CSAI.cn 希赛®

飞思考试中心 Fecit Examination Center

全国计算机技术与软件专业技术资格(水平)考试用书

软件设计师考试

新大纲

考点分析与真题详解

(软件设计技术篇)(第二版)

王勇 刘智成 主编

希赛IT教育研发中心 组编

飞思教育产品研发中心 监制

電子工業出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

http://www.phei.com.cn

飞思考试中心
Fecit Examination Center

全国计算机技术与软件专业技术资格(水平)考试用书

软件设计师考试考点分析与真题详解

新大纲

(软件设计技术篇)(第二版)

王　勇　刘智成　　主编
希赛IT教育研发中心　　组编
飞思教育产品研发中心　　监制

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内容简介

本书由希赛 IT 教育研发中心编写，在参考和分析计算机技术与软件专业技术资格（水平）考试历年试题的基础上，着重对新版的考试大纲内容有重点地进行了细化和深化，是此考试中的软件设计师级别的考试辅导用书。分为“计算机与软件工程知识篇”和“软件设计技术篇”两册，内容涵盖了最新的软件设计师考试大纲的所有知识点，书中选取了 1991—2006 年的软件设计师试题中的重点和难点部分，并进行了详尽的分析和解答。

准备参加考试的人员可通过阅读本书掌握考试大纲规定的知识，把握考试重点和难点，熟悉考试方法、试题形式、试题的深度和广度，以及解答问题的方法和技巧等。

本书适合于参加软件设计师考试的人员，也可作为程序员、软件设计师、计算机专业教师的教学和工作参考书。

图书在版编目（CIP）数据

软件设计师考试考点分析与真题详解（软件设计技术篇）/ 王勇，刘智成主编. —2 版. —北京：电子工业出版社，2006.9
（飞思考试中心）
ISBN 7-121-02972-3

Ⅰ.软... Ⅱ.①王...②刘... Ⅲ.软件设计－工程技术人员－资格考核－自学参考资料 Ⅳ.TP311.5

中国版本图书馆 CIP 数据核字（2006）第 087683 号

责任编辑：杨 鸽
印 刷：北京东光印刷厂
出版发行：电子工业出版社
北京海淀区万寿路 173 信箱 邮编：100036
经 销：各地新华书店
开 本：787×1092 1/16 印张：27.5 字数：704 千字
印 次：2006 年 9 月第 1 次印刷
印 数：8 000 册 定价：39.80 元

凡购买电子工业出版社的图书，如有缺损问题，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。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

丛书编写委员会

主　　编：张友生

组　　编：希赛IT教育研发中心

编　　委：（排名不分先后）

出版说明

知己知彼　百战百胜

自2000年初至今，飞思教育产品研发中心先后与微软、金山、新动力集团、Adobe、Autodesk、红旗 Linux、拓林思（TurboLinux）、网虎 Linux、北航海尔等知名软件开发商的授权培训管理中心共同携手，成功推出了以标准培训、权威认证为代表的“培训专家”系列教材。除了“培训专家”，认证考试用书和行业培训教材等也是培训教材不可分割的一部分。在认证考试用书方面，“飞思考试中心”系列丛书已经推出了《研究生入学考试要点、真题解析与模拟试卷》和《全国计算机等级考试考试要点、题解与模拟试卷》等考试用书，其中计算机等级考试丛书上市一年就突破了20万册的发行量。

中国计算机技术与软件专业技术资格（水平）考试（通常简称为“软考”）是国家级的IT专业人员从业资格考试。2003年年底，人事部和信息产业部联合发布了国人部发[2003]39号文件，以软考为基础，对IT领域职称评定进行全面改革，使得已有逾十年历史的软考具有了更诱人的内涵：以考代评全面实现，考过即可获得相应职称。通过软考，在校大学生就可成为工程师或者高级工程师。

但是，软考是一个难度很大的考试，十多年来，考生平均通过率极低。主要原因是考试范围十分广泛，牵涉到计算机专业的每门课程，还要加上数学、外语、系统工程、信息化和知识产权等知识，且注重考查新技术和新方法的应用。考试不但注重广度，而且还有一定的深度。为了更好地服务于考生，引导考生在较短时间内掌握解题要领，并顺利通过考试，我们将多年的考试辅导与培训经验进行浓缩，特别编写了这套“全国计算机技术与软件专业技术资格（水平）考试”辅导用书。

