第13章 操作系统设计

在过去的12章中，我们讨论了许多话题并且分析了许多与操作系统相关的概念和实例。但是研究现有的操作系统不同于设计一个新的操作系统。在本章中，我们将简述操作系统设计人员在设计与实现一个新系统时必须要考虑的某些问题和权衡。

在系统设计方面，关于什么是好，什么是坏，存在着一定数量的民间传说在操作系统界流传，但是令人吃惊的是这些民间传说很少被记录下来。最重要的一本书可能是Fred Brooks的经典著作*The Mythical Man Month*（中文译名《人月神话》）。在这本书中，作者讲述了他在设计与实现IBM OS/360系统时的经历。该书的20周年纪念版修订了某些素材并且新增加了4章（Brooks, 1995）。

有关操作系统设计的三篇经典论文是"Hints for Computer System Design"（计算机系统设计的忠告，Lampson, 1984）、"On Building Systems that Will Fail"（论建造将要失败的系统，Corbató, 1991）和"End-to-End Arguments in System Design"（系统设计中端到端的论据，Saltzer等人，1984）。与Brooks的著作一样，这三篇论文都极其出色地经历了岁月的考验，其中的大多数真知灼见在今天仍然像文章首次发表时一样有效。

本章吸收了这些资料来源，另外加上了作者作为三个系统的设计者或合作设计者的个人经历，这三个系统是：Amoeba（Tanenbaum等人，1990）、MINIX（Tanenbaum和Woodhull, 1997）和Globe（Van Steen等人，1999a）。由于操作系统设计人员在设计操作系统的最优方法上没有达成共识，因此与前面各章相比，本章更加主观，也无疑更具有争议。

13.1 设计问题的本质

操作系统设计与其说是精确的科学，不如说是一个工程项目。设置清晰的目标并且满足这些目标非常困难。我们将从这些观点开始讨论。

13.1.1 目标

为了设计一个成功的操作系统，设计人员对于需要什么必须有清晰的思路。缺乏目标将使随后的决策非常难于做出。为了明确这一点，看一看两种程序设计语言PL/I和C会有所启发。PL/I是IBM公司在20世纪60年代设计的，因为在当时必须支持FORTRAN和COBOL是一件令人讨厌的事，同时令人尴尬的是，学术界背地里嚷嚷着Algol比这两种语言都要好。所以IBM设立了一个委员会来创作一种语言，该语言力图满足所有人的需要，这种语言就是PL/1。它具有一些FORTRAN的特点、一些COBOL的特点和一些Algol的特点。但是该语言失败了，因为它缺乏统一的洞察力。它只是彼此互相竞争的功能特性的大杂烩，并且过于笨重而不能有效地编译。

现在考察C语言。它是一个人（Dennis Ritchie）为了一个目的（系统程序设计）而设计的。C语言在所有的方面都取得了巨大的成功，因为Ritchie知道他需要什么，不需要什么。结果，在面世几十年之后，C语言仍然在广泛使用。对于需要什么要有一个清晰的洞察力是至关重要的。

操作系统设计人员需要什么？很明显，不同的系统会有所不同，嵌入式系统就不同于服务器系统。然而，对于通用的操作系统而言，需要留心4个基本的要素：

1) 定义抽象概念。
2) 提供基本操作。
3) 确保隔离。
4) 管理硬件。

下面将描述这些要素。

一个操作系统最重要但可能最困难的任务是定义正确的抽象概念。有一些抽象概念，例如进程和文