案例 1：美国 IBM 公司在 1963 年至 1966 年开发的 IBM360 机的操作系统。这一项目花了 5000 人-年的工作量，最多时有 1000 人投入开发工作，写出了近 100 万行源程序。据统计，这个操作系统每次发行的新版本都是从前一版本中找出 1000 个程序错误而修正的结果。

这个项目的负责人 F. P. Brooks 事后总结了他在组织开发过程中的沉痛教训时说：“…正像一只逃亡的野兽落到泥沼中做垂死的挣扎，越是挣扎，陷得越深，最后无法逃脱灭顶的灾难。…程序设计工作正像这样一个泥潭，…一批批程序员被迫在泥沼中拼命挣扎，…谁也没有料到问题竟会陷入这样的困境…”。

IBM360 操作系统的历史教训成为软件开发项目的典型事例为人们所记取。而 Brooks 博士随后写出了软件工程领域的经典著作《人月神话》(The Mythical Man-Month)，至今畅销不衰。[1]

案例讲解重点：软件工程中不是人多力量大!

案例 2：[摘自教材] In the late 1960s, a bright-eyed young engineer was chosen to “write” a computer program for an automated manufacturing application. The reason for his selection was simple. He was the only person in his technical group who had attended a computer programming seminar. He knew the in’s and out’s of assembler language and Fortran, but nothing about software engineering and even less about project scheduling and tracking.

His boss gave him the appropriate manuals and a verbal description of what had to be done. He was informed that the project must be completed in two months.

He read the manuals, considered his approach, and began writing code. After two weeks, the boss called him into his office and asked how things were going.

“Really great,” said the young engineer with youthful enthusiasm, “This was much simpler than I thought. I’m probably close to 75 percent finished.”

The boss smiled. “That’s really terrific,” he said. He then told the young engineer to keep up the good work and plan to meet again in a week’s time.

A week later the boss called the engineer into his office and asked, “Where are we?”

“Everything’s going well,” said the youngster, “but I’ve run into a few small snags. I’ll get them ironed out and be back on track soon.”

“How does the deadline look?” the boss asked.

“No problem,” said the engineer. “I’m close to 90 percent complete.”

If you’ve been working in the software world for more than a few years, you can finish the story. It’ll come as no surprise that the young engineer stayed 90 percent complete for the entire project duration and only finished (with the help of others) one month late.

案例讲解重点：没有专业的设计规划，匆忙开始写代码，注定悲剧。

案例 3：In the early 1980s, the United States’ Internal Revenue Service (IRS) hired Sperry Corporation to build an automated federal income tax form processing system. According to the *Washington Post*, the “system has proved **inadequate to the workload, cost nearly twice what was expected and must be replaced soon**” (Sawyer 1985). In 1985, an extra **$90 million** was needed to enhance the original **$103 million** worth of Sperry equipment. In addition, because the problem prevented the IRS from returning refunds to taxpayers by the deadline, the IRS was forced to pay **$40.2 million** in interest and **$22.3 million** in overtime wages for its employees who were trying to catch up.

In 1996, the situation had not improved. The *Los Angeles Times* reported on March 29 that

there was still no master plan for the modernization of IRS computers, only a six-thousand-page technical document. Congressman Jim Lightfoot called the project "a **$4-billion** fiasco that is floundering because of **inadequate planning**" (Vartabedian 1996). [2]
案例讲解重点：组织计划不周带来的后果是灾难性的，软件开发远远不是只有编程。

案例 4：某公园有一游船码头，负责人希望开发一游船管理系统，要求如下：当游客租船时，管理员输入 S 表示租船周期开始；当游客还船时，管理员输入 E 表示租船周期结束。一天结束时，要求系统打印出租船次数和平均租船时间。[1]

开发人员给出的算法：

```
Number = Total_time = 0;
Get Message;
While ( !End_of_stream ) {
  if (Code == S) {
    Number ++;
    Total_time  –= Start_time; }
  else  Total_time  +=  End_time;
  Get Message; }
Print Number;
If (Number)  Print (Total_time / Number);
```

新需求：

1. 输出一天中的最长租用时间。
2. 将报告分上午和下午输出。
3. 当通信线路出问题时，能从计算中删除一切不完整的租船信息。

案例讲解重点：可以发现任何一个新需求都将导致程序员完全重写代码！所以不要以为软件的第 1 个版本推出就没事了，那才是维护工作的开始。软件设计必须有充分的远见，预计到各种可能的需求扩展。