✦ 丛书特色

- ✧ 全面反映新大纲：丛书在参考和分析历年考试试题的基础上，着重对新版的考试大纲规定的内容有重点地进行细化和深化。阅读本丛书，就相当于阅读了一本详细的考试大纲的精解。
- ✧ 试题最新最全：丛书详细分析了1991年至2006年上半年的全国计算机技术与软件专业技术资格（水平）考试试题，题量大、内容新，从而便于读者摸清考试新趋向，紧跟考试动态，熟悉考试方法、试题形式，了解试题的深度和广度，以及内容的分布。
- ✧ 名师精心锤炼：丛书由名师主笔，亲授解题技巧。内容全面翔实，文字表达简洁明了，层次清晰，结构严谨，特别突出了解题方法，强调知识的综合与提高，导向准确。
- ✧ 题型分析透彻：丛书重点定位在考试知识点的介绍和解题方法与技巧上，不仅授人以“鱼”，更授人以“渔”，对例题进行了细致深入的分析、完整的解答和点评扩展，能让读者达到触类旁通、举一反三之功效。
- ✧ 全真试题实战：本丛书不但配有例题分析，最后还提供了两套完整的模拟试题，

并给出了详细的试题分析与解答，便于读者实战演练，自测、提高。

✦ 读者对象

丛书作为计算机技术与软件专业技术资格（水平）考试的辅导教程，特别适合于希望在较短时间内通过考试的广大应试考生，也可作为软件设计师、数据库工程师、网络工程师、系统分析师及高等院校师生的工作和教学参考用书。

✦ 关于作者

丛书由飞思教育产品研发中心组织编写，希赛IT教育研发中心负责本书的具体编写工作，作者们不但具有扎实的理论知识，而且具有丰富的实践经验，参与了制定计算机技术与软件专业技术资格（水平）考试大纲的工作，对考试进行了长期的跟踪和研究，其中大多数作者已经参加了多年的软考阅卷工作。

✦ 鸣谢

在此，首先对丛书所选用的参考文献的著作者，以及丛书所引用试题的出题老师表示真诚的感谢，同时也感谢其他朋友对这套书的大力支持。

由于时间仓促，学识有限，书中不妥之处，敬请广大读者指正。

飞思教育产品研发中心

联系方式

咨询电话：（010）68134545　　88254160

电子邮件：support@fecit.com.cn

服务网址：http://www.fecit.com.cn　　http://www.fecit.net

通用网址：计算机图书、飞思、飞思教育、飞思科技、FECIT

前 言

计算机技术与软件专业技术资格（水平）考试已走过了十几年历程，我们深感该考试对于推进国家信息化建设和软件产业化发展起着重要的作用。

计算机技术与软件专业技术资格（水平）考试广泛调动了专业技术人员工作和学习的积极性，为选拔高素质的专业技术人员起到了积极的促进和推动作用，并且为广大的专业技术人员的专业技术水平和职称的评定提供了一个客观、公正的机会，使得优秀、年轻的专业技术人才能够脱颖而出。

然而，计算机技术与软件专业技术资格（水平）考试是一个难度很大的考试，考试通过率极低。主要原因是考试范围十分广泛，牵涉到计算机专业的每门课程，还要加上数学、外语、系统工程、信息化和知识产权等知识，且注重考查新技术和新方法的应用。考试不但注重广度，而且还有一定的深度。特别是高级资格考试（系统分析师考试），不但要求考生具有扎实的理论知识，还要具有丰富的实践经验。

“软件设计师考试考点分析与真题详解（第二版）”是为全国计算机技术与软件专业技术资格（水平）中的软件设计师级别考试编写的辅导用书，分为“计算机与软件工程知识篇”、“软件设计技术篇”两册。其中“计算机与软件工程知识篇”介绍了计算机系统综合知识，内容涵盖了最新的软件设计师考试大纲的所有知识点。“软件设计技术篇”介绍了软件设计知识，内容包括软件设计的基本方法、数据库设计、常用算法设计等，书中选取了1991—2006年软件设计师（高级程序员）试题的重点和难点部分，并进行了详细的分析和解答。

“软件设计师考试考点分析与真题详解（第二版）”在参考和分析历年考试试题的基础上，着重对新版的考试大纲的内容有重点地进行了细化和深化。准备考试的人员可通过阅读本书掌握考试大纲规定的知识，熟悉考试方法、试题形式、试题的深度和广度，以及内容的分布、解答问题的方法和技巧等。

