthewWilson在《STL扩展技术手册（卷1）：集合和迭代器》中展示了如何超越C++标准并扩展标准模板库（StandardTemplateLibrary，STL），进入包含API和非标准集合的更广阔的C++世界，以编写更有效、更有表达力、更灵活、更健壮的软件。

在《STL扩展技术手册（卷1）：集合和迭代器》中，Wilson使用自己创新的技术帮助读者掌握STL扩展特性，这体现在两个方面：将专用技术库和操作系统API适配为STL兼容的集合，并定义精密的迭代适配器，使STL潜在的效率和表现力得以实现。

Wilson用实际的例子阐明了几个强大的概念和技巧，让你在连STL的创造者都未曾预见的方向上扩展STL，其中包括集合、元素参考类别、外部迭代器失效和推断式接口适配。

对于那些对STL知之甚微的C++程序员，《STL扩展技术手册（卷1）：集合和迭代器》将是宝贵的资源。

版权说明