这本书的定位是非常明确而又清晰的。这本书包括了微软普遍的测试实践、工具和技术，在讲解这些专业性较强的内容的同时，也穿插了一些有意思的小故事。所以，无论是对于一个测试人员还是对于一个对微软测试感兴趣的非测试人员，这本书都是具有很强的可读性的。虽然是讲述了微软内部测试工作相关内容，但是穿插的小故事，浅显易懂的类比都让人觉得，这不仅是一部介绍微软的书籍，同时也具有极强的可读性。

这本书大致分为了四个部分，介绍的内容循序渐进，使得读者更加容易理解。首先介绍了微软内部做测试的大致环境，如：产品、工程师、测试人员、测试的作用等等；紧接着，介绍了许多在微软中较常使用的测试工具和实践案例；第三部分介绍了“测试工具和系统”，较为细致专业；最后一部分探讨了未来微软在软件测试方面的展望及发展的方向。简单来说，这本书描述了微软怎么用很多已知的软件测试技术和工具来改进软件质量。

我们的读书笔记也将按照书中的思路来进行。

首先，讲一讲第一部分“关于微软”。

第一部分中，主要讲述了“微软的软件工程”，“微软的软件测试工程师”以及“工程生命周期”。

首先，本书在开篇首先介绍了微软年会的盛况，在其笔下的微软是一个充满着活力与创造力的地方，书中介绍了微软的各个主要部门以及微软的组织结构。其中，有三个软件开发本部，分别是：平台产品及服务本部（PSD），企业本部（MBD），娱乐及设备事业本部（E&D）。其基本划分原则是基于不同的市场划分和客户群体。比如：E&G专供娱乐，MBD做企业软件，而PSD致力于机那里能够帮助生意伙伴和微软其他部门为其他公司和顾客提供解决方案的平台。

而在于微软这样大型的软件工程公司，有条不紊的完成各项工作无疑是个技术活，截止2008年，微软在总员工数达到了10万，这是一个庞大的数字。因此，微软主要有两个模式来组织工程师队伍，分别是：产品部门经理模式和共享团队模式，这两个模式在一个产品的开发过程中是可以相互转换的，目的是统一的，做出更好的产品，书中对于微软工程队伍和产品组织给了一个很好的比喻：像创作戏剧一样研制和发布软件产品，还累毙了爱丽丝梦游仙境这个奇妙的童话故事。

接下来介绍一个令人印象深刻的故事：比尔盖茨的思考周，每两年一次公司大大小小工程师提交自己的思考白皮书，刚开始比尔盖茨甚至会钦此审阅并亲手批注，好的idea会被全公司共享甚至成为现实。这个故事对于任何人都是触动极大的，这应该也是微软能够保持创造力和活力的原因之一。而在这个部分，这本书还介绍了微软的只能种类，以及微软的全球化运营，其中在中国和印度的从业人员比例是不断上涨的，作为一个中国人，看到还是觉得非常有意思的。这体现出微软发展的全球化趋势。

介绍完微软的软件工程，软件测试工程师的部分介绍了软件测试工程师在微软中的工作职能、职业发展道路，并且尤为强调了软件测试工程师的重要性。对于微软来说，软件测试工程师与软件开发工程师的人员比例甚至可以达到1：1之多，而软件测试工程师也不是微软一开始创办就有的。微软初建立时还无法达到软件工程化开发这个水平，微软最早的测试工程师是一位在1979年加入微软的高中实习生，这也是个有趣的小故事，而软件测试工程师这个职位更加是在1985才被搬上了台面，而在微软内部对于这个职位的名称也经历了一番讨论，才确定了SDET“软件开发测试工程师”这个名称。在这一章节中，介绍了一些对于测试微软不同职位的品质需求以及如何成为微软的一名SDET，这里就不再赘述。而对于微软SDET的职业发展方向，微软实行管理和技术轨道交叉发展的模式。对于一名测试工程师来说，你可以选择成为一名测试架构师，测试独立贡献者或者测试管理人员。尽管成为一名管理人员并不意味着升值。而对于测试工程师的招聘，包含了校园招聘和业界招聘两种途径，对于加入微软测试行列的人们来说，他们共同保证了微软产品的质量与工程的水平。

