**软件工程复习知识点**

**第一章**

1.软件的分类：

按照功能的不同：系统软件、支撑软件、应用软件

基于规模不同：微型、小型、中型、大型、超大型软件

2.软件危机的主要表现：

（1）开发出来的软件产品不能满足用户的需求

1. 相比越来越廉价的硬件，软件代价过高
2. 软件质量难以得到保证，且难以发挥硬件潜能
3. 难以准确估计软件开发、维护的费用以及开发周期
4. 难于控制开发风险，开发速度赶不上市场变化
5. 软件产品修改维护困难，集成遗留系统更困难
6. 软件文档不完备，并且存在着文档内容与软件产品不符的情况

3. 软件危机出现的原因

（1）忽视软件开发前期的需求分析

（2）开发过程缺乏统一的、规范化的方法论的指导

（3）文档资料不齐全或不准确

（4）忽视与用户之间、开发组成员之间的交流

（5）忽视测试的重要性

（5）不重视维护或由于上述原因造成维护工作的困难

（6）从事软件开发的专业人员对这个产业认识不充分，缺乏经验

（7）没有完善的质量保证体系

4.软件不仅包括程序，还包括数据。

5．软件的特点

（1）软件是一种逻辑实体，具有抽象性

（2）软件的生产与硬件的制造不同，是人类智力劳动的产物

（3）软件在运行使用过程中，不会磨损

（4）软件的开发至今尚未完全摆脱手工艺的开发方式

（5）软件的开发和运行必须依附于特定的计算机系统环境

6. 软件工程的概念

IEEE对软件工程的定义为：

（1）将系统化、严格约束的、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件

1. 对（1）中所述方法的研究

具体说来，软件工程是以借鉴传统工程的原则、方法，以提高质量，降低成本为目的指导计算机软件开发和维护的工程学科。它是一种层次化的技术

7．软件工程研究的内容

（1）方法与技术：主要讨论软件开发各种方法及其工作模型

（2）工具与环境：软件工具为软件工程方法提供支持，研究计算机辅助软件工程，建立软件工程环境

（3）管理技术：对软件工程全过程的控制和管理

（4）标准与规范：使得各项工作有章可循，以保证软件生产效率和软件质量的提高

8. 软件工程目标和原则

软件工程要达到的基本目标包括：

达到要求的软件功能

取得较好的软件性能

开发出高质量的软件

付出较低的开发成本

需要较低的维护费用

能按时完成开发工作，及时交付使用

9. 软件工程的七条基本原则

用分阶段的生没周期计划进行严格的管理

坚持进行阶段评审

实行严格的产品控制

采用现代程序设计技术

软件工程结果应能清楚地审查

开发小组的人员应该少而精

承认不断改进软件工程实践的必要性

10．软件过程又称为软件生命周期过程，是软件生命周期内为达到一定目标而必须实施的一系列相关过程的集合

11. 软件产品的生存周期一般可以划分为6个阶段

可行性研究

需求分析

软件设计

编码

软件测试

软件维护

12. 常见的软件生命周期模型包括：瀑布模型、快速原型模型、增量模型、螺旋模型、喷泉模型、统一过程以及敏捷过程与极限编程

13.瀑布模型:

