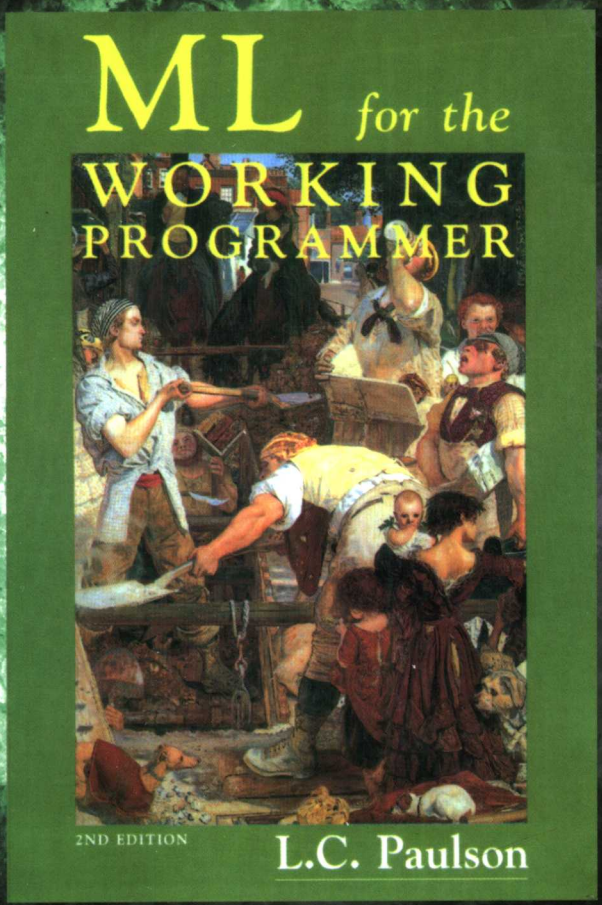
计 算 机 科 学 丛 书

原书第2版

ML程序设计教程

(英) Lawrence C. Paulson 著　　柯 韦 译



ML for the Working Programmer

Second Edition

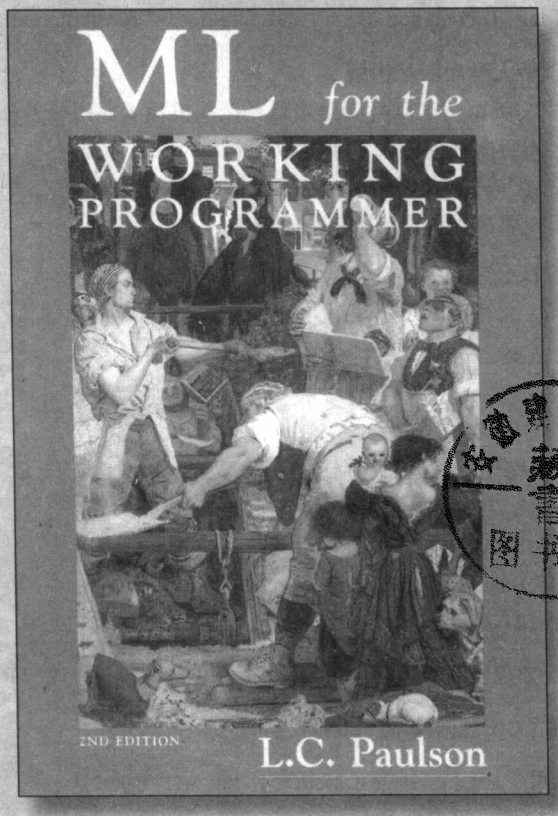
机械工业出版社
China Machine Press

计 算 机 科 学 丛 书

原书第2版

ML程序设计教程

(英) Lawrence C. Paulson 著　　柯 韦 译



ML for the Working Programmer
Second Edition

机械工业出版社
China Machine Press

本书详细讲解如何使用ML语言进行程序设计，并介绍函数式程序设计的基本原理。书中特别讲述了为ML的修订版所设计的新标准库的主要特性，并且给出大量例子，涵盖排序、矩阵运算、多项式运算等方面。大型的例子包括一个一般性的自顶向下语法分析器、一个λ-演算归约程序和一个定理证明机。书中也讲述了关于数组、队列、优先队列等高效的函数式实现，并且有一章专门讨论函数式程序的形式论证。