本书不仅对准备参加计算机技术与软件专业技术资格（水平）考试的读者有很大的作用，而且对从事软件设计工作的IT从业人员、计算机教学工作的老师，以及参加其他类似考试的读者也是有帮助的。

本书由希赛IT教育研发中心组编，由王勇和刘智成主编，张友生审阅了所有稿件。

“软件设计技术篇”第1、4章由王勇编写，第2章由徐鹏飞编写，第3章由施游编写，第5章由陈建忠编写，第6章由相红利编写，第7章由周峻松编写，第8章由聂作明编写，第9章由施游和王勇编写。

在本书出版之际，要特别感谢全国计算机技术与软件专业技术资格（水平）考试办公室的命题专家们。编者在本书中引用了历年考试的部分真题。同时，本书在编写的过程中参考了许多相关的资料和书籍，在此恕不一一列举（详见书后的参考文献），编者在此对这些参考文献的

作者表示真诚的感谢。

读者的进步是我们的心愿。您如果发现书中有任何疑惑之处，可在希赛网（http://www.csai.cn）“技术社区”中的“CSAI辅导教程”版块上与作者们进行交流。

由于编者水平有限，且本书涉及的知识点较多，书中难免有不妥和错误之处，编者诚恳地期望各位专家和读者不吝赐教，对此，我们将深为感激。

丛书编委会

目 录

第 1 章　软件设计概述

从功能上的划分来看，软件设计应该是软件设计师的工作。作为一名软件设计师，必须懂得软件设计的基本原则和理论，掌握基本的软件设计方法，具有丰富的软件设计经验。

1.1　软件设计基本原则

在软件设计过程中，必须遵循一些原则，例如信息隐蔽和模块独立性是两个最基本的原则。

1.1.1　信息隐蔽

在一节不和谐的课堂里，老师叹气道："要是坐在后排聊天的同学能像中间打牌的同学那么安静，就不会影响到前排睡觉的同学了。"

这个故事告诉我们，如果不想让坏事传播开来，就应该把坏事隐藏起来，"家丑不可外扬"就是这个道理。为了尽量避免某个模块的行为去干扰同一系统中的其他模块，在设计模块时就要注意信息隐藏。应该让模块仅仅公开必须让外界知道的内容，而隐藏其他一切内容。

在软件设计中同样有信息隐蔽原则。Parnas 提出：在概要设计时列出将来可能发生变化的因素，并在模块划分时将这些因素放到个别模块的内部。也就是说，每个模块的实现细节对于其他模块来说是隐蔽的，模块中所包含的信息（包括数据和过程）不允许其他不需要这些信息的模块使用。这样，在将来由于这些因素变化而需修改软件时，只需修改这些个别的模块，其他模块不受影响。信息隐蔽技术不仅提高了软件的可维护性，而且也避免了错误的蔓延，改善了软件的可靠性。现在信息隐蔽原则已成为软件工程学中的一条重要原则。

1.1.2　模块独立性

软件设计中的模块独立性是指软件系统中每个模块只涉及软件要求的具体子功能，而和软件系统中其他的模块接口是简单的。模块独立的概念是模块化、抽象、信息隐蔽和局部化概念的直接结果。

如何定义模块大小，Meyer 定义了以下 5 条标准。

- 模块的可分解性：如果一种设计方法提供了将问题分解成子问题的系统化机制，它就能降低整个系统的复杂性，从而实现一种有效的模块化解决方案。
- 模块的可组装性：如果一种设计方法使现存的（可复用的）设计构件能被组装成新系统，它就能提供一种不需要一切从头开始的模块化解决方案。
- 模块的可理解性：如果一个模块可以作为一个独立的单位（不用参考其他模块）

被理解，那么它就易于构造和修改。

- 模块的连续性：如果对系统需求的微小修改只导致对单个模块，而不是整个系统的修改，则修改引起的副作用就会被最小化。
- 模块的保护性：如果模块内部出现异常情况，并且它的影响限制在模块内部，则错误引起的副作用就会被最小化。