（1）线性开发模型，具有不可回朔性，文件驱动；

（2）瀑布模型的优点是过程模型简单，执行容易；缺点是无法适应变更；

（3）瀑布模型适应于具有以下特征的软件开发项目

在软件开发的过程中，需求不发生或发生很少变化，并且开发人员可以一次性获取到全部需求

软件开发人员具有丰富的经验，对软件应用领域很熟悉

软件项目的风险较低

14.快速原型模型

（1）快速原型模型是不带反馈环的，这正是这种过程模型的主要优点：软件产品的开发基本上是按线性顺序进行的

15.增量模型

（1）增量模型具有以下优点

将待开发的软件系统模块化，可以分批次地提交软件产品，使用户可以及时了解软件项目的进展

以组件为单位进行开发降低了软件开发的风险

开发顺序灵活

（2）增量模型的缺点是要求待开发的软件系统可以被模块化

（3）增量模型适用于具有以下特征的软件开发项目

软件产品可以分批次地进行交付

待开发的软件系统能够被模块化

软件开发人员对应用领域不熟悉，难以一次性地进行系统开发

项目管理人员把握全局的水平较高

16.螺旋模型

（1）螺旋模型的基本思想是，使用原型及其他方法来尽量降低风险，在每个阶段之前都增加了风险分析过程的快速原型模型

（2）螺旋模型有许多优点：

对可选方案和约束条件的强调有利于已有软件的重用，也有助于把软件质量作为软件开发的一个重要目标；

减少了过多测试或测试不足所带来的风险；

在螺旋模型中维护只是模型的另一个周期，在维护和开发之间并没有本质区别

（3）螺旋模型主要适用于内部开发的大规模软件项目

（4）通常用快速构建原型的方法来排除风险，如果风险不能排除，则停止开发工作或大幅度的削减项目规模；如果成功地排除了所有的风险，则启动下一个开发步骤。

（5）螺旋模型是风险驱动的。

17.喷泉模型：

（1）喷泉模型体现了“迭代”和“无缝”的特性。

18．统一软件开发过程模型是基于迭代思想的软件开发模型。

阶段：先启、细化、构建、产品化，每一个阶段都是一个里程碑。

19.迭代的优点：

采用迭代的软件工程思想可以多次执行各个工作流程，有利于更好地理解需求，设计出合理的系统构架，并最终交付一系列渐趋完善的成果；迭代是一次完整的经过所有工作流程的过程。

20.敏捷软件开发宣言

（1）“个体和交互”胜过“过程和工具”

（2）“可以使用的软件”胜过“面面俱到的文档”

（3）“客户合作”胜过“合同谈判”

（4）“响应变化”胜过“遵循计划”

21.极限编程的有效实践：

客户作为开发团队的成员;

使用用户素材;短交付周期;

验收测试;结对编程;

测试驱动开发;集体所有;

持续集成;

可持续的开发速度;

开放的工作空间;

及时调整计划;

简单的设计;

代码重构;

使用隐喻

22．常见的软件开发方法

结构化方法

面向数据结构方法

面向对象方法

形式化方法

问题分析法

可视化开发方法

23.软件工程工具通常有三种分类标准

按照功能划分：可视化建模工具、程序开发工具、自动化测试工具、文档编辑工具、配置管理工具、项目管理工具

按照支持的过程划分：设计工具、编程工具、维护工具

按照支持的范围划分：窄支持（工具）、较宽支持（工作台）和一般支持工具（环境）

**第二章**

1.项目立项：项目发起、项目论证、项目审核和项目立项

2.可行性研究（项目论证）的结论有三种情况：

可行，按计划进行

基本可行，需要对解决方案做出修改

不可行，终止项目

3. 可行性研究就是指在项目进行开发之前，根据项目发起文件和实际情况，对该项目是否能在特定的资源、时间等制约条件下完成做出评估，并且确定它是否值得去开发。可行性研究的目的不在于问题如何被解决，而在于确定问题是否值得去解决，是否能够解决。

4.可行性研究的内容

战略可行性研究

操作可行性研究

计划可行性研究

技术可行性研究

社会可行性研究

市场可行性研究

经济可行性研究

风险可行性研究

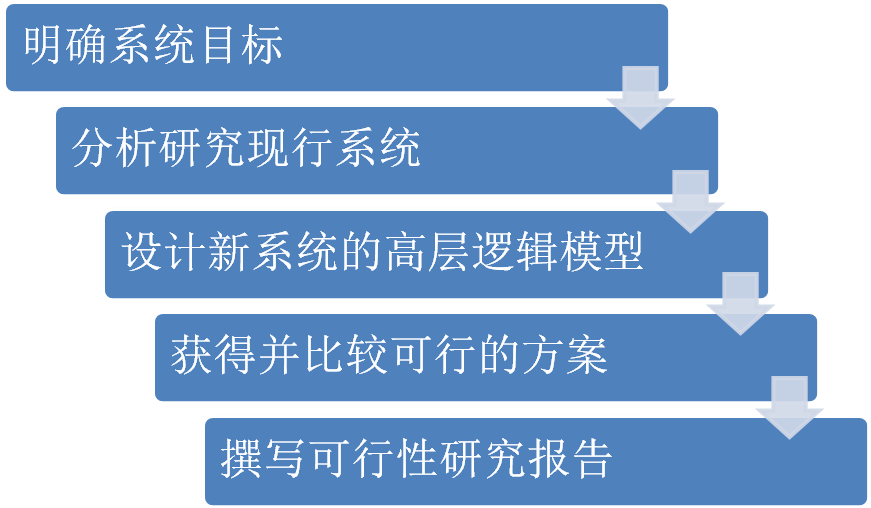
5.从三方面对现有系统进行分析：系统组织结构定义、系统处理流程分析、系统数据流分析。