本书可作为高等院校计算机专业相关课程的教材，也适合广大程序设计人员参考。

本书版权登记号：图字：01-2004-3938

图书在版编目（CIP）数据

ML程序设计教程（原书第2版）/（英）保罗森（Paulson, L. C.）著；柯韦译. -北京：机械工业出版社，2005.5

（计算机科学丛书）

书名原文：ML for the Working Programmer, 2nd edition

ISBN 7-111-16121-1

Ⅰ.M…　Ⅱ.①保…②柯…　Ⅲ.程序语言-程序设计　Ⅳ.TP312

中国版本图书馆CIP数据核字（2005）第017060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：朱起飞　冯春丽

北京瑞德印刷有限公司印刷·新华书店北京发行所发行

2005年5月第1版第1次印刷

787mm × 1092mm 1/16 · 24.25印张

印数：0 001-3000册

定价：45.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭橥了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短、从业人员较少的现状下，美国等发达国家在其计算机科学发展的几十年间积淀的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出Tanenbaum，Stroustrup，Kernighan，Jim Gray等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及庋藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专诚为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都步入一个新的阶段。为此，华章公司将加大引进教材的力度，在“华章教育”的总规划之下出版三个系列的计算机教材：除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”；同时，引进全美通行的教学辅导书“Schaum's Outlines”系列组成“全美经典学习指导系列”。为了保证这三套丛书的权威性，同时也为了更好地为学校和老师们服务，华章公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域

的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这三套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经三十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证，但我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。教材的出版只是我们的后续服务的起点。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

电子邮件：hzedu@hzbook.com
联系电话：（010）68995264
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

专家指导委员会

（按姓氏笔画顺序）

尤晋元	王　珊	冯博琴	史忠植	史美林
石教英	吕　建	孙玉芳	吴世忠	吴时霖
张立昂	李伟琴	李师贤	李建中	杨冬青
邵维忠	陆丽娜	陆鑫达	陈向群	周伯生
周立柱	周克定	周傲英	孟小峰	岳丽华
范　明	郑国梁	施伯乐	钟玉琢	唐世渭
袁崇义	高传善	梅　宏	程　旭	程时端
谢希仁	裘宗燕	戴　葵		

译 者 序

多年前我曾经认为自己是一个程序员，对程序设计有着执著的兴趣。试图洞悉本质的渴望以及相关知识的匮乏往往令我陷入苦思。于是我开始了新一轮的阅读。在一个图书馆中书架林立的某处，我发现了这本写给程序员的书。在我相对贫乏的阅读历史中，书名带有“程序员”（programmer）的寥寥无几，又怎能不留意呢？这本书最终没有让我失望。

如果说过去我对程序设计有过一些有意义的思考，而从中得到了乐趣的话，函数式程序设计则系统地叙述了这些令人兴奋的闪光之处。这种基于某种数学概念的方法把我们从程序运行的细节中解放出来，转而关注如何去表达。程序因此变成一个个静态的方程式，安详地躺在那里，展示着简单而丰富的内涵。我们则不再被传统的动态步骤困扰，思想在清晰的、静态的表达基础上轻松地延伸着，这是多么美妙的感受啊！我们曾经认为，如果无法理解赋值语句就不能进行程序设计，我们费了多少力气去习惯它，以至于后来已完全在脑海中根深蒂固了。现在我们要从这种被扭曲的思维中走出来，回到更为自然的思考方式去，而函数式程序设计则是众多方法中的一个。

