标准模板库 - 维基百科，自由的百科全书

本文介绍的是C++的模板库。关于STL的其他用法，请见“STL”。

标准模板库（英文：Standard Template Library，缩写：STL），是一个C++软件库，大量影响了C++标准程序库但并非是其的一部分。其中包含4个组件，分别为算法、容器、函数、迭代器。[1]

模板是C++程序设计语言中的一个重要特征，而标准模板库正是基于此特征。标准模板库使得C++编程语言在有了同Java一样强大的类库的同时，保有了更大的可扩展性。

目录

[隐藏]

历史[编辑]

标准模板库系由Alexander Stepanov创造于1979年前后，这也正是比雅尼·斯特劳斯特鲁普创造C++的年代。

虽然David R. Musser于1971年开始即在计算机几何领域发展并倡导某些泛型程序设计观念，但早期并没有任何编程语言支持泛型程序设计。第一个支持泛型概念的语言是Ada。[来源请求] Alex和Musser曾于1987开发出一套相关的Ada library.

标准模板库设计人Stepanov早期从事教育工作，1970年代研究泛型程序设计，那时他与其同事一起在GE公司开发出一个新的程序语言——Tecton。

1983年，Stepanov先生转至Polytechnic大学教书，继续研究泛型程序设计，同时写了许多Scheme的程序，应用在graph与network的算法上，1985年又转至GE公司专门教授高级程序设计，并将graph与network的Scheme程序，改用Ada写，用了Ada以后，他发现到一个动态（dynamically）类型的程序（如Scheme）与强制（strongly）类型的程序（如Ada）有多么的不同。

在动态类型的程序中，所有类型都可以自由的转换成别的类型，而强制类型的程序却不能。但是，强制类型在出错时较容易发现程序错误。

1988年Stepanov先生转至HP公司运行开发泛型程序库的工作。此时，他已经认识C语言中指针(pointer)的威力，他表示一个程序员只要有些许硬件知识，就很容易接受C语言中指针的观念，同时也了解到C语言的所有数据结构均可以指针间接表示，这点是C与Ada、Scheme的最大不同。

Stepanov并认为，虽然C++中的继承功能可以表示泛型设计，但终究有个限制。虽然可以在基础类型（superclass）定义算法和接口，但不可能要求所有对象皆是继承这些，而且庞大的继承体系将减低虚拟（virtual）函数的运行效率，这便违反的前面所说的“效率”原则。

到了C++模板观念，Stepanov参加了许多有关的研讨会，与C++之父Bjarne讨论模板的设计细节。如，Stepanov认为C++的函数模板（function template）应该像Ada一样，在声明其函数原型后，应该显式的声明一个函数模板之实例（instance）；Bjarne则不然，他认为可以通过C++的重载（overloading）功能来表达。

Stepanov想像中的函数模板：

```
in*.hpptemplate<classT>Tsquare(Tx)
{returnx*x;}in*.cppdoublesquare(double);cout<<square(3.3);intsquare(int);cout<<square(3);
```

Bjarne认为的函数模板：

```
in*.hpptemplate<classT>Tsquare(Tx){returnx*x;}in*.cppcout<<square(3.3);cout<<square(3);
```

几经争辩，Stepanov发现Bjarne是对的（参考侯俊杰《标准模板库讲座・第三章》）。事后Stepanov回想起来非常同意Bjarne的作法。

“	template 引数推导机制（arguments deduction，在标准模板库中占非常重要的角色。Alexander Stepanov（标准模板库创造者）i 道‘1992 我重回generic-library的开发工作。这时候C++有了template 我发现Bjarne完成了一个非常美妙的设计。之前我在Bell Lab曾参与数次template的相关设计讨论，并且非常粗暴地要求 可能像Ada generics那样。我想由于我的争辩是如此地粗暴，他决定反对。我了解到在C++中除了拥有template classes之 。然而我认为template function应该像Ada generics一样，也就是说它们应该是显式实例，Bjarne没有听进我的话，他设计 的template是以一个重载化机制 （overloading mechanism来进行隐式实例这项特殊的技术对我的工作具有关键性的影响， 可能完成的许多动作。我非常高兴Bjarne当初没有听我的意见。’（DDJ 1995 年三月号）

事实上，C++的模板，本身即是一套复杂的宏语言（macro language），宏语言最大的特色为：所有工作在编译时期就已完成。显式的声明函数模板之实例，与直接通过C++的重载功能隐式声明，结果一样，并无很大区别，只是前者加重程序员的负担，使得程序变得累赘。

1992年Meng Lee加入Alex的项目，成为另一位主要贡献者。

1992年，HP泛型程序库项目结束，小组解散，只剩下Stepanov先生与Meng Lee小姐（她是东方人，标准模板库的英文名称其实是取STepanov与Lee而来[2]），Lee先前研究的是编译器的制作，对C++的模板很熟，第一版的标准模板库中许多程序都是Lee的杰作。

