

软件项目管理系列丛书

Software Runaways

Lessons Learned from Massive Software Project Failures

软件开发的滑铁卢

重大失控项目的经验与教训

(美) 罗伯特·格拉斯 著

陈河南 等 译



電子工業出版社
Publishing House of Electronics Industry



培生教育出版集团

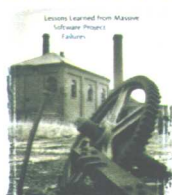
失控的软件项目案例与分析

Lessons Learned from Massive Software Project Failures

本书提供了 17 个惨痛的软件灾难案例：

- 美国航空公司失败的机票预订系统 CONFIRM;
- 美国机动车协会新泽西分部的 4GL 灾难;
- 几乎摧毁了客户关系的 NCR 库房管理系统;
- 失败的下一代 FAA 空中交通控制系统;
-

SOFTWARE RUNAWAYS



Robert L. Glass

无论是政府信息化推进部门、IT 主管、项目经理，还是开发人员，通过本书，都可以汲取别人所犯错误的教训，从中获得启发，而且，您不必亲身体验实际犯错误的痛苦经历。

ISBN 7-5053-7371-4



9 787505 373716 >

世纪波文化发展有限公司

咨询电话: (010)68132199



策划编辑: 傅豫波

本书贴有激光防伪标志，凡没有防伪标志者，属盗版图书。

ISBN 7-5053-7371-4/TP · 4252

定价: 20.00 元

软件项目管理系列丛书

Software Runaways

Lessons Learned from Massive Software Project Failures

软件开发的滑铁卢

重大失控项目的经验与教训

(美) 罗伯特·格拉斯 著

陈河南 等 译

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

Software Runaways: Lessons Learned from Massive Software Project Failures, First Edition by Robert L. Glass.

Copyright ©1998 by Prentice Hall PTR.

All rights reserved. No any part of this book may be reproduced or transmitted in any form and any means.

Chinese Simplified language edition published by Publishing House of Electronics Industry, Copyrights 2002.

本书由美国培生教育出版集团公司授权电子工业出版社翻译出版, 未经出版者书面许可, 不得以任何方式复制和抄袭本书任何部分。

版权所有, 侵权必究。

版权贸易合同登记号 图字: 01-2001-5509

图书在版编目(CIP)数据

软件开发的滑铁卢——重大失控项目的经验与教训/(美)格拉斯(Glass, R.L.)著; 陈河南等译.—北京: 电子工业出版社, 2002.2

(软件项目管理系列丛书)

书名原文: Software Runaways: Lessons Learned from Massive Software Project Failures

ISBN 7-5053-7371-4

I. 软… II. ①格… ②陈… III. 软件开发-项目管理-经验-美国 IV. F471.266

中国版本图书馆CIP数据核字(2001)第093278号

策划编辑: 傅豫波

印刷者: 北京大中印刷厂

出版发行: 电子工业出版社 <http://www.phei.com.cn>

北京市海淀区万寿路173信箱 邮编: 100036

经 销: 各地新华书店

开 本: 850×1168 1/32 印张: 10.125 字数: 180千字

版 次: 2002年2月第1版 2002年2月第1次印刷

定 价: 20.00元

凡购买电子工业出版社的图书, 如有缺页、倒页、脱页者, 请向购买书店调换, 若书店售缺, 请与本社发行部联系。联系电话: (010) 68279077

向着软件产业国际化迈进

——《软件项目管理系列丛书》总序

软件产业是一个朝阳的、新兴的知识型产业。一个国家软件业的发达程度，也体现了国家的综合国力，决定着国家未来的国际竞争地位。

目前，中国的软件企业正处于高速发展、急需规范管理并以项目为主导的环境中。企业每天所面对的不仅仅是几个越来越大的大型项目，而将是成百上千不断发生和进行的项目。产生这种变化的因素是多方面的，这包括客户需求的不断提高导致产品生命周期缩短；产品开发项目数量大增；新技术导致了对研究和开发项目需求的增加；为了提高业务赢利能力，改进业务模式的项目需求大增等。在这种多项目并发、技术含量高、变化速度快、资源有限的环境下，如何对企业、项目、资源实施科学的管理，加强团队能力，实现软件企业的生产规模化、规范化、国际化，是当前我国软件业面临的最大挑战。

