现代软件工程技术分析 Therac－25 灾难事故

The analysis of therac－25 disaster event using modern software engineering

李三波 （丽水职业技术学院机电信息分院 浙江丽水 323000）
项 赟 （浙江省丽水市地税局）

摘要：Therac－25 事件到今已有 20 年的历史，往事不堪回首，针对事故的复杂性，本文利用系统工程的观点和现代软件工程的基本原理，对事故的复杂性未重视、系统工程被漠视、软件工程被忽视、用户界面被藐视和软件测试被轻视等五个方面进行了分析，并对系统总体安全设计在软件开发过程中的重要性作了论证。

关键词：Therac－25 系统工程 软件工程

1 问题的提出

Therac－25 是由加拿大原子能有限公司制造的放射治疗仪，设备在 1985 年 6 月至 1987 年 1 月使用期间共发生了 6 个剂量辐射事件，结果造成 4 位病人死亡、2 人重伤的特大医疗事故[1]。Therac－25 事件至今已有 20 年的历史，事故表面现象是由超剂量辐射造成，但实际上，深层的原因是系统和软件的安全性设计方面存在严重问题。

2 Therac－25 事故重现

放射线治疗肿瘤技术起源于 20 世纪 60 年代，Therac－6 和 Therac－20 是 Therac－25 的前身，Therac－25 属于第三代医用高能电子线性加速器，采用双通概念，使仪器的空间更加紧凑和易于使用。Therac－25 比早期的设备具有更高的能量，能够对深部的病变进行治疗，同时降低了治疗费用，缩短了治疗时间。但 Therac－25 在使用不到二年的时间里，却出现六次灾难事件，回放如下[3]：

（1）1985 年 6 月，一名 61 岁的妇女，到 Marietta 的 Kennestone 肿瘤中心接受锁骨部位的 10 兆电子伏特电子射线照射，治疗中病人感到炙热和疼痛，大声喊叫，医生 Tim Still 无法解释，怀疑与过量辐射有关，并与 AECL 电话联系。AECL 工程师答复称设备没有发生超剂量的可能。

（2）1985 年 7 月，一个 40 岁女性子宫颈癌患者，在加拿大安大略 Hamilton 接受 Therac－25 治疗，治疗剂量为 200 拉德，治疗过程中机器停机，显示出现出现"HTILT"错误，同时控制台显示"No dose"和治疗暂停，操作人员按键盘恢复继续运行，同样错误再次发生，在发生 5 次之后，机器进入悬挂状态，进行了重启动的操作。病人当时反应有强烈烧灼感和电击麻刺感，该病人在 5 个月后死亡。据以后的分析，该病人在治疗过程中实际受到 15000 拉德的辐射，对人体而言，辐射剂量达到 1000 拉德就已经是致命的了。

（3）1985 年 12 月，一名妇女经过 X 射线治疗后，对准 Therac－25 光束发射槽的肤色褪色。幸运的是她虽然受伤但幸免于难。

（4）1986 年 3 月，一个男性背部肿瘤患者，在美国东得克萨斯 ETCC 泰勒医院接受 Therac－25 治疗，治疗模式为电子射线，剂量为 180 拉德，面积为 10 厘米 × 17 厘米，操作人员对设备操作十分熟悉，迅速敲击键盘，输入相关数据，发现模式显示为 X（X 射线），于是更改为 E（电子射线），启动机器，机器很快停机，显示"Malfunction 54"，这个信息的含义在说明书中没有明确定义，其解释是能量已发射，可能过低或过高。控制台显示为剂量过低，操作人员于是进行了恢复和重启动的操作，这时治疗舱内的病人已无法忍受，跳下床敲门，治疗被迫终止。5 个月后该病人死亡，据分析该病人接受了 16000 至 25000 拉德的辐射。

