C 語言的誕生，竟然是一個失敗的項目?

博文視點Broadview 昨天

以下文章來源於CSDN，作者Carol

整理| Carol

出品| CSDN（ID：CSDNnews）

很多人認為，C 語言是一門“古董”語言。也有不少人認為，它沒有Python 簡潔，沒有Java 安全，甚至有可能要退出歷史舞台。而事實上，時至今日，C 語言憑藉其在不同編譯環境的穩定性、可移植性、快速的運行速度，仍在多個領域發揮著重要作用。

首先，Unix 就是用C 語言編寫的。雖然最初Unix 採用的是彙編語言，但是Unix 早在1973 年就用C 語言進行重寫，這使得Unix 在不同的機器上更具可移植性，並有助於其變得流行。如果沒有重寫的Unix，那麼我們今天使用的所有操作系統——Linux，MacOS X，Android，iOS，Chrome OS 都可能不會存在。

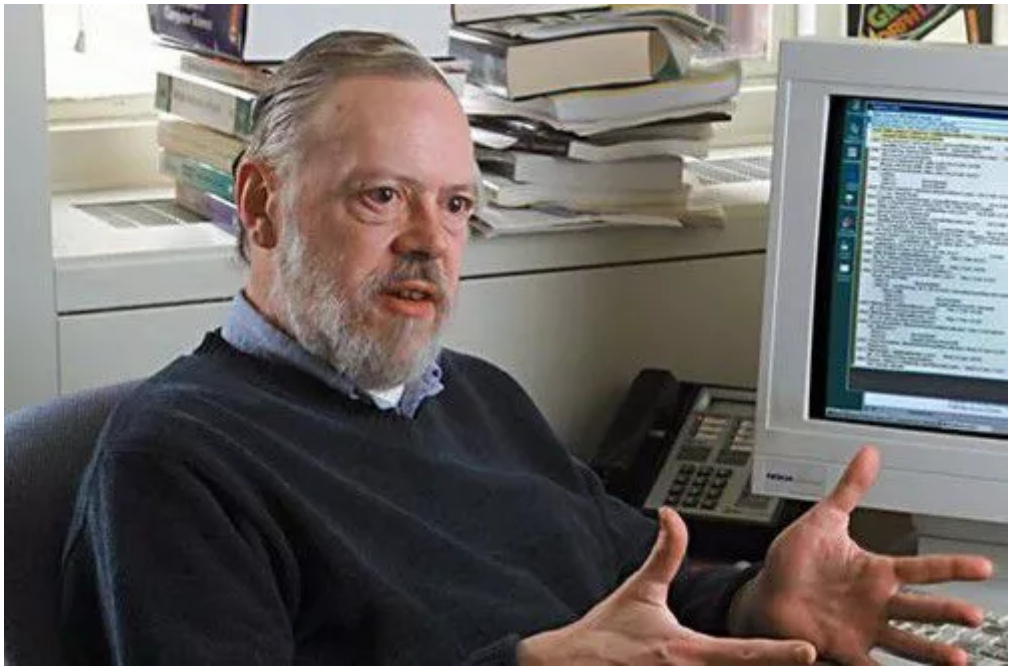
而除了操作系統以外，Oracle 數據庫、MySQL 等數據庫管理系統也都是由C 語言編寫。雖然他們中的大多數數據庫後來又用C++重寫，但這也代表它們都是C 的直系後裔。

即使你一直使用的是Python， 那你也從未停止使用過C 語言：如CPython 。CPython 是特指C 語言實現的Python，也就是最原始版本的Python。當我們從Python 官方網站下載並安裝好Python 後，直接獲得了一個官方版本的解釋器：CPython。這個解釋器是用C 語言開發的，所以叫CPython。在命令行下運行Python 就是啟動CPython 解釋器。CPython 是使用最廣的Python 解釋器。教程的所有代碼也都在CPython 下執行。

之所以使用CPython 這個詞，是因為Python 語言從規範到解釋器都是開源的，任何人都可以通過編寫Python 語言解釋器，比如Jython，就是Java 版的Python，還有燒腦的PyPy，則是使用Python 把Python 再實現了一遍。

