软件质量保证课程结课论文

题目： 浅谈如何保证软件设计的质量

学院： 软件学院

专业： 软件工程

姓名： 韦 薇

学号： 14126151

北京交通大学

2015 年 5 月 23 号

目录

一、软件设计的定义

软件设计是从软件需求规格说明书出发，根据需求分析阶段确定的功能设计软件系统的整体结构、划分功能模块、确定每个模块的实现算法以及编写具体的代码，形成软件的具体设计方案[1]。

软件设计把许多事物和问题抽象起来，并且抽象它们不同的层次和角度。将问题或事物分解并模块化使得解决问题变得容易，分解的越细模块数量也就越多。它的副作用是使得设计者考虑更多模块之间耦合度的情况。

软件设计一般分为：

1、 体系结构设计

软件体系结构设计是高层次设计，其将软件需求转化为数据结构和软件的系统结构，并定义子系统和它们之间的通信或接口。

2、详细设计

详细设计在过去被称为总体设计或概要设计。详细设计通过对结构进行细化，得到软件详细的数据结构和算法。

二、保证软件设计质量的意义

软件设计在软件的开发过程中处于至关重要的地位，其重要性体现在以下几个方面：

1、软件设计在整个软件项目的建设中起着承上启下的重要作用。

从整个软件项目开发阶段来看，软件项目可以分为需求、设计、编码、验证四个阶段。设计承接需求分析，基于准确的需求分析，对项目目标进行结构化搭建。设计阶段产生的设计说明书以及设计规范是编码阶段的作业指导，也是测试人员开发测试用例的指导书。

2、软件设计是对软件项目质量进行保障的关键步骤。

软件项目的质量与需求分析、设计、编码、验证段这四个阶段的质量之间的关系，可以用 C 语言表达为：最终的软件质量 = 需求分析质量 && 设计质量 && 编码质

量 && 验证质量，这种“与”的关系表明任何一个阶段出现质量纰漏，软件项目的最终质量都无法保障[2]。

3、设计阶段提供的软件表示，使软件项目质量的评价成为可能。

反映软件设计质量的要素有：准确性、稳健性、安全性、通信有效性、处理有效性、可操作性、完备性、一致性、可追踪性、可见性、可扩充性、复用性、模块性、清晰性、自描述性、简单性、结构性、硬件系统无关性、软件系统无关性、文档完备性等。通过这些考核要素对设计阶段质量进行控制，从而达到从项目前端控制软件质量的效果。同时该阶段的设计规范也是进行软件质量评价的参照标准与基本要求。

因此，想做好整个软件项目的质量保障，必须充分重视设计阶段的质量保障工作。

另外，在《中华人民共和国国家标准——计算机软件质量保证计划规范》中也有对设计阶段的规定：在软件概要设计结束后必须进行概要设计评审，以评价软件设计说明书中所描述的软件概要设计在总体结构、外部接口、主要部件功能分配、全局数据结构以及各主要部件之间的接口等方面的合适性。在软件详细设计阶段结束后必须进行详细设计评审，以确定软件设计说明书中所描述的详细设计在功能、算法和过程描述等方面的合适性[3]。

可见软件设计阶段在整个软件开发过程中的重要性，因此，软件设计的质量就显得尤为重要。软件设计的质量决定着整个软件项目的质量。

三、出现质量问题的原因

尽管软件设计在软件开发过程中处于十分重要的地位，但是由于种种原因，软件设计的质量仍然不尽人意。软件设计阶段经常出现的质量问题从大的方面看有以下几种原因：

1、需求分析阶段工作不充分

好的软件设计必然基于准确的需求分析，离开正确的需求分析，软件设计从源头开始便是错误的，再好的设计也没有任何意义。软件项目可能因为工期紧张、乙方软件企业管理不规范、甲方用户技术受限或配合不到位、承建方需求分析人员业务或者技术经验不足等原因，需求调研不够完善深入，甚至基本的业务逻辑也没有得到确认，就匆忙进行需求分析以及架构设计，其结果必然是灾难性的。

2、设计不充分

设计不充分的原因，首先是许多软件企业对软件设计工作不重视，甚至有的企业省略设计阶段便开始编码；其次，设计人员技术不够过硬或者缺乏经验、对业务理解不够深入、未能充分考虑后期需求变动对设计的影响也是造成设计不充分的重要原因。

设计不充分往往导致频繁变更与诸多性能、安全方面的漏洞。对于成本来说，在软件项目里，越是在项目前期发现问题，解决成本越低。据相关机构统计，在设计阶段发现偏差比在需求分析阶段发现并修正要高出 5 倍，在编码阶段觉察偏差则会提高到 10 倍，而如果延续到单元测试或系统测试阶段发现设计缺陷修正成本则会提高到 20 倍。