一些调查表明，大约70%的软件开发项目超出了估算的时间，大型项目平均超出计划交付时间20%到50%，90%以上的软件项目开发费用超出预算，并且项目越大，超出项目计划的程



APIS 10/01

目 录

第1章 简介	1
1.1 什么是软件失控项目	2
1.2 软件危机的呼声	5
1.3 “两难境地”和“死亡行军”项目	9
1.4 相关的研究与发现	13
第2章 失控的软件项目案例	23
2.1 没有指定完整的项目目标	25
2.1.1 案例一：BAE自动化系统(A)——丹佛国际 机场行李处理系统	27
2.1.2 案例二：BAE自动化系统(B)——丹佛国际 机场行李处理系统的部署	57
2.1.3 案例三：佛罗里达福利救济系统的惨败	62
2.1.4 案例四：失控项目的剖析——什么使AAS 搁浅	67
2.1.5 总结：一些大型计算机项目失控的原因	79
2.2 拙劣的计划和评估	108
2.2.1 案例五：痛苦的诞生——创建新软件成为 ON Technology 公司极其痛苦的任务	109
2.2.2 案例六：来自地狱的项目	121
2.3 采用新技术	128
2.3.1 案例七：遭遇失败	131
2.3.2 案例八：智能电子公司认识到新技术	

软件开发的滑梯卢

Software Runaways

隐藏的危险	143
2.3.3 案例九: 4GL灾难剖析	148
2.3.4 案例十: 威斯特帕克银行项目失控的剖析	168
2.4 缺乏或根本不具备项目管理方法	176
2.4.1 案例十一: 每年花费纳税人500亿美元的 IRS项目的失败	178
2.4.2 案例十二: 无路可走的政府机构	189
2.4.3 案例十三: 美洲银行的MasterNet系统—— 风险评估的案例研究	192
2.5 团队中缺少资深人员	225
2.5.1 案例十四: 无法控制的突发性灾难	228
2.5.2 案例十五: 专业标准变得松懈—— CONFIRM的失败及其教训	232
2.6 硬件/软件供应商的低劣表现	259
2.7 其他——性能(效率)问题	260
2.7.1 案例十六: NCR库存管理系统实际上成 为了怠工者	263
2.7.2 案例十七: Lisp错误毁坏了MCC CAD项目	274
第3章 软件失控项目的补救措施	279
3.1 风险管理	280
3.2 问题管理	288
3.3 失控项目的补救措施	292
3.4 为将来预留的补救措施	298
第4章 结论	303

第1章

简 介

人类只能够在第二级追求中取得成功，比如说，成为百万富翁或者总理、赢得战争、飞跃同温层或者登陆月球。第一级追求，不可避免地要涉及到努力去理解生命的意义以及努力去转化这种理解，这就不可避免地会导致失败感的产生。拿破仑、邱吉尔、罗斯福式的人物可以感受自己的成功感，但苏格拉底、帕斯卡、布莱克式的人物永远不会有这种感觉。对生命意义的理解永远是遥不可及的事情。在从事这种探索时势必蕴藏着失败，这种探索仍然只是一件值得认真关注的事情。

—— 马尔科姆·马格里奇 (Malcolm Muggeridge)
引自他播出的“妇女时空”广播节目，1965年8月5日，后来被广为引用，包括埃德·扬顿 (Ed Yourdon) 的《死亡行军》(Death March) 一书。