簡而言之，C 語言簡直是無處不在。但是被廣泛應用的C 語言的誕生，卻不是輕易就成功的。相反，它是一個長期失敗的產物。

在業內，大家所熟悉的、被稱為"C 語言之父"的或許是這位：Dennis Ritchie。



Dennis Ritchie（丹尼斯·里奇），同時也是Unix 之父

當然，Dennis Ritchie的成就是不可磨滅的，他將C語言帶到世界面前，其影響力在今日依舊巨大。Dennis Ritchie的合作夥伴Brian Kernihan（布萊恩·克尼漢）曾這樣評價他：牛頓說他是站在巨人的肩膀上，如今，我們都站在Ritchie的肩膀上。

但今天的主人公卻並不是Ritchie。因為在Ritchie 之前，曾有一位學校教師，如果非要追溯源頭，那麼他應該才算是創造C 語言的第一人。

可以這麼說，如果這個世界上沒有這位喜歡在假期裡編程的教師，可能世界上也就不會有C 語言。

1

一位和艾倫• 圖靈是朋友的教師

今天的主人公，就是 Christopher Stratchey（克里斯托弗·斯特拉奇），Strachey 于 1916 年出生于英格兰一个家境比较显赫的家庭，也曾在诺福克格雷沙姆学校和剑桥国王学院接受教育。

1938 年 10 月，是他在剑桥国王学院学习的第四年，面临毕业的 Strachey 却似乎由于长期忽视学习，仅从及格分数线上低空飘过，惊险地获得了毕业证。虽然顺利毕业了，但这样的结果也令他想要获得研究奖学金的希望化为泡影。

即使 Strachey 家庭条件还不错，也还是要面临来自现实的压力。1939 年 8 月，为了养活自己，Stratchey 接受了 STC (Standard Telephones and Cables Limited) 的物理学家职位，拿着平均每周 4 英镑的酬劳，开始在伦敦 STC 的开发实验室工作。

在 STC 期间，Strachey 大部分时间都在研究厘米雷达阀的理论设计，主要是推导出阀门参数的分析公式和它的实验验证。他的数学工作涉及微分方程的积分，其中一些特别棘手，因此，Strachey 与同事开始使用微分分析器获得数值解。

后来，Strachey 开始把这次使用计算机的经历看作是一个转折点，他对计算机的兴趣在这个时候才被激发起来。

1944 年 7 月，在 STC 已经工作了 5 年的 Strachey 突然被调往伦敦的 STC 无线电部门工作，这是一项与电气和机械设计有关的工作。突如其来的调动让 Strachey 很不适应，同时他也发现，这活儿比理论工作更不合自己的口味，而且认为这种氛围“相当狭隘和肮脏”。

确定了内心的想法后，Strachey 迅速从 STC 辞职。一年后，也就是 1945 年 10 月，他开始在英国圣埃德蒙进修"physics-cum-mathematics" 硕士学位。

从圣埃德蒙毕业之后，通过多次申请，他终于在英国哈罗公学（Harrow School，英国历史悠久的著名公立学校之一）获得了一个教师职位，并于 1949 年春天离开了圣埃德蒙学院。

Strachey 于 1949 年 9 月开始在哈罗学校任教，在校任职期间，Strachey 经常与组织社团和俱乐部一起表演、搞活动，由于在圣埃德蒙时自学过巴松管，Strachey 还加入了学校的管弦乐队。可以说，在这样忙碌的“教学”氛围中，Strachey 对计算的兴趣在离开 STC 后基本上处于休眠状态了。

他第一次接触存储程序计算机是在 1951 年 1 月，当时一个共同的朋友向他介绍了国家物理实验室 (NPL) 的迈克·伍德格(Mike Woodger)，由于这一场相识，让之前沉迷于文艺表演的 Stratchey 在国家物理实验室的 Pilot ACE 上度过了他的学校假期。值得一提的是，这是第一台配备 Alan Turing 自动计算引擎的计算机。