1993年，Andy Koenig到史丹佛演讲，Stepanov便向他介绍标准模板库，Koenig听后，随即邀请Stepanov参加1993年11月的ANSI/ISO C++标准化会议，并发表演讲。

Bell实验室的Andrew Koenig于1993年知道标准模板库研究计划后，邀请Alex于是年11月的ANSI/ISO C++标准委员会会议上展示其观念。并获得与会者热烈的回应。

1994年1月6日，Koenig寄封电子邮件给Stepanov，表示如果Stepanov愿意将标准模板库的帮助文档撰写齐全，在1月25日前提出，便可能成为标准C++的一部分。Stepanov回信道：″Andy, are you crazy?″ 。 Koenig便说：″Well, yes I am crazy,but why not try it?″。

Alex于是在次年夏天在Waterloo举行的会议前完成其正式的提案，并以百分之八十压倒性多数，一举让这个巨大的计划成为C++ Standard的一部分。

标准模板库于1994年2月年正式成为ANSI/ISO C++的一部分，它的出现，促使C++程序员的思维方式更朝向泛型编程（generic program）发展。

内容[编辑]

STL 将“在数据上执行的操作”与“要执行操作的数据分开”，分别以如下概念指代：

- 容器：包含、放置数据的地方。
- 迭代器：在容器中指出一个位置、或成对使用以划定一个区域，用来限定操作所涉及到的数据范围。
- 算法：要执行的操作。

容器[编辑]

标准模板库包含了序列容器（sequence containers）与关系容器（associative containers）。

数据容器
序列容器 - 有序集
vector
list
forward_list （单向链表）
deque （双端队列）
array
关联容器 - 无序集
set
multiset
map
multimap
unordered_set **unordered_multiset** **unordered_map** **unordered_multimap**
其他类型的容器
bitset
valarray

迭代器[编辑]

迭代器是泛化的指针，通过使用迭代器，开发者可以操作数据结构而无需关心其内部实现。根据迭代器的操作方式的不同，迭代器分为五种[3]：

- 输入迭代器
- 输出迭代器
- 前向迭代器
- 双向迭代器
- 随机访问迭代器

算法[编辑]

STL提供了一些常见的算法，如排序和搜索等。这些算法与数据结构的实现进行了分离。因此，用于也可对自定义的数据结构使用这些算法，只需让这些自定义的数据结构拥有算法所预期的迭代器。[4]。

函数对象[编辑]

狭义的函数对象即重载了操作符()的类的实例，而广义来讲所有可用 x(...) 形式调用的 x 都可称为函数对象、或曰可调用对象。[5]。

适配器（Adaptor）[编辑]

适配器为一个模板类，用于提供接口映射。[6]。

与C++标准程序库的差异[编辑]

一个常见的误解是STL是C++标准程序库的一部分，但事实上并非如此。例如hash table的数据结构实现在STL中有模板可供调用，但C++标准程序库一直到C++11才加入了。参见无序关联容器_(STL)。

参考文献[编辑]

1. 跳转 ^Holzner, Steven. C++ : Black Book. Scottsdale, Ariz.: Coriolis Group. 2001: 648. ISBN 1-57610-777-9. The STL

is made up of *containers*, *iterators*, *function objects*, and *algorithms*

2. 跳转 ^# Al Stevens Interviews Alex Stepanov. After all, STL stands for Stepanov and Lee...

3. 跳转 ^Alexander, Stepanov. The Standard Template Library(PDF): 6. 1995. Depending on the operations defined on them, there are five categories of iterators: input iterators, output iterators, forward iterators, bidirectional iterators and random access iterators.

4. 跳转 ^Alexander, Stepanov. The Standard Template Library(PDF): 41. 1995.

5. 跳转 ^Alexander, Stepanov. The Standard Template Library(PDF): 14. 1995.

6. 跳转 ^Alexander, Stepanov. The Standard Template Library(PDF): 55. 1995.

参见[编辑]

- C++标准程序库

外部链接[编辑]

- C/C++ reference includes a section on the STL
- STL programmer's guide official guide from SGI
- Bjarne Stroustrup on The emergence of the STL (Page 5, Section 3.1)
- Apache stdcxx portable Open Source implementation based on Rogue Wave STL
- STLport multiplatform STL implementation
- Dinkumware commercial STL implementation (ships with Visual C++ and several other compilers)
- Rogue Wave STL Class Reference
- MPTL shared-memory parallel extension of the STL using the Native POSIX Thread Library。
- STXXL: an STL implementation for external memory.
- STLSoft libraries： open-source, 100% header-only, C/C++ libraries of technology-specific facades and STL extensions.