3、过度设计

与设计不充分相对应的一种情况是设计过度。过度设计是由于设计人员在做项目分析设计时，过分的考虑潜在的、未来的以及准备扩展的等因素，过度地抽象，过多地思考封装、分离解耦，导致太多颗粒单位，太多插件等，这对设计资源造成了不必要的浪费，并且可能导致原本可以简单实现的逻辑变得复杂，造成系统整体性能的下降与维护成本的上升等，以至于影响到用户体验或者系统无法使用[4]。

四、软件设计原则和衡量质量的标准

软件设计既然容易有设计不充分或者过度设计的问题，那么软件设计就应该有一定的原则让设计者去遵循，从而能够正确、适度地对软件项目进行设计。软件设计的思想原则如下：

1)、用户需求远比技术重要；

2)、需求其实很少改变，改变的是对需求的理解；

3)、接受变化；

4)、不要低估软件规模的需求；

5)、在软件设计中没有捷径可以走；

6)、任何体系结构都有它自身的优点和缺点，设计模式也一样。

根据这些原则得到的软件设计，一般用可以从以下几个方面来衡量其质量[5]：

1）功能性：包括完全性、正确性、安全性、兼容性和互用性。完全性包括功能点覆盖率，重点功能点覆盖率，优先功能覆盖率。正确性包括需求一致度。安全性根据软件需求的不同有不同的安全性要求。

2）效率：从产品运行的时间效率和利用的硬件资源两方面来考虑。

3）维护性：包括架构的可改正性，可扩充性以及可测试性。如果用户的一个很小的需求变更会引起架构设计很大的变化，那么这样的架构设计的可改正性和可扩充性是较差的。

4）可移植性：包括硬件独立性、软件独立性、可安装性和可重用性。其考虑因素应该包括软件设计是否模块化、每个模块的可复用性。

5）可靠性：包括缺陷数量、容错性和可用性。

6）使用性：包括可理解性、易学习性、可操作性和易沟通性。软件项目的最终目的是被用户使用，易用性、可操作性不佳都会影响用户对软件的接纳程度。因此软件的可使用性也是非常重要的。

五、保证软件设计质量的措施

为了达到软件设计的质量标准，针对上述软件设计可能出现的质量问题，根据软件设计的原则，可以总结出以下几点保证软件设计质量的措施：

1、思想上重视

首先是要充分认识设计阶段的重要性，从思想上强调设计阶段质量保障工作的必要性与重要性。关于软件设计的重要性前文已从几个方面作了总结，不再赘述。项目团队成员与甲方都要充分理解并一致认同设计规范与设计评审等质量管理措施对整个项目的意义与重要性。

2、选用合适的设计思想、设计方法

在设计阶段，设计人员应在充分了解需求与项目背景的前提下，根据项目的具体情况与应用场景采用恰当的设计思想和设计方法、合适的建模方法，从指导思想与方法上避免设计阶段的质量瑕疵：

（1）耦合和内聚

耦合和内聚是两个用来评估软件设计质量的方法。因此要降低耦合度，提高系统模块的内聚性。

（2）采用合适的软件体系结构。

软件体系结构为软件系统提供了一个结构、行为和属性的高级抽象，由构成系统的元素的描述、这些元素的相互作用、指导元素集成的模式以及这这些模式的约束组成。设计人员应该根据实际情况采用不同的体系结构，常见的体系结构如下：

1）C/S 软件体系结构

传统的二层 C/S 结构存在局限性。

三层 C/S 结构将应用功能分为表示层、功能层和数据层，可以用较少的资源建立起具有很强伸缩性的系统。

2）B/S 软件体系结构

B/S 结构是对 C/S 结构的一种改进。

B/S 结构和三层 C/S 结构比较接近，但也具有自己的特点。

3）中间件的多层分布式的体系结构

具有客户端的表示层、中间的业务逻辑层和数据库服务器的三层或多层体系结构。多层体系结构将客户和资源分开，降低了服务器的负载。多层分布式系统中，不同的组件可以用不同的语言来实现。

4）此外，可以适当考虑异步体系结构，其优缺点如下：

A、异步体系结构优点：

更快的响应时间

负载平衡

具有更好的容错能力

支持断续连接的系统

B、异步体系结构缺点：

利用通知或轮询进行状态跟踪

处理超时

创建和执行补偿逻辑

（3）采用设计模式

设计模式使得人们可以更加简单和方便地去复用成功的软件设计和体系结构，从而能够帮助设计者更快更好地完成系统设计[6]。

模式是一条由三部分组成的规则，它表示了一个特定环境、一个问题和一个解决方案之间的关系。每一个模式描述了一个在我们周围不断重复发生的问题，以及该问题的解决方案的核心。这样，你就能一次又一次地使用该方案而不必做重复劳动[7]。