一般采用两个准则度量模块的独立性，即模块间耦合和模块内聚。

耦合是模块之间的相对独立性（互相联系的紧密程度）的度量。模块之间的联系越紧密，联系越多，耦合性就越高，而其模块独立性就越弱。

内聚是模块功能强度（一个模块内部各个元素彼此结合的紧密程度）的度量。一个模块内部各个元素之间的联系越紧密，则它的内聚性就越高；相对地，它与其他模块之间的耦合性就会减低，而模块独立性就越强。因此，模块独立性比较强的模块应是高内聚、低耦合的模块。

1.1.2.1 内聚

内聚是信息隐蔽功能的自然扩展。内聚的模块在软件过程中完成单一的任务，同程序其他部分执行的过程交互很少，简而言之，内聚模块（理想情况下）应该只完成一件事。在设计模块时应尽量争取高内聚。

一般模块的内聚性分为 7 种，如图 1-1 所示。

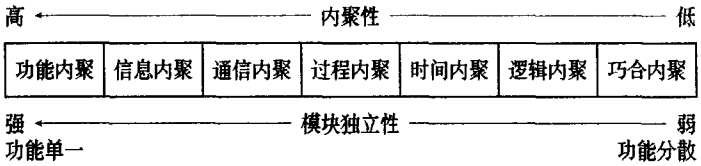

图 1-1 模块的内聚性

一般认为，巧合（偶然）、逻辑和时间上的聚合是低聚合性的表现；信息的聚合则属于中等聚合性；顺序的和功能的聚合是高聚合性的表现。表 1-1 列出了各类聚合性与模块各种属性的关系。

表 1-1 各类聚合性与模块各种属性的关系

	内部联系	清晰性	可重用性	可修改性	可理解性
巧合内聚	很差	差	很差	很差	很差
逻辑内聚	很差	很差	很差	很差	差
时间内聚	差	中	很差	中	中
过程内聚	中	好	差	中	中
通信内聚	好	中	中	中	中
信息内聚	好	好	中	好	好
功能内聚	好	好	好	好	好

1. 功能内聚（Functional Cohesion）

一个模块中各个部分都是完成某一具体功能必不可少的组成部分，或者说该模块中所

有部分都是为了完成一项具体功能而协同工作、紧密联系、不可分割的，则称该模块为功能内聚模块。它是内聚程度最高的，也是模块独立性最强的模块。

2. 信息内聚（Informational Cohesion）

这种模块完成多个功能，各个功能都在同一数据结构上操作，每一项功能有一个唯一的入口点。这个模块将根据不同的要求，确定该执行哪一个功能。由于这个模块的所有功能都是基于同一个数据结构（符号表）的，因此，它是一个信息内聚的模块。

信息内聚模块可以看成是多个功能内聚模块的组合，并且达到信息的隐蔽。即把某个数据结构、资源或设备隐蔽在一个模块内，不为别的模块所知晓。

3. 通信内聚（Communication Cohesion）

如果一个模块内各功能部分都使用了相同的输入数据，或产生了相同的输出数据，则称为通信内聚模块。通常，通信内聚模块是通过数据流图来定义的。

4. 过程内聚（Procedural Cohesion）

使用流程图作为工具设计程序时，把流程图中的某一部分划出组成模块，就得到过程内聚模块。例如，把流程图中的循环部分、判定部分、计算部分分成 3 个模块，这 3 个模块都是过程内聚模块。