系统组织结构定义：组织结构图

系统处理流程分析：系统流程图

系统数据流分析：数据流图和数据字典

6. 可行性研究的步骤：



7.需求分析的任务

需求分析的过程中应该遵守一些原则

需求分析是一个过程，它应该贯穿于系统的整个生存周期中

需求分析应该是一个迭代的过程

需求的表述应该具体、清晰，并且是可测量的、可实现的

需求分析主要有两个任务：

建模阶段：建立起系统的分析模型

描述阶段：把需求文档化，用软件需求规格说明书的方式把需求表达出来

8. 需求分析的步骤：

1. 需求获取：收集并明确用户需求的过程

获取需求的方法有多种，比如问卷调查、访谈、实地操作、建立原型等

（2）分析建模

常用的建模方法有数据流图、实体关系图、状态转换图、控制流图、用例图、类图、对象图等

（3）需求描述：编制需求分析阶段的文档

复杂的软件系统：需求阶段产生系统定义文档（用户需求报告）、系统需求文档（系统需求规格说明书）、软件需求文档（软件需求规格说明书）

简单的软件系统：需求阶段只需要输出软件需求文档

（4）需求验证：确保需求的一致性、完整性、现实性、有效性

9.需求管理

**用于查找、记录、组织和跟踪系统需求变更的系统化方法。可用于：**

获取、组织和记录系统需求；

使客户和项目团队在系统变更需求上达成并保持一致

**需求管理涉及三个主要问题 :**

识别、分类、组织需求，并为需求建立文档；

需求变化

需求的可跟踪性

10.结构化分析的具体步骤

建立当前系统的“具体模型”

抽象出当前系统的逻辑模型

建立目标系统的逻辑模型

考虑人机界面和其他一些问题

11. 结构化分析中经常用到的建模方法

数据流图（DFD）

实体关系图（ER）

控制流图（CFD）

状态转换图（STD）

12.分析模型的核心是“数据字典”，它描述软件使用或产生的所有数据对象。

13.ER图：联系分为3种：1:1,1:N,M:N

14.数据流图符号

正方形（或立方体）表示数据的源点或终点

圆角矩形（或圆形）代表变换数据的处理

开口矩形（或两条平行横线）代表数据存储

箭头表示数据流

15．数据流图有4种成分：源点或终点、处理、数据存储、数据流

16.状态转换图：

* 状态转换图（简称状态图）通过描绘系统的状态及引起系统状态转换的事件，来表示系统的行为
* 状态图指出了作为特定事件的结果系统将做哪些动作
* 状态图提供了行为建模机制

17. 状态

* 初态 （唯一）
* 终态 （0至多个）
* 中间状态

18. 符号

* 初态：实心圆
* 终态：一对同心圆
* 中间状态：圆角矩形

19.数据字典：

* 描述数据的信息的集合，对系统中使用的所有数据元素的定义的集合

20. 绝大多数数据字典都包含下列信息

* + 名字
  + 别名
  + 使用地点与方式
  + 内容描述
  + 补充信息

21. 软件设计的意义和目标

* 软件设计在软件开发过程中处于核心地位，它是保证质量的关键步骤

22. 指导评价良好设计演化的3个特征 ：

* + 设计必须实现所有包含在分析模型中的明确需求，而且必须满足用户期望的所有隐含需求
  + 设计必须是可读的、可理解的指南
  + 设计必须提供软件的全貌，从实现的角度说明数据域、功能域和行为域