本书的绝大部分是由火药味十足的案例组成。这些案例对软件失控项目进行了深入而有趣的分析，其中一些较为著名的案例主要摘自有关文献。这些文献包括大众出版物的报告（如《华尔街日报》）、计算机媒体的报告（如《信息周刊》）以及学术报告（如《哈佛案例研究》）。但是在触及这些充满火药味的案例之前，还需要有一些介绍性的内容。我们将会在1.1节中回答“什么是软件失控项目（software runaway）”这样的定义性问题；在1.2节中，我们将会论述有关软件开发实践正处于危机关头的说法；而在1.3节中，我们将会讨论一些相关但是不同的观点——项目处于“两难境地”（Crunch Mode）时出现的情况以及构成“死亡行军”^①项目的内容。在本章的最后，我们还会展示在探索软件失控项目过程中一些饶有趣味的发现，并从各个方面做出重要的结论。

1.1 什么是软件失控项目

媒体常常大肆张扬地报道由软件导致的系统崩溃。他们这样说：“XYZ公司在生产SYS系统时落后进度表M个月，而预算也超出了D元，看来取消这个项目已经不可避免了，这总共花费了该公司BUX美元。”当你看到这些M、D、BUX时，都是用

^① Death March 的原意是指战俘等被迫进行艰苦的长途步行——译者注。

很大的数字表示的，就是后边有很多个零的那种。

本书的主要突破点是要展示一些XYZ和SYS的案例，包括它们是如何糟糕到M和D的水平以及真正损失了多少BUX。正如你将在这些案例中看到的，XYZ代表一些知名公司，而SYS则包括一些频繁地见诸于媒体的项目，它们的进展超出了开发者所能想像或希望的程度。

但是在研究这些案例之前，正如软件业界所说的，要进行一些初始化工作，也就是说，将这些案例引入到总体的环境中至关重要。设置这些环境就是本书以下这些初始化、介绍性的小节所起的作用。

首先，让我们来定义一下什么是软件失控项目。

软件失控项目就是由于在创建系统所需软件时遇到困难，从而导致大大超出可控制范围的项目。

“项目失控”暗示着项目变得无法管理，从而无法达到最初制定的目标，甚至无法接近目标。如果从进度表的角度考虑这些目标（时间是最可能出现超出控制范围的方面），软件失控项目所用的时间接近预计所用时间的两倍甚至更多；如果目标为支出费用，那么经过认真核查预算，这些项目通常也会超出费用指标，它们会消耗接近预计费用的两倍甚至更多；如果目标是满足其功能性需求的可靠产品，那么这些产品不能满足那些目标，实际上，这些项目将不得不取消，因为它们常常是根本

无法满足任何目标。

我们的定义与某种程度上更为著名的定义有所不同。

KPMG的定义是：

软件失控项目是显著未能实现目标和（或）至少超出原定预算30%的项目【KPMG 1995】。

可以看出，KPMG对失控项目的定义比我们的范围更广，也就是说，如果一个项目超出费用指标30%或者更多，按照KPMG的定义来说就是失控项目，但是根据我们的定义则必须超出预算100%。使用KPMG的定义将会有比使用我们的定义更多的失控项目。这里提到KPMG的定义，完全因为该定义出自目前我们所知的惟一有关失控项目的研究报告，是已知的与我们的定义不同的量化表示。在1989年和1995年，KPMG进行了两次对失控项目的调查，以确定它们出现的频率、产生的原因、使用的补救措施以及对其企业的影响。我们将会在本章后边的小节中展示这项研究的发现。

为什么我们对于失控项目的定义如此大惊小怪？原因是人类进入软件领域只有40多年。在早期阶段，创建软件系统最常见的问题不是软件系统本身构造的问题，而是估计构造费用的问题。为什么会有这种估计的问题呢？因为从前人们对软件领域项目的费用没有进行过认真的研究。构造软件是一项极其复杂的任务——有人说这是人类从事过的最复杂的任务。