5. 时间内聚（Classical Cohesion）

时间内聚又称为经典内聚。这种模块大多为多功能模块，但模块的各个功能的执行与时间有关，通常要求所有功能必须在同一时间段内执行，如初始化模块和终止模块。

6. 逻辑内聚（Logical Cohesion）

这种模块把几种相关的功能组合在一起，每次被调用时，由传送给模块的判定参数来确定该模块应执行哪一种功能。

7. 巧合内聚（Coincidental Cohesion）

巧合内聚又称为偶然内聚。模块内各部分之间没有联系，或者即使有联系，这种联系也很松散，则称这种模块为巧合内聚模块，它是内聚程度最低的模块。

1.1.2.2　耦合

耦合是程序结构中模块相互关联的度量。耦合取决于各个模块间接口的复杂程度、调用模块的方式，以及哪些信息通过接口。

耦合的强度依赖于以下几个因素：

- 一个模块对另一个模块的调用。
- 一个模块向另一个模块传递的数据量。
- 一个模块施加到另一个模块的控制的多少。
- 模块之间接口的复杂程度。

一般模块之间可能的连接方式有 7 种，它们构成耦合性的 7 种类型，如图 1-2 所示。

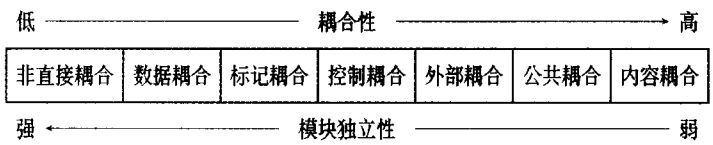

图 1-2　模块之间的耦合性

耦合是影响软件复杂程度的一个重要因素。在软件设计过程中，应尽量使用数据耦合，少用控制耦合，限制公共耦合的范围，完全不用内容耦合。表 1-2 列出了各类耦合性与模块各种属性的关系。

表 1-2 各类耦合性与模块各种属性的关系

	对修改的敏感性	可重用性	可修改性	可理解性
内容耦合	很强	很差	很差	很差
公共耦合	强	很差	中	很差
外部耦合	一般	很差	很差	中
控制耦合	一般	差	差	差
标记耦合	不一定	中	中	中
数据耦合	不一定	好	好	好
非直接耦合	好	好	好	好

1. 非直接耦合（Nondirective Coupling）

如果两个模块之间没有直接关系，它们之间的联系完全是通过主模块的控制和调用来实现的，这就是非直接耦合。这种耦合的模块独立性最强。

2. 数据耦合（Data Coupling）

如果一个模块访问另一个模块时，彼此之间是通过简单数据参数（不是控制参数、公共数据结构或外部变量）来交换输入、输出信息的，则称这种耦合为数据耦合。

3. 标记耦合（Stamp Coupling）

如果一组模块通过参数表传递记录信息，就是标记耦合。这个记录是某一数据结构的子结构，而不是简单变量。

4. 控制耦合（Control Coupling）

如果一个模块通过传送开关、标志、名字等控制信息，明显地控制选择另一模块的功能，就是控制耦合。

5. 外部耦合（External Coupling）

一组模块都访问同一全局简单变量而不是同一全局数据结构，而且不是通过参数表传递该全局变量的信息，称为外部耦合。

6. 公共耦合（Common Coupling）

若一组模块都访问同一个公共数据环境，则它们之间的耦合就称为公共耦合。公共的数据环境可以是全局数据结构、共享的通信区、内存的公共覆盖区等。

公共耦合的复杂程度随耦合模块的个数增加而显著增加。若只是两模块间有公共数据环境，则公共耦合有两种情况：松散公共耦合和紧密公共耦合。

7. 内容耦合（Content Coupling）

如果发生下列情形，两个模块之间就发生了内容耦合：

- 一个模块直接访问另一个模块的内部数据。
- 一个模块不通过正常入口转到另一模块内部。
- 两个模块有一部分程序代码重叠（只可能出现在汇编语言中）。
- 一个模块有多个入口。

1.1.2.3　深度、宽度、扇出与扇入

深度表示软件结构中控制的层数。如果层数过多则应考虑是否有许多管理模块过于简单了，能否适当合并。

宽度是软件结构中同一个层次上的模块总数的最大值。一般说来，宽度越大系统越复杂。对宽度影响最大的因素是模块的扇出。

一个模块的扇出是指该模块直接调用的下级模块的个数。扇出大表示模块的复杂度高，需要控制和协调过多的下级模块；但扇出过小（例如总是 1）也不好。扇出过大一般是因为缺乏中间层次，应该适当增加中间层次的控制模块。扇出太小时可以把下级模块进一步分解成若干个子功能模块，或者合并到它的上级模块中去。

一个模块的扇入是指直接调用该模块的上级模块的个数。扇入大表示模块的复用程度高。

设计良好的软件结构通常顶层扇出比较大，中间扇出较小，底层模块则有大扇入。

但我们也应当注意，不应为了单纯追求深度、宽度、扇出与扇入的理想化而违背模块独立原则，分解或合并模块必须符合问题结构。

1.1.2.4　作用域和控制域

模块的作用域是指受该模块内一个判定影响的所有模块的集合。模块的控制域是指该模块本身及被该模块直接或间接调用的所有模块的集合。软件设计时，模块的作用域应在控制域之内，作用域最好是做出判定的模块本身及它的直属下级模块。