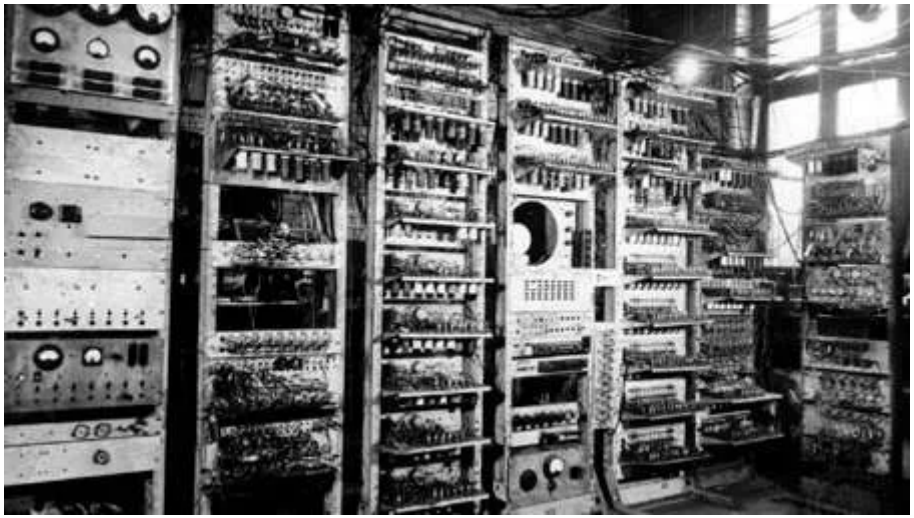
Stratchey 在这个特殊的假期里开始了他的研究，而他的目的是：希望教会计算机如何下棋。这在计算机主要用于快速求解方程的时代是一件匪夷所思的事。

然而，由于 Pilot ACE 没有完成这项工作所需的存储容量，他的第一次尝试没有成功。第二年春天，Stratchey 学习了刚安装在曼彻斯特大学的费兰蒂 Mark I 计算机。这台机器的存储空间比 Pilot ACE 大得多，相应地，Strachey 的编程范围也更大。更巧的是，Alan Turing （艾伦·图灵）当时正是曼彻斯特大学计算机实验室的助理主任，并为这台机器编写了程序员手册。

而 Strachey 恰好是 Alan Turing 在国王学院的老朋友，顺理成章地，Strachey 获得了一份程序员手册副本。

1951 年 7 月，Strachey 第一次访问了Manchester Mark I Computer（第一台存储程序数字计算机）。当 Strachey 解释了他“希望教会计算机如何下棋”的想法时，Alan Turing 表示这个想法很有趣，并提出另一个可能：让机器模拟自己，并以这种方式为剑桥大学的 EDSAC

（电子延迟存储自动计算器，Electronic Delay Storage Auto-matic Calculator）开发解释性跟踪例程。



Manchester Mark I ，图片来源：britannica.com

Strachey 被这个想法吸引了，暂时将原本的计划搁置一边，先着手实现 Alan Turing 提出的这个可能性。最终，Strachey 设计的跟踪程序大约有 1,000 条指令，这是迄今为止为机器编写的最长的程序。

通过上面的叙述，相信大家对 Strachey 的计算机水平已经有了一定的了解，那么 Strachey 的故事先介绍到这里，接下来向读者介绍 3 个新的人物。

2

三个 David 想要一种新语言

60 年代，剑桥大学准备购入一台新电脑。剑桥想为这台全新的计算机设计一个新的操作系统，而另外三位剑桥大学的研究员 David Hartley, David Wheeler 和 David Barron 则有更进一步的想法。三位 David 想要创造一门全新的编程语言，并由此来开发新的操作系统。

开发一门新语言，这一听就非常有趣！自信的大卫们认为自己可以轻松创造出更好的语言，因此并没有做好前期的需求调研工作。他们没有向未来的用户询问旧语言的优缺点，也没有做好项目规划，就这样开工了。

他们还为这门即将诞生的新语言起了一个响当当的名字 CPL，Cambridge Programming Language 的缩写。多年以后，Hartley 指出，试图创造一种新语言是"一个愚蠢的想法"。