23. 软件设计分类

从活动任务来看

* 数据设计
* 体系结构设计
* 接口设计
* 构件设计
* 部署设计

从工程管理角度来看

* 概要设计
* 详细设计

24. 概要设计

* 目标系统的总体架构
* 每个模块的功能描述、数据接口描述及模块之间的调用关系
* 数据库、数据定义和数据结构等

25. 详细设计

* 过程设计：描述系统中每个模块的实现算法和细节
* 数据设计：对各模块所用到的数据结构的进一步细化
* 接口设计：软件系统各模块之间的关系或通信方式以及目标系统与外部系统之间的联系

26. 模块独立

* 开发具有独立功能而且和其他模块之间没有过多的相互作用的模块

27. 耦合

* 耦合是对各个模块之间互连程度的度量，不同模块之间互相依赖得越紧密则耦合程度越高
* 为了提高模块的独立性，应该尽量降低模块之间的耦合程度

28. 内聚

* 内聚是指模块内部各个元素之间彼此结合的紧密程度
* 模块的高内聚通常意味着低耦合
* 在软件设计时，应该尽量提高模块的内聚程度

29. 启发规则

* 改进软件结构提高模块独立性
* 模块规模应该适中
* 深度、宽度、扇出和扇入都应适当
* 模块的作用域应该在控制域之内
* 力争降低模块接口的复杂程度
* 设计单入口单出口的模块
* 模块功能应该可以预测

30. HIPO图是美国IBM公司发明的“层次图加输入/处理/输出图”的英文缩写

31. 结构图

* 一个方框代表一个模块，框内注明模块的名字或主要功能
* 方框之间的箭头（或直线）表示模块的调用关系
* 带注释的箭头表示模块调用过程中来回传递的信息
* 箭头尾部是空心圆表示传递的是数据，实心圆表示传递的是控制信息