ML语言是函数式语言中较为经典的，它的语法简单优美，语义清晰准确并易于理解，语言的实现也相当高效和严谨。和最新式的函数式语言不同，ML牺牲了一些数学上的纯粹性，换来了相对简单的语义描述，理解它不需要十分高深的数学抽象思考。这对于语言的实用性是十分重要的。正如书中所说，我们不希望从机器语言繁琐的运行步骤中走出来，而立即又掉进同样复杂繁琐的数学模型中去。作为一种入门的函数式程序设计语言，ML不仅对于初学程序设计的学生是一个好的语言，对于有经验的程序员来说也是一个好的桥梁，程序员们将或多或少地发现，他们过去的一些思考竟有如此简单的归宿。我想，这也是原书命名的由来。

原书是一本非常流行的ML语言和函数式程序设计教材，尽管第2版面世至今已有很长时间了，但依旧被国外许多大学的相关科目广泛采用。书中在介绍ML语言的同时更多地阐述了语言和函数式程序设计的思考方法。大量的例子条理分明，深浅搭配适度。随处可见的精心安排的旁白涉及了计算机语言理论研究的许多方面，作为科普读物来说也非常赏心悦目。我很喜欢这本书，衷心地希望能将之介绍给更多的中国读者。我的翻译工作是认真而仔细的，然而由于水平有限，令人遗憾之处恐在所难免，希望读者不要因此受阻，能在阅读中有所收获。

我感谢机械工业出版社，她们出版了许多计算机基础理论的译著，这一本也不例外。我衷心希望此举能将计算机不仅作为一种技术，也作为一种科学推广，产生更多的思想积淀，形成更广泛的基础。我还要感谢在此书翻译过程中给我宝贵指点和帮助的各位尊敬的老师和编辑们。原书的作者Paulson博士也对我的提问作出了耐心的讲解。

第2版序

每次重新印刷，书中的一些小错误都会悄悄地消失。但是重新印刷并不会做大的改进，尽管改进是很有意义的。因为这样做会影响页码编号，使内容产生差别而互不兼容。重大的改动积累起来（以及编辑的催促）便产生了这个第2版。

非常幸运的是，对ML语言的改动都凑到了一起。ML有了新的标准库，并且语言本身也经过了修订。值得强调的是，这些改动没有牺牲ML固有的可靠性。一些晦涩的技术要点已经得到简化，原先定义中不合适的地方也改正了。现有的程序几乎不用改动就可以继续运行。最明显的改动是增加了新的字符类型，还有一套新的顶层库函数。

这版新书及时反映了语言的变化，并在格局安排上做了很大的改进。模块提前到第2章就介绍了，而不是放到第7章，它的使用贯穿全书。这使得重点有所转移，从数据结构（比如二叉搜索树）变为抽象类型（比如字典）。抽象类型通常在某一小节引入，并给出它的ML签名。然后讲解实现背后的思想，最后给出ML结构的代码。虽然评审对第1版较为宽容，但是许多读者要求重新组织内容。

书中的程序不仅移动了位置，而且重新编写过。它们反映了如何使用模块的新思路。由于open声明会隐藏模块结构，所以已经很少出现了。函子也仅在需要的时候才使用。现在，程序的缩进也排得很仔细了，加上其他改进，代码变得更加易读，质量也更好了，表现在合并排序更为简单迅速，优先队列也更快了。

新标准库也要求尽早地提到模块。尽管这样做就必须修改现有代码，但它能使ML在现实语言中的地位更加稳固。新标准库的设计经过了漫长的磋商过程，主要是为了提供全面的支持，同时避免过分的复杂化。它的组织显示了ML模块的优点。字符串处理、输入输出和系统接口这些模块都提供了实际的功能改进。

新标准库导致很多代码必须重写。当新标准库包含函数*foldl*时，读者一般不愿意再看到以前类似的函数*foldleft*。但是这些函数不是完全一样的，所以，重写并不只是改个名字那么简单。很多曾讲述实用函数的章节，现在都要对照新标准结构进行检查更新。

更新过的参考文献说明了函数式程序设计和ML在各领域的广泛应用。ML符合构建可靠系统的要求。软件工程师们需要一种能提供类型安全、模块化、编译时一致性检测和容错（异常）的语言。ML程序是可移植的，这要部分归功于标准库。商业化的编译器正在不断提高质量和效率。ML的运行速度可以和C媲美，特别是在需要复杂存储管理的应用场合。本书的名字（指英文书名）曾引来一些嘲笑，但却给出了很好的提示。