如果决定开发一种新语言是一个愚蠢的想法，那么选择让 Strachey 来监督这个项目更是"一个愚蠢的决定"。虽然他的计算能力不容置疑，但他似乎不是一个好的项目经理，他对这个项目十分热爱，以至于根本无法确定优先级。因此，这个由 Strachey 带领的开发团队只关注开发过程中的小问题，有决定性价值的大问题却无人问津。很快，CPL 就被戏称为 Christopher 的编程语言。在 Stratchey 的坚持下，团队专注于设计 CPL 的所有小细节，导致 CPL 变得过于复杂，难以实现。甚至当他们想为这门语言编写一个编译器时，发现生成的机器代码效率过于低下，导致他们不得不放弃。

但这个故事还没有结束。

3

C语言成功了！

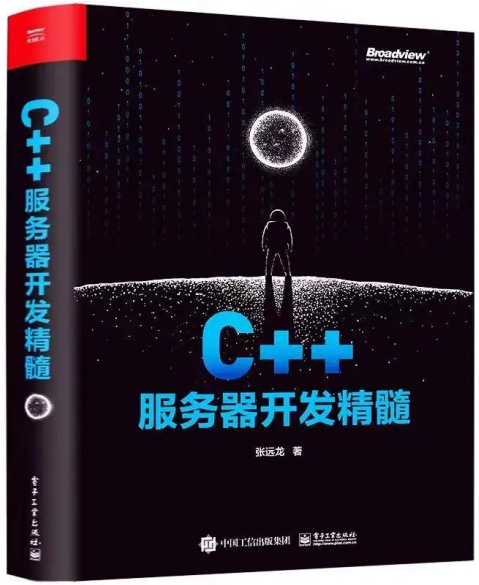
1967 年，Martin Richards 加入了 CPL 团队并开始着手简化它。当然，那个时候他的目标是获得一些能够产生良好编译器和高效机器代码的东西。这就是 Richards 从 CPL 中开发 BCPL — Basic CPL 的初衷，在某种程度上，这可以算得上是一种新语言——这也就意味着承认旧 CPL 惨遭失败。

与此同时，贝尔实验室（AT&T Bell Laboratories）的研究员 Kenneth Lane Thompson 闲来无事，手痒难耐，想玩一个他自己编的，模拟在太阳系航行的电子游戏——Space Travel。他背着老板，找到了台空闲的小型计算机——PDP-7。但这台电脑没有操作系统，而游戏必须使用操作系统的一些功能，于是他着手为 PDP-7 开发操作系统。后来，这个操作系统被命名为——UNICS（Uniplexed Information and Computing Service）。

1969 年，Thompson 以 BCPL 语言为基础，设计出很简单且很接近硬件的 B 语言（取 BCPL 的首字母），并且用 B 语言写了初版 UNIX 操作系统（叫 UNICS），两年后，同样酷爱 Space Travel 的 Richards 为了能早点儿玩上游戏，加入了汤普森的开发项目，合作开发 UNIX。他的主要工作是改造 B 语言，使其更成熟。

1972 年，Richards 在 B 语言的基础上最终设计出了一种新的语言，他取了 BCPL 的第二个字母作为这种语言的名字，这就是 C 语言。

C 语言终于诞生了！C 最终以这样奇妙迂回的方式成功了。虽然前教师 Strachey 把很多事情复杂化了，从而开始了一连串的失败。但是没有这些，C 甚至可能还没有被发明出来。



《C++服务器开发精髓》

张远龙 著

- 从操作系统原理角度讲解C++服务器开发技术栈
- 内容详尽细致、版本新
- 重磅级C++服务器开发红宝书

本书详细讲解如何掌握C++服务器开发技术，以及如何成为合格的C++开发者，秉承的思想是，通过掌握技术原理，可以轻松制造“轮子”，灵活设计出优雅、鲁棒的服务，并快速学习新技术。

无论是对于C/C++开发者、计算机专业的学生，还是对于想了解操作系统原理的读者，本书都极具参考价值。

（京东限时活动，快快扫码抢购吧！）

如果喜欢本文

欢迎 **在看** | **留言** | **分享至朋友圈** 三连

热文推荐

- 这本书，让我秒懂了微服务架构
- 一套小白也能看懂的算法书！
- 书单 | 打好算法基础，深入AI实战！
- Flutter企业级应用开发方案

阅读原文