（5）1986 年 4 月在上述事件发生三周之后，美国

东得克萨斯 ETCC 为一个男性面部皮肤癌患者作 Therac－25 电子射线治疗，剂量为 180 拉德，仍然由相同的操作人员操作，事件发生过程几乎与上例完全相同，病人剧痛大声叫喊。由于脑部受损，20 天后死亡，据分析该病人接受了 25000 拉德的辐射。

(6) Therac－25 废除前的最后一次过剂量辐射发生在 1987 年 1 月。由于另外一种软件错误，导致病人遭到强电子束照射，结果病人于 4 月份死亡。

3 软件工程基本原理

软件工程主要是针对 20 世纪 60 年代"软件危机"而提出的，它是一门研究用工程化方法构建和维护有效的、实用的和高质量的软件的学科。软件工程的目标是提高软件的质量与生产率，最终实现软件的工业化生产。软件工程的基本原理有七条：

(1) 用分阶段的生命周期计划严格管理

(2) 坚持进行阶段评审

(3) 实行严格的产品控制

(4) 采用现代程序设计技术

(5) 结果应能清楚地审查

(6) 开发小组的人员应该少而精

(7) 承认不断改进软件工程实践的必要性

4 Therac－25 事故的现代工程技术分析

到了今天事故原因早已查明：Therac－25 故障不仅是由于一般的软件错误造成，故障的根本原因是系统总体安全设计问题。现运用现代软件工程技术，对 Therac－25 医疗事故作如下分析：

4.1 事故的复杂性未重视

任何事故的发生，很少是单纯的，通常包含在诸多相互关联事件构成的一个复杂网络中，涉及诸如技术、人文、组织等因素。这次导致 Therac－25 多次事故发生的重要原因在于没有非常明确的证据的条件下，就确信事故的原因已经查明，例如将 Hamilton 事故中的微型开关作为事故的主要原因，并且忽略了对其他各种可能的相关因素的分析。另外一个错误的假定是认为，改正了一个软件错误，就会预防事故今后发生，实际上软件故障总是一个接一个地不断暴露。如果把事故原因简单地归结为人为错误、计算机错误和软件错误都是没有帮助的和毫无意义的。从现在来看，Therac－25 医疗事故接二连三的发生，其中一个重要的因素就是对事故的复杂性没能引起充分的重视。

4.2 系统工程被漠视

现用系统工程的观点，从复杂系统事故的角度来分析 Therac－25 事故，涉及的因素概括为：

(1) 管理缺位，缺乏确定的程序跟踪所有报告的事故；

(2) 对软件过分信任，以至删除了所有的硬件互锁装置，使软件成为可引发事故的单点失效；

(3) 低水平的软件工程实践；

(4) 不实际的风险评估和过分信赖评估的结果。

在本案例中，一个主要错误在于对软件过分信任。非软件的专业人士似乎认为软件是不会失效的，从而过分依赖计算机控制功能。其实软件虽然不会发生如同硬件一样的损耗失效，但是软件的设计错误更难于发现和消除。硬件的失效模式一般是有限的，所以构建保护机构比较容易，从 Therac－25 得到的教训是在实施计算机控制时不要删除标准的硬件互锁装置。

硬件备份、互锁和其他的安全保护器件，在许多不同的系统中现在已经被软件置换，其中包括商用飞机、核电站和武器系统。在硬件互锁装置存在的地方，他们也常被软件控制。在设计危险系统时，如果认为一个故障就足以导致严重事故，那就违反了系统工程的基本原理。软件需要作为一个单独的元件来处理，软件不应该单独承担安全的职能，在系统设计中，必须避免单个软件错误或软件工程错误就足以造成灾难性后果。

当前工程界的另外一个趋势是忽略软件，Therac－25 的第一次安全分析，没有包括软件，当问题出现后，分析者又将注意力完全集中在硬件上。调查软件可能的影响不应成为事故分析的最后一个环节，事实上软件错误可以导致硬件的瞬时故障，因为软件给被控制的硬件发布指令。