32. 面向数据流的设计方法

* 依据一定的映射规则，将需求分析阶段得到的数据描述从系统的输入端到输出端所经历的一系列变换或处理的数据流图转换为目标系统的结构描述

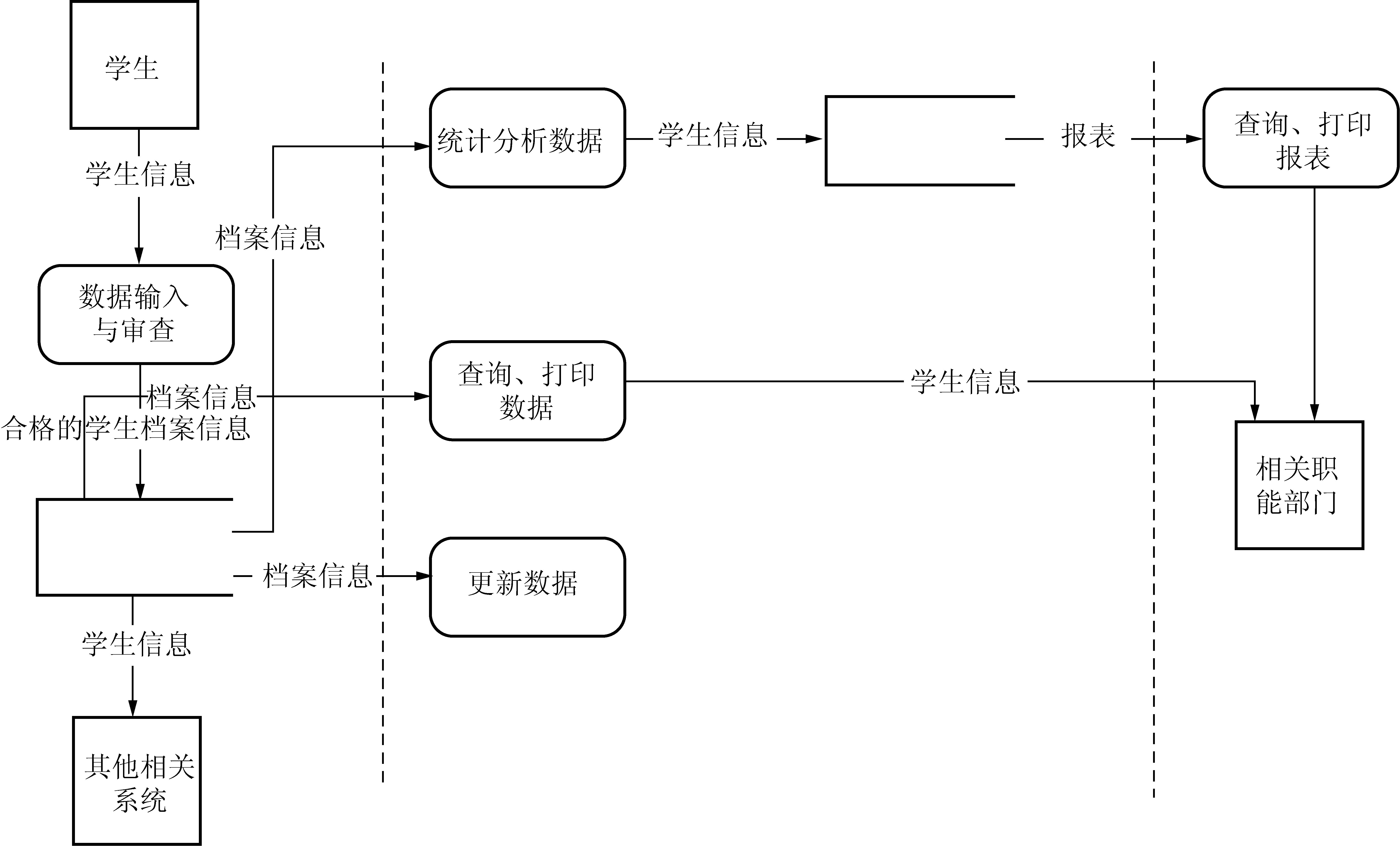
33.数据流

* + 变换型数据流
  + 事务性数据流

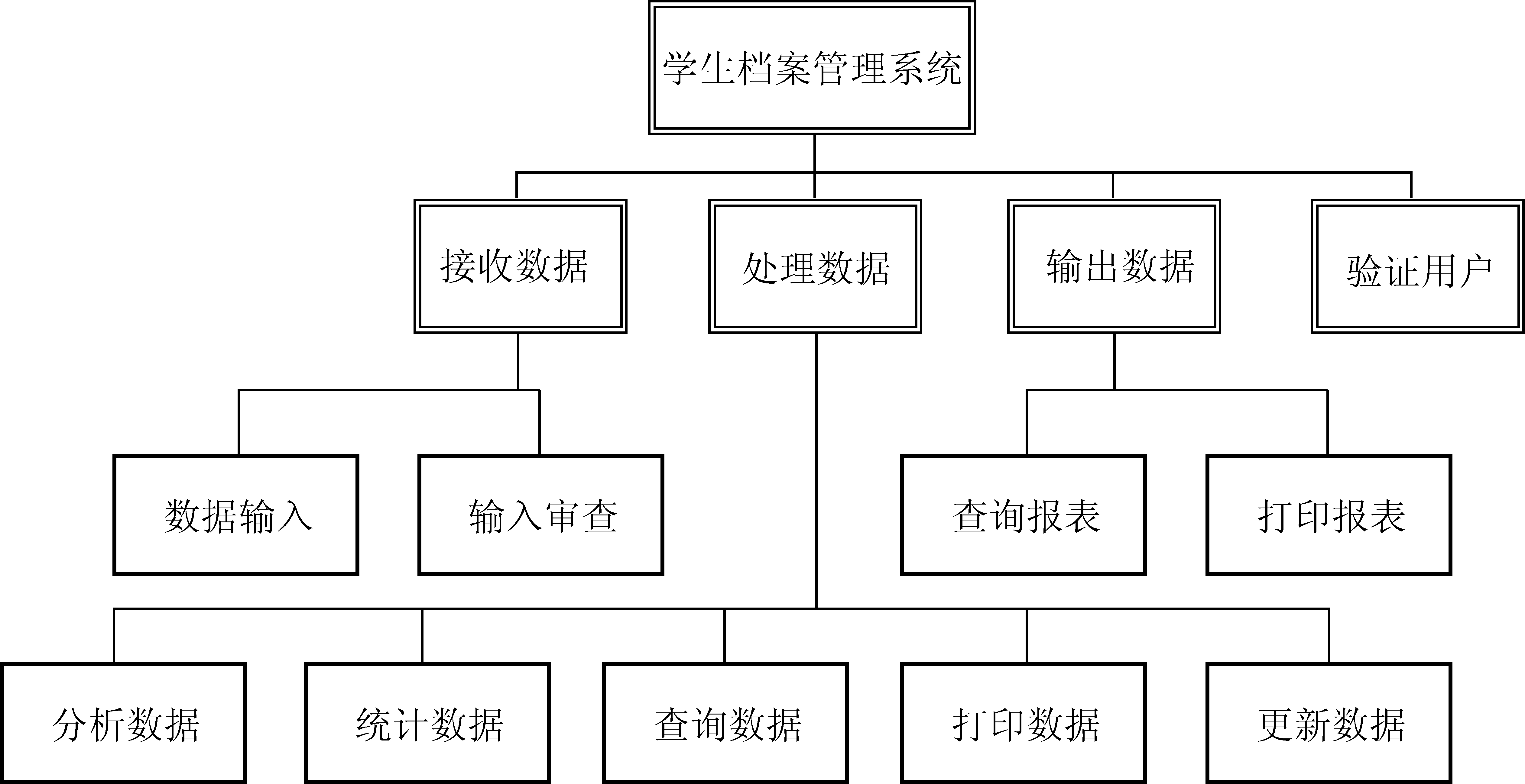
34. 针对变换型数据流的设计可以分为以下几个步骤

* + 区分变换型数据流中的输入数据、变换中心和输出数据，并在数据流图上用虚线标明分界线
  + 分析得到系统的初始结构图

对系统结构图进行优化

35. 

**学生档案管理系统的具有边界的数据流图**



**学生档案管理系统优化的系统结构图**

36. 针对事务型数据流的设计可以分为以下几个步骤

* + 确定以事务为中心的结构，找出事务中心、接收数据、处理路径三个部分
  + 将数据流图转换为初始的系统结构图
  + 分解和细化接收分支和处理分支

37.数据元素的逻辑关系：顺序、选择、重复

38. Jackson图的优点

* 便于表示层次结构，而且是对结构进行自顶向下分解的有力工具