软件测试工程师被我们了解之后，书中提及了工程生命周期，与平时课本上了解到的最最基本的瀑布模型、螺旋模型、敏捷开发等不同的是，作为一个商业化的企业，微软的开发模型要更加的高效且便捷，但绝大部分是基本软件工程模型的改进或者变体，这里面体现出微软对于优点的继承以及不断的革新。而微软也有属于自己的一套软件生命周期工作流程，主要由：ISO9000，六西格玛，CMMI，精益六西格玛。可靠的流程保证了软件的质量与品质。而在软件发布之前会从“作战室”发布，这个过程包含了明确的讨论以及问题的处理，可高效、优质的解决问题。这一章节中，作者将做软件类比做饭，这个有着固定模式但又需要不断调整容许不断尝试的过程，私以为这是十分恰当的，也许做软件的过程并不能说是最好的，但显然，微软是不断追求做好的软件的。

第二部分，是“关于测试”。

这部分首先讲的便是软件测试用例设计的实用方法，这部分介绍了一些我知道的方法：比如黑盒白盒测试、用好坏数据进行测试等等，但是更多的像是软件测试模式、测试时间估计、如何问问题、测试可行性，甚至还有配对测试，这些对于一个刚刚接触软件测试的人来说都是很新奇而且有趣的，在阅读这部分的时候也感觉在专业知识方面大有收获。更加重要的是，这里所讲述的一切都告诉我们，没有完美的技术可以保证完全优秀的测试，一个优秀的软件测试工程师必须要有一双善于发现的眼睛和一颗勤于思考的心。

“功能性测试”这部分，讲了几种功能性测试相关的技术：边界值分析、等价类划分、组合分析以及状态转换测试等等。这一段最令我印象深刻的是讲述的保护菜园里的蔬菜不被害虫破坏的小故事，通过这个故事我们了解到，没有完美的杀虫方案。而对于功能新测试，等价类划分、边界值分析、组合测试都进行了较为详尽的解释，给出了一些尤为典型的例子，如windows XP的文件名问题（等价类划分），千年虫问题（边界值分析）等。这些技术系统的分析了一个程序的功能性组件，增加了测试的覆盖（相比于单纯的黑盒测试），不同技术的使用有助于暴露不同的、特定类型的缺陷，这些技术有效的增加了测试的有效性，减少了测试的冗余性，也更加的科学可靠。

“结构测试技术”，开篇伊始，作者讲了一个关于苹果颜色的小故事，来阐明软件隐藏在用户界面之下的计算特征。而结构测试技术通过对函数控制流程的详细分析帮助降低风险，这样的测试对于研究分析程序的部分代码很有效。这属于一种白盒测试设计方法。不同于行为、探索性测试，这种方法不是评估项目外在体验而是真正对于内部问题进行剖析。这里面提到了老师上课所讲的一些方法：比如块测试、决策测试、条件测试、基础路经测试等等，有其思路以及一些具体的代码实现、分析。并且就像老师上课时所讲的，也对这些不同的测试方法进行了一番优劣、包含比较，令人获益匪浅。在这里，作者也指出，行为测试方法、系统化的功能测试方法、结构测试方法是从不同方面根据不同的信息来评测软件，并不能够说某一个是不必要的。

“用代码复杂度分析风险”这一篇章，告诉我们：对于软件的测试，要有的放矢，学会筛选，有针对性的投入测试力量，更加高效、合理的进行软件测试。书中指出测试常常是风险管理的过程。将可用资源集中于最需要的区域是十分有必要的。那么，怎么才能做好这样的风险管理呢？这与代码复杂度息息相关，而代码复杂度的测定可以根据：代码行数、贿赂复杂度、Halstead度量、面向对象的度量等，对于测定了复杂度的代码，也会产生误诊以及偏颇的认知，对于复杂但清晰已维护少缺陷的代码，易出现误诊，而对于真心复杂的新代码而言，不妨尝试对代码重新分解。而代码复杂度是识别应用程序中可能存在缺陷的一种基本度量，对于识别代码为户型也有同等价值。但是要注意，使用的时候一定要足够“聪明”，否则也会带来不必要的麻烦。

“基于模型的测试”这部分强调了模型对于复杂问题简单化、形象化的重要作用，进而讲解了模型测试：设计模型、模型化软件、建立有限状态模型、模型自动化、图论和MBT、随机模型、语法模型等，这些介绍过程中用到的大多是数据结构和编译原理的知识，强调了运用。接下来文章介绍了微软基于模型的几种测试工具：Spec Explorer、MBT及其使用实例，并且声明建模的意义在于使用，而任何类型的建模都是有意义的，这样的测试是很强大的，有助于理解复杂的系统、帮助管理风险以及寻找漏洞。具有很强的实际操作意义。