在 Therac－25 中，病人的反应成为衡量辐射程度的唯一实际指示。Therac－25 完全依赖操作人员，没有独立的机构检查操作是否正确，而机器本身也不能检测大剂量辐射是否发生。Therac－25 的离子室不能掌握高密度的电子束，在它们饱和的时候，显示的是低辐射剂量。

任何一家公司在建造安全关键系统的时，应该进

行核查实验，一旦发现问题的任何线索，应该有既定的事故分析程序，医生 Tim Still 的第一个电话就应该促使 Kennestone 和 AECL 进行广泛调查，第一件诉讼就应该立即启动应急反应程序。每一个使用危险设备的企业都应该有危险记录和跟踪，同时使事故的报告和分析成为其质量控制过程的一部分，这不仅有助于事故的预防，而且可以降低保险费率，并在诉讼发生后提供背景资料。

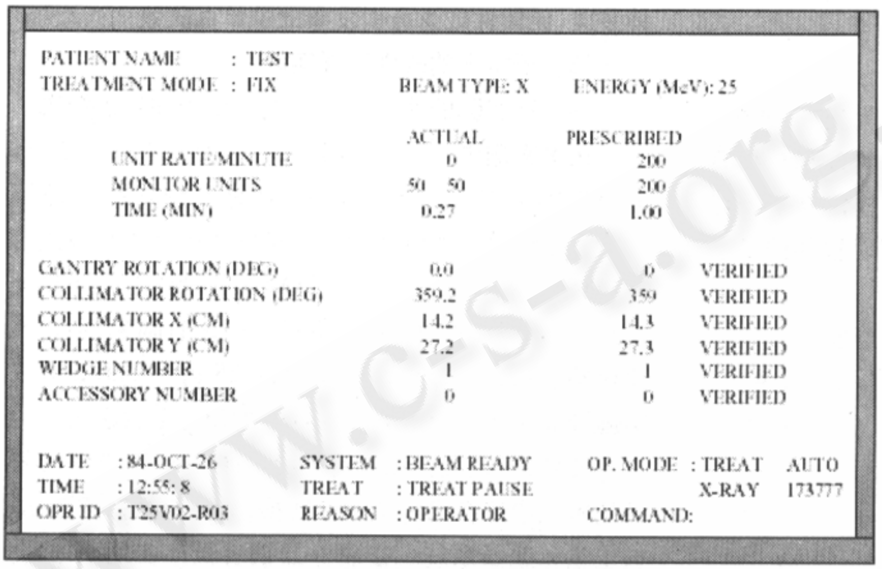

图 1　控制台屏幕显示图

最后，过分依赖安全性分析的数字结果是不明智的，在 Hamilton 事故发生、微开关故障改进后宣称安全性提高了 5 个数量级是无法验证的。

4.3 软件工程被忽视

Therac－25 中包括了软件编码错误，这个问题在其他与计算机相关的一般事故中是少见的。一般计算机错误主要涉及需求、环境条件和系统状态等。虽然实施软件工程不能完全消除软件中的错误，但是可以极大地减少错误。许多公司在他们的系统中使用软件，但是并不象软件工程师那样严肃对待，下述软件工程的基本原则在 Therac－25 中受到明显的破坏：

（1）编制软件文档不应是一种事后行为；

（2）应该建立软件质量保证体系和标准；

（3）应保持设计简单性；

（4）如何得到错误信息，例如软件核查试验，应该从软件设计开始时就制订出方案；

（5）软件应该在模块级和软件级进行广泛的测试和分析，仅进行系统测试是不正确的；

安全性必须构造入软件系统，任何安全关键软件项目必须包括特殊的安全性分析和设计程序，此外不管软件错误是否存在，系统级安全性必须得到确切的保证。Therac－20 包含有导致泰勒事件的同样软件错误，但是硬件的互锁消除了故障的后果。

4.4 用户界面被藐视

用户界面在 Therac－25 事件中受到了某种程度的关注，实际上它在这次事件中只有部分影响，虽然软件的界面和这个软件的其他部分一样，存在改进的余地。软件工程师需要接受更多的界面设计培训，从人－机工程的角度需要更多的数据输入。不过在当时 Therac－25 是由小型机软件控制，因此该机器曾被吹捧为特别"用户友好的"[3]。但实际上 Therac－25 控制台屏幕显示图 1 所示，从窗口可发现操作介面并不友好。