* 形象直观可读性好
* 既能表示数据结构也能表示程序结构（因为结构程序设计也只使用上述3种基本结构）

39. 程序流程图

顺序型、选择型、先判定型循环（WHILE-DO）、后判定型循环（DO-WHILE）和多分支选择型

40. 程序流程图的主要优点

* 采用简单规范的符号，画法简单
* 结构清晰，逻辑性强
* 便于描述，容易理解

41.程序流程图的主要缺点

* 不利于逐步求精的设计
* 图中可用箭头随意地对控制进行转移，与结构化程序设计精神相悖
* 不易于表示系统中所含的数据结构
* 当目标系统比较复杂时，流程图会变得很繁杂、不清晰

41. 盒图（N-S图）的主要特点：

* + 不允许随意的控制转移，有利于严格的结构化程序设计
  + 可以很方便地确定一个特定控制结构的作用域，以及局部数据和全局数据的作用域
  + 可以很方便地表示嵌套关系以及模块之间的层次关系

42. PAD图的主要特点

* 表示的程序结构的执行顺序是自最左边的竖线的上端开始，自上而下，自左向右
* 表示的程序片断结构清晰、层次分明
* 支持自顶向下、逐步求精的设计方法
* 只能用于结构化的程序设计
* 不仅可以表示程序逻辑，还能表示数据结构