我最惊讶的是看到第1版出现在初级程序员的手中，尽管第一页上写着让他们看点别的书。为了帮助初学者，我加入了一些特别简单的例子，并将大多数参考引用从正文中移走。重写了第1章，试图以一种既适合初学者又适合有经验的C程序员的方式介绍基本的程序设计概念。这比听上去要容易：C并不想给程序员一个解决问题的环境，而仅仅是给底层的硬件稍加装扮。第1章仍旧包含一些基本的计算机常识，但教师也许还是喜欢从第2章开始讲述，因为它带有

简单的会话过程。

在书的末尾列出了一些项目建议。我故意说得不很明确，因为一个大项目的第一步就是准确地分析需求。我希望看到越来越多的人在项目中采用ML，选择ML，特别是取代像C这样不安全的语言，最终会被看作是专业的一种标志。

我非常感谢所有对这一版给出有用的意见、建议或代码的人们。他们分别是Matthew Arcus、Jon Fairbairn、Andy Gordon、Carl Gunter、Michael Hansen、Andrew Kennedy、David MacQueen、Brian Monahan、Arthur Norman、Chris Okasaki、John Reppy、Hans Rischel、Peter Sestoft、Mark Staples和Mads Tofte。Sestoft还给了我一个Moscow ML的预发行版，含有库的更新。CUP的Alison Woollatt编写了LaTeX的类文件。Franklin Chen和Namhyun Hur报告了前一版中的错误。

前　言

本书源于对Standard ML和函数式程序设计的讲稿。它仍可以作为函数式程序设计的课本——一本面向实用，而不是标准的、理想化的书——然而，它主要是一本有效使用ML的指南。它甚至讨论了ML的命令式特性。

有些内容需要离散数学的知识，例如初等逻辑和集合论。读者会发现以往的程序设计经验是有用的，但不是必需的。

本书是一本程序设计手册，而不是参考手册。它覆盖了ML的主要方面，但并不尽述所有的细节。它在理论原理上花费了一些篇幅，但主要还是关心高效的算法和实际的程序设计。

本书的组织反映了我的教学经验。高阶函数出现得较晚，在第5章讲述。惯常的做法是在一开始就介绍一些不甚自然的例子，这样做只能使学生们感到困惑。高阶函数的概念是不容易理解的，需要充分的预备知识。所以，本书从基本类型、表和树开始讲述。当讲到高阶函数时，很多相关的例子已经是现成的了。

练习的难度相差很大。它们不是用来评测学生的，而是为了提供实践机会，拓展内容和激发讨论的。

*本书一览。*大多数章节都专注于ML的各个方面。第1章介绍了函数式程序设计的背景思想，以及ML的历史概况。第2～5章涵盖了ML的函数式部分，包括对模块的简介。讲述了基本类型、表、树和高阶函数。对函数式程序设计的更广泛的原理也有所讨论。

第6章给出了论证函数式程序的形式方法。看上去似乎偏离了程序设计的主题，然而错误的程序是没用的。易于形式论证是函数式程序设计的一大好处。

第7章详细讲述了模块，包括函子（带参数的模块）。第8章讲述了ML的命令式特性：引用、数组和输入输出。本书的其余部分由较大的例子构成。第9章给出了函数式的语法分析器和一个λ-演算解释器。第10章给出了一个定理证明机，这是ML的传统应用。

书中的例子非常丰富。其中一些只是为了说明ML的某个方面，但大多数本身就有一定用途——排序、函数式数组、优先队列、搜索算法、美化打印。请注意：虽然我测试过这些程序，但是它们仍不免含有错误。