因为历史纪录的缺乏和任务的复杂性，从而产生了一个障碍，一个任何数学方法和专门技能或者单个专家都无法逾越的障碍：软件项目超出费用和进度表目标实在太普遍了，因为那些目标本身就完全是错误的。

我们认为项目失控是由那些由于比错误的估计更深层次的原因导致的。这就是我们提高了KPMG的30%界限的原因。我们想要确定，我们之所以把那些项目称做软件失控项目，是由于开发工作本身失去了控制，而不管最初的费用估计是否接近实际数字。很多时候，超出成本目标30%的项目不会成为有问题的或者失败的项目。所以，即使KPMG的研究包括了它们，我们也不想在本书中包括这类项目。

1.2 软件危机的呼声

过去的10年里，在计算机文献，有时甚至在大众媒体中经常能够看到“软件危机”（software crisis）的说法。虽然不同的作者似乎用这个术语来意指不同的事物，但“软件危机”最常见的定义是：

软件总是超出预算、落后于进度表，而且不可靠。

但是对我来说，由此着手开始讨论是十分重要的，因为

我不相信存在软件危机。

在本书的开始提出这个观点很重要，因为不这样做的话很容易得出这种结论：关于本书的作者认为软件失控是软件危机的症状和范例。这样就离真相越来越远了。

当然，存在很多软件失控的现象，本书就将讲述这些案例，但是这些失控的软件项目案例只代表所有业经尝试过的软件项目中的很小的一部分。

值得注意的是，那些确信软件危机的人就失控项目出现的频率提供了很多估计。但是这些估计都有一个基本的问题，即他们所包括的范围大相径庭！根据GAO（Government Accounting Office，美国政府审计局）的研究，最著名的统计数字是有98%以上的项目都失败了（“只有不到2%的合同订购软件在发布时具有可用性”）。但是这些数字存在一个严重的问题，即政府审计局研究的是一些有麻烦的项目（这就是政府审计局对其进行研究的原因），因此得出98%的软件项目有问题并最终将导致失败这样的结论就没什么可惊奇的了。欲知对误用政府审计局数字的驳斥，请参见【Blum 1991】。

很多作者和演讲者都重复了政府审计局的结论，根本没有想到政府审计局研究的对象与他们的观点不同。但是，即使那些没有重复98%这个数字的人也得出了软件开发项目失败的比例很大的结论，我曾经在不同的文献中看到像60%、40%、35%

这样的数字。我个人认为在不可相信政府审计局98%数字的同时，同样没有理由相信这些数字。问题的症结在于，没有人真正进行过充分地调查来告诉我们有多大比例的软件项目失败了。实际上，人们几乎无法对失败得出适当的定义，但准确地进行定义，又是在进行有意义的调查之前，理所当然必不可缺的一环。请注意本书在定义“失控项目”时遇到的困难，由于“失控”在某种意义上更为引人注目，其实它是比“失败”更容易定义的术语。

那些软件危机的呼声以及具有欺诈性的量化数字尤其令我愤慨，其原因有二。

(1) 软件危机的呼声中有一种暗示，即软件从业人员从骨子里都是妄自尊大的，他们的编程作业仅限于书面形式。

(2) 还有一种暗示，即软件的实际应用通常充满了失败，只有很少的应用能够成功。

我本人坚信这些暗示都是彻头彻尾的错误。看看四周，我看到的是一个计算机软件不可或缺的世界。计算机和软件处理我的机票预订，控制我的银行事务，把人类送到太空——其可靠性毋庸置疑。