设计模式在工程小组成员之间提供了通用的语义。设计模式可以更加简单方便的复用成功的设计和体系结构。设计模式有助于作出有利于系统复用的选择，避免设计损害系统复用性。设计模式可以帮助设计者更快更好的完成系统设计。

根据 Foutse Khomh 和 Yann-Gaël Guéhéneuc 关于模式对质量属性影响的实验，得到结论：设计模式对软件质量属性的正影响大于负影响，因此，恰当使用设计模式可以在一定程度上提高软件质量。

典型的设计模式有 MVC 模式、Bridge 模式、工厂模式、单实例模式、享元模式、门面模式、适配器模式、观察者模式等。由于篇幅原因，此处只简单介绍 MVC 模式和 Bridge 模式：

1）MVC 模式

即 Model-View-Controller，把一个应用的输入、处理、输出流程按照 Model、View、Controller 的方式进行分离，这样一个应用被分成三个层，即模型层、视图层、控制层。在设计模式中，MVC 实际上是一个比较高层的模式，它由多个更基本的设计模式组合而成，Model-View 的关系实际上是 Observer 模式，模型的状态和视图的显示相互响应，而 View-Controller 则是由 Strategy 模式所描述的，View 用一个特定的 Controller 的实例来实现一个特定的响应策略，更换不同的 Controller，可以改变 View 对用户输入的响应。而其它的一些设计模式也很容易组合到这个体系中。比如，通过 Composite 模式，可以将多个 View 嵌套组合起来；通过 FactoryMethod 模式来指定 View 的 Controller，等等。

使用 MVC 的好处，一方面，分离数据和其表示，使得添加或者删除一个用户视图变得很容易，甚至可以在程序执行时动态的进行。Model 和 View 能够单独的开发，增加了程序了可维护性，可扩展性，并使测试变得更为容易。另一方面，将控制逻辑和表现界面分离，允许程序能够在运行时根据工作流、用户习惯或者模型状态来动态选择不同的用户界面。

2）Bridge 模式

《Design Patterns: Elements of Reusable Object-Oriented Software》在提出桥梁模式的时候指出，桥梁模式的用意是"将抽象化(Abstraction)与实现化(Implementation)脱耦，使得二者可以独立地变化"。这句话有三个关键词：抽象化、实现化和脱耦。

Bridge 模式的优点为：

A、分离接口及其实现部分；

B、提高可扩充性；

C、实现细节对客户透明。

Bridge 模式不足之处：

A、桥接模式的引入会增加系统的理解与设计难度，由于聚合关联关系建立在抽象层，要求开发者针对抽象进行设计与编程。

B、桥接模式要求正确识别出系统中两个独立变化的维度，因此其使用范围具有一定的局限性。

3、开展软件设计质量评审

设计质量评审是运用早期预警、预防风险的原理，把各方面的经验集中运用于设计开发，为设计决策提供咨询，从而达到发现和弥补设计开发缺陷、确保产品的使用性和经济性的目的。

除了本公司内部进行软件设计质量评审，还可引入专业的第三方质量保障服务机构指导。在一般的软件项目建设中，若乙方充当质量保障的角色，往往会由于节约成本、精力有限或者经验缺乏等原因，设计质量无法令人满意；而甲方由于人员、技术或者精力的限制，加上对软件模块间逻辑关系把握不准，也无法保证对项目的质量进行深入关注。第三方质量保障服务机构靠技术与服务来赢得客户信任，因而更加重视项目的质量与最终用户体验。从而会更加专业的对待项目过程中的质量管理。

六、总结

本文通过对软件设计质量的意义、软件质量可能出现问题的原因、软件设计的原则和检验其质量的因素的阐述，总结出了能够帮助保证软件设计质量的几点措施。软件设计在整个软件的开发过程中处于承上启下的地位，有着至关重要的作用，因此其质量也是决定软件开发成败的决定性因素，设计人员必须在思想上重视软件设计并采

用正确的、合适的设计思想和设计方法，根据项目实际情况进行设计工作，才能为接下来的开发和测试提供根据和保障。

参考文献：

[1]钟志水 姚珺．大学计算机应用基础．重庆：重庆大学出版社，2012：245．

[2]http://www.mypm.net/blog/user11/wcabt/archives/2015/1022710.html

[3]SBTS.计算机软件质量保证计划规范[S]1990.

[4]http://www.mypm.net/blog/user11/wcabt/archives/2015/1022710.html

[5]http://www.mypm.net/articles/show_article_content.asp?articleID=15623

[6]刘伟主编.设计模式[M].北京：清华大学出版社.2011.

[7]（美）亚历山大（Alexander，C.）著；赵 冰译.建筑的永恒之道[M].北京：中国建筑工业出版社.1989.