*信息和警告块。*技术性的旁白、库函数的叙述以及为进一步学习而给出的笔记都会不时地出现。它们被加以如下图标以便有些读者可以跳过：

> 亨利王的要求。他们拿不出什么理由可以反对陛下向法兰西提出王位的要求，只除了这一点，那个在法拉蒙时代制定的一条法律，In terram Salicam mulieres ne succedant，‘在撒利族的土地上妇女没有继承权’：而法国人就把这‘撒利族的土地’曲解为法兰西的土地，并且把法拉蒙认做是这条法律的创制人和妇权的剥夺者。可是他们的历史学家却忠实地宣称撒利区是在日耳曼的土地上……[⊖]

⊖ 书中的技术性旁白不会像这位大主教的演讲那么长，它有62行。
（两段译文分别出自莎士比亚《亨利五世》和《理查三世》中译本。——译者注）

ML并不完美。某些缺陷会使简单的编码错误浪费掉程序员几个小时的时间。而且，新的标准库使得新旧编译器不兼容。因此，本书中有一些这样的警告图标：

小心葛罗斯特公爵。呵，勃金汉！小心那个狗东西：要知道，摇尾的狗会咬人；咬了人，它的牙毒还会叫你痛极而死；莫同他来往，千万留意；罪恶、死亡和地狱都看中了他，地下的大小役吏都在供他使唤。

我要赶紧补充一点，在ML里不会产生这么可怕的后果。程序里的错误是不能冲垮ML系统本身的。另一方面，程序员必须牢记，即使是正确的程序也可能给外部世界带来伤害。

*如何得到Standard ML编译器。*由于Standard ML刚出现不久，很多学院没有编译器。下面列出了现有的一些Standard ML编译器，并附有联系地址。书中的例子是在Moscow ML、Poly/ML和Standard ML of New Jersey下开发的。我尚未尝试其他的编译器。

要得到MLWorks，请联系Harlequin Limited, Barrington Hall, Barrington, Cambridge, CB2 5RG, England。他们的电子邮件地址是web@harlequin.com。

要得到Moscow ML，请联系Peter Sestoft, Mathematical Section, Royal Veterinary and Agricultural University, Thorvaldsensvej 40, DK-1871 Frederiksberg C, Denmark。或从互联网上得到该系统：

http://www.dina.kvl.dk/~sestoft/mosml.html

要得到Poly/ML，请联系Abstract Hardware Ltd, 1 Brunel Science Park, Kingston Lane, Uxbridge, Middlesex, UB8 3PQ, England。他们的电子邮件地址是lambda@ahl.co.uk。或从互联网上得到该系统⊖：

http://www.polyml.org/

要得到Poplog Standard ML，请联系Integral Solutions Ltd, Berk House, Basing View, Basingstoke, Hampshire RG21 4RG, England。他们的电子邮件地址是`isl@isl.co.uk`。

要得到Standard ML of New Jersey，请联系Andrew Appel, Computer Science Department, Princeton University, Princeton NJ 08544-2087, USA。更好的是可以从互联网上得到文件⊖：

http://www.cs.princeton.edu/~appel/smlnj/

http://www.smlnj.org/

书中的程序和一些练习答案可以通过电子邮件得到，我的电子邮件地址是lcp@cl.cam.ac.uk。如果可能，请使用互联网，我的主页在

http://www.cl.cam.ac.uk/users/lcp/

*致谢。*编辑，David Tranah，在写作的各个阶段提供了帮助，并建议了书名。Graham Birtwistle、Glenn Bruns和 David Wolfram仔细阅读了文本。Dave Berry、Simon Finn、Mike Fourman、Kent Karlsson、Robin Milner、Richard O'Keefe、Keith van Rijsbergen、Nick Rothwell、Mads Tofte、David N. Turner和Harlequin的工作人员也对文本提出了意见。Andrew Appel、Gavin Bierman、Phil Brabbin、Richard Brooksby、Guy Cousineau、Lal George、Mike Gordon、Martin Hansen、Darrell Kindred、Silvio Meira、Andrew Morris、Khalid Mughal、