1.1.2.5　功能的可预测性

功能可预测是指对相同的输入数据能产生相同的输出。软件设计时应保证模块的功能是可以预测的。

1.2　结构化设计方法

结构化设计方法是在模块化、自顶向下逐层细化、结构化程序设计等程序设计技术的基础上发展起来的。该方法实施的过程如下：

（1）总结出系统应有的功能，对一个功能，从功能完成的过程考虑，将各个过程列出，标识出过程转向和传递的数据。这样，可以将所有的过程都画出来。

（2）细化数据流。确定应该记录的数据。

（3）分析各过程之间的耦合关系，合理地进行模块划分以提高它们之间的内聚性。

1.2.1　系统结构图中的模块

模块就如同人的器官，具有特定的功能。人体中最出色的模块设计之一是手，手只有几种动作，却能做无限多的事情。人体中最糟糕的模块设计之一是嘴巴，嘴巴将最有价值但毫不相干的几种功能，如吃饭、说话、亲吻，混为一体，使之无法并行处理，真乃人类之不幸。

在系统结构图中，不能再分解的底层模块称为原子模块。如果一个软件系统的全部实

际加工都由原子模块来完成，而其他所有非原子模块仅仅执行控制或协调功能，这样的系统就是完全因子分解的系统。如果系统结构图是完全因子分解的，就是最好的系统。但实际上，这只是力图达到的目标，大多数系统做不到完全因子分解。

一般来说，结构图中可能出现如图 1-3 所示的 4 种类型的模块。

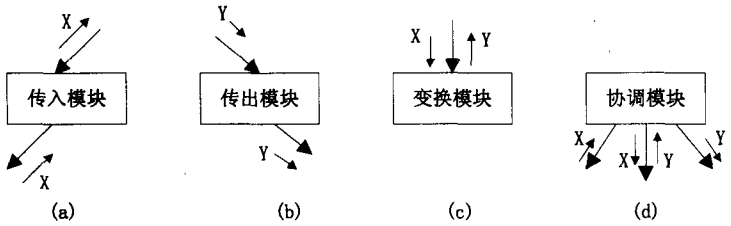

图 1-3 4 种模块类型

- 传入模块：如图 1-3（a）所示，从下属模块取得数据，经过某些处理，再将其传送给上级模块。它传送的数据流叫做逻辑输入数据流。
- 传出模块：如图 1-3（b）所示，从上级模块取得数据，进行某些处理，传送给下属模块。它传送的数据流叫做逻辑输出数据流。
- 变换模块：如图 1-3（c）所示，从上级模块取得数据，进行特定处理后，送回原上级模块。它加工的数据流叫做变换数据流。
- 协调模块：如图 1-3（d）所示，对其下属模块进行控制和管理的模块。在一个好的系统结构图中，协调模块应在较高层出现。

值得注意的是，结构图着重反映的是模块间的隶属关系，即模块间的调用关系和层次关系。它和程序流程图（常称为程序框图）有着本质的差别。程序流程图着重表达的是程序执行的顺序及执行顺序所依赖的条件。结构图则着眼于软件系统的总体结构，它并不涉及模块内部的细节，只考虑模块的作用，以及它和上、下级模块的关系。而程序流程图则用来表达执行程序的具体算法。

没有学过软件开发技术的人，一般习惯于使用流程图编写程序，往往在模块还未做划分、程序结构的层次尚未确定以前，便急于用流程图表达他们对程序的构想。这就像造一栋大楼，在尚未决定建筑面积和楼层有多少时，就已经开始砌砖了。这显然是不合适的。

Adele Goldberg 在《Succeeding with Objects》中叙述了一位犹太教教士在新年伊始的宗教集会上讲述的故事：

一位教士登上一列火车，由于他经常乘坐这辆车，因此列车长认识他。教士伸手到口袋中掏车票，但没有找到，他开始翻他的行李。列车长阻止了他："教士，我知道您肯定有车票。现在别急着找。等找到后再向我出示。"但教士仍在找那张车票。当列车长再次见到他时，教士说："你不明白。我知道你相信我有车票，但……我要去哪里呢？"

有太多项目失败就是因为它们没有明确的目标就开始了。

在结构化分析和设计技术中，通常存在着两种典型的问题类型：变换型问题和事务型问题。它们的数据流图和结构图都有明显的特征。下面分别讨论它们的数据流图形态及其映射成结构图的过程。

结构图（Structured Charts，简称 SC）是准确表达程序结构的图形表示方法，它能清楚