那么，为什么有那么多的人在叫喊着软件危机呢？因为这样做有利可图。某些经销商叫喊危机是为了卖出他们声称能够提供对策的产品或者服务；某些研究者叫喊危机是为了获得他

们声称同样（最终）将提供对策的研究项目的经费；某些学术界人士叫喊危机是为了促使人们接受并阅读他们提出对策的专业论文。

实际上关于危机的呼声有不少可笑之处（如果提出严肃暗示的问题可以称为可笑的话）。大多数叫喊着危机的人都在试图出售或者推广某种为创建软件提供更好技术的东西。但是大多数软件失败的案例研究表明，产生问题的根源在于低劣的管理，而不在于低劣的技术。因此，即便存在着软件危机，这些人所提供的东西不会也不能解决问题。

实际上，上述提及的KPMG的研究发现以及本书中我们自己的发现将会说明，最近以来，技术日益成为软件项目失败更为常见的原因。但是真正具有讽刺意味的是，事实上，这些基于技术的失败大多正是那些称为解决方案的技术所导致的。为了达到他们的目的，这些人首先叫喊着软件危机，并对这些技术的益处做出过分而错误的声明。在本书后面有由于使用这些正规方法、专家系统以及第四代语言而导致项目失控的案例。这些技术不仅远远没有成为对策，有时反而成为沉重的负担，并将项目拉向失败。

对危机的叫喊，某种程度上表现了“恶人先告状”的心理。我们已经提到过，软件项目未能达到费用和进度表的目标，经常是因为这些目标本身就是错误的。这样，软件从业人员辛辛

苦苦地工作，其实是去实现不现实的目标，尽管实现目标希望渺茫，还是要反复耗费大量时间去迎合目标，最终由于从项目开始便无法控制的问题而受到谴责。对软件估算做出的实际调查表明，最为常见的是由营销人员或者客户来制定费用目标和进度表，其次是由管理人员制定，很少有实际完成项目的技术人员涉足其中。由于无法掌握自己的命运，在出现问题时这些实践者们总是受到指责。

本书写到此处，我希望我们大家暂停片刻，表达一下感激之情。正是所有那些奋勇克服前进道路上障碍的软件实践的先驱者们，才使我们的时代真正成为计算机时代，用半个世纪左右的时间使计算机技术成为我们社会的主导力量。让我们对他们的努力表示感谢，并赞美他们的成功。

1.3 “两难境地”和“死亡行军”项目

丰富多彩的术语成为计算机业界人士追求的时尚。我们毫不犹豫地发明新术语和新缩写。这种现象泛滥成灾，甚至为术语的浪费而发明了一个术语——“计算机语”（Computerese），我们新创这个词表示计算机业界人士讨论事情时使用的专门语言。

“失控项目”就是一个这类的术语，“软件危机”（software crisis）也是其中的一个。在本小节中，我将会讨论另外两个相

关的术语——“两难境地”（Crunch Mode）和“死亡行军”（Death March）项目。

“两难境地”项目是约翰·博比（John Boddie）在他的同名书中使用的术语【Boddie 1987】，是描述项目状态的术语。处于两难境地的项目面临着无法达到最初目标的威胁（费用、进度表、功能性等等），而项目团队在努力想要跨越该困境。“我们正处于两难境地，在半夜之前我是不会回家的。”软件专家会在办公室中给家人这样打电话。家人也不会吃惊。两难境地的状况可能会持续几天、几周、甚至是几个月，这取决于项目本身持续的时间以及它偏离目标的程度。“两难境地”这个术语不是博比的发明，而是由发现自己被迫完成项目的软件从业者们发明的。

“死亡行军”项目是埃德·扬顿在他的同名书【Yourdon 1997】中使用的术语，也用于描述项目状态。如果项目的各项参数超出正常标准至少50%（参数可能包括进度、人员、预算以及功能性），就是死亡行军的项目。“这是一个死亡行军项目，我希望自己不要参与进去。”软件专家可能在办公室中这样给家人打电话。家人以前极有可能听说过这种情况，遗憾的是，现在有太多的项目成为死亡行军，而家人可能知道——甚至比软件专家本人更为了解，避免参与是不可能的。术语“死亡行军”不是扬顿发明的，而是由那些发现自己涉及到只有超常地努力、