⊖⊖ 这两个网址原书没有，中译本加上的。——译者注

Tobias Nipkow、Kurt Olender、Allen Stoughton、Reuben Thomas、Ray Toal和Helen Wilson发现了前几次印刷中的错误。 Piete Brooks、John Carroll和Graham Titmus在计算机使用方面给予了帮助。我还要感谢Dave Matthews开发了Poly/ML，这是多年以来唯一高效的Standard ML的编译器。

在众多的参考文献中，Abelson和Sussman（1985）、Bird和Wadler（1988）以及Burge（1975）的著作特别有帮助。Reade（1989）的书中包含了在ML中实现惰性表的有用思想。

The Science and Engineering Research Council在过去20多年来给予了LCF和ML大量的研究资助。

本书的大部分写作工作都是我从剑桥大学休假的过程中完成的。我感谢计算机实验室（Computer Laboratory）和卡莱尔学院（Clare College）给予休假，以及爱丁堡大学对我六个月的招待。

最后，我要感谢Sue，感谢她所给我的一切帮助，以及天天耐心倾听我关于每一章进展的报道。

目　录

第1章 Standard ML

第一个ML编译器是在1974年实现的。随着用户群的成长，各种语言的变体开始出现。于是，ML社群便集中起来开发和普及一种通用的语言，Standard ML，有时简称为SML，或干脆就叫ML。现在可以找到一些很好的Standard ML编译器。

没用多久Standard ML就变得相当流行。世界各地都有大学采用它作为教授学生的第一门程序设计语言。开发者选择它作为一些具有相当规模的项目的实现语言。也许对于这种流行的初步解释是：用ML很容易写出清晰、可靠的程序。要得到更满意的答案，首先要分析一下我们是怎样看待计算机系统的。

计算机是非常复杂的。在一部典型的工作站里面所包括的硬件和软件就远不是一个人所能完全掌握的。不同的人对工作站有不同层次的理解。对于一般的用户，工作站是一部文字处理机或电子表格。对于维修工来说，工作站是一个盒子，里面有电源和电路板等。对于机器语言程序员，工作站则提供了一个巨大的字节存储器，连接在一个能够进行算术和逻辑运算的处理器上面。而应用程序员则会借助他所选择的程序设计语言作为媒介来理解工作站。

这里，我们把“电子表格”、“电源”和“处理器”都理解为理想的、抽象的概念。我们只考虑它们的功能和局限，却不关心它们是如何构造的。通过良好的抽象，我们可以有效地使用计算机，而不会被它的复杂性拖垮。

普通的高级程序设计语言在抽象的层次上并没有比机器语言高多少。这些语言提供了方便的记号，但仅限于那些可以被直接映射到机器码上去的操作。程序中一个小的错误可以破坏其他数据，甚至是程序本身。这种行为的结果若能解释的话也只能是在机器语言的层次上进行。

ML则高出机器语言层很多。它支持*函数式程序设计*（functional programming），其中，程序是由函数所组成的，这些函数操作简单的数据结构。函数式程序设计在许多方面对于解决问题都是非常理想的，下面会对这个问题进行简单的讨论，也会贯穿全书来进行演示。程序设计任务可以通过数学方式来达成，而不用事先知道计算机的内部工作情况。ML也提供*可变的*（mutable）变量和数组。可变的对象可以通过赋值语句来改变，利用这些，可以很容易表达任何传统的代码。为了构造大的系统，ML提供了*模块*（module），程序的某一部分可以分别描述和编码。

最为重要的是，ML可以防止程序员犯错误。在程序可以运行之前，编译器会检测所有模块的接口是否彼此相容，以及所有的数据是否被一致地使用。例如，一个整数不会被用作存储地址。（一个真正的程序不应该依赖于这种技巧。）在程序运行中，更进一步的检测保证了安全性：甚至是一个错误的ML程序也会表现得像一个ML程序。它可能会永远运行下去，或者返回一个错误信息给用户。但是它不会崩溃。

ML支持一种面向程序员要求的抽象层次，而不是面向硬件的。ML系统可以保证这种抽象，即使程序是错误的。其他的程序设计语言很难提供这种保障。