必须着重指出在用户友好界面和安全性方面存在着潜在冲突。用户界面设计的一个目的是尽可能方便操作者使用，但是在 Therac－25 软件中，操作的简单性是以牺牲系统安全性为代价的。最后不仅在初始设计中必须考虑软件和软件界面的安全性，而且需要记录决策理由，使得以后的变更有依据可查。

4.5 软件测试被轻视

事实上，对于软件来讲，不论采用什么技术和什么方法，软件中仍然会有错。测试的目的是发现程序中的错误，是为了证明程序有错。软件测试在产品开发中占据相当重要的一部分，统计表明，在典型的软件开发项目中，软件测试工作量往往占软件开发总工作量的 40% 以上。而在软件开发的总成本中，用在测试上的开销要占 30% ～50% 。如今微软的软件测试人员是开发人员的 1.5～2.5 倍。正是由于清晰地认识到了软件测试的重要性，微软的产品质量才有了明显的提高。

ETCC 泰勒医院 1986 年 4 月 Therac－25 事故发生后，ETCC 立即停止 Therac－25 工作，并通知 AECL。

ETCC 的医生立即对事故进行了仔细的调查。由于机器的女操作员能够确切记忆事故发生时设备实际操作过程，经过一段时间的努力，终于使错误信息"Malfunction 54 "信息得以重现。他们发现，在机器的编辑阶段，数据的输入速度是该错误出现的关键因素，也就是说，对于一个熟练的操作人员，在重复同样的操作千百次之后，编辑速度越来越快，最终将使"Malfunction 54 "信息出现，即超剂量辐射事故发生。参加实验的医生是在经过了相当长的实践后达到了临界的编辑速度。次日对结果感到迷惑的 AECL 工程师来到现场，直到他在医生和操作人员的训练下使"Malfunction 54 "信息再次出现时，才接受了医院的看法，并测量这时的辐射剂量已经达到饱和的 25000 拉德。

得到这个消息的美国芝加哥大学联合辐射中心的医生在他们的教学设备 Therac –20 上进行了实验，两个月后医生们观察到在学生实验中经常出现保险丝烧毁和继电器断路的事件[3]，经分析其实质与 Therac –25 故障完全相同，但是由于有硬件电路的保护，没有造成任何严重的影响。

现在我们可以设想一下，假如当时测试能更加全面一点、科学一些，有步骤地通过单元测试、集成测试和规范的系统测试。Therac –25 灾难事故也许可以避免。

5 结论

随着社会对软件的日益依赖，软件产品的功能复杂性和结构复杂性越来越高，因软件系统失效造成事故将不可避免，如何利用现代软件工程技术，减少或者避免 Therac –25 类似事故的发生，是软件开发部门必须要面对课题。

参考文献

1 Evelyn stiller Cathie LeBlanc 著，基于项目的软件工程面向对象研究方法[M]，贲可荣等译，北京：机械工业出版社，2002. 6.

2 Evelyn Stiller, Cathie LeBlanc. Project – Based Software Engineering: An Object – Oriented Approach. Pearson Addison Wesley. 2004.

3 Peterson, I. Fatal Defect: Chasing Killer Computer Bugs. New York: Random House, 1995.

4 Grady Booch. Object Solutions: Managing the Object – Oriented Project. Addison – Wesley Pub Co, 2002.

5 Scott W Ambler，面向对象软件开发教程，车皓阳，刘悦译，第 2 版，北京：机械工业出版社，2003.

6 Howard, Michael and David LeBlanc, Writing Secure Code, Second Edition, Microsoft Press, Redmond, Washington, 2003.

7 Swiderski, Frank and Window Snyder, Threat Modeling, Second Edition, Microsoft Press, Redmond, Washington, 2004.