43. 判定表

* 判定表能够清晰地表示复杂的条件组合与应做的动作之间的对应关系

44.判定表的组成

* 左上部列出所有条件
* 左下部是所有可能做的动作
* 右上部是表示各种条件组合的一个矩阵
* 右下部是和每种条件组合相对应的动作

45. 判定树

* 判定树是判定表的变种，也能清晰地表示复杂的条件组合与应做的动作之间的对应关系

46.判定树的优点在于，它的形式简单到不需任何说明，一眼就可以看出其含义，因此易于掌握和使用

47. 过程设计语言（PDL，伪码）

PDL应该具有下述特点

关键字的固定语法，它提供了结构化控制结构、数据说明和模块化的特点

自然语言的自由语法，它描述处理特点

数据说明的手段

模块定义和调用的技术，应该提供各种接口描述模式

48. PDL的优点

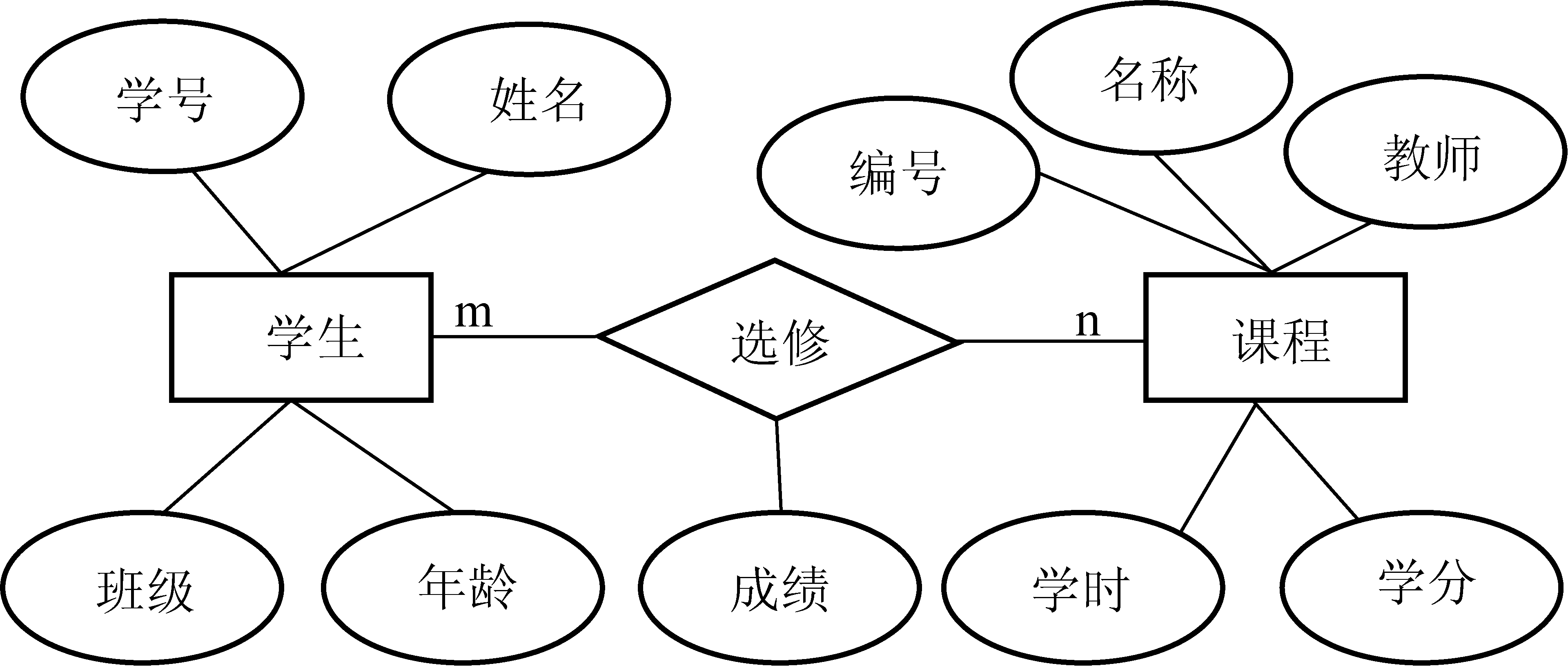
* + 可以作为注释直接插在源程序中间
  + 可以使用普通的正文编辑程序或文字处理系统，很方便地完成PDL的书写和编辑工作
  + 已经有自动处理程序存在，而且可以自动由PDL生成程序代码

49. PDL的缺点

* + 不如图形工具形象直观
  + 描述复杂的条件组合与动作间的对应关系时，不如判定表清晰简单

50. 数据库结构设计包括概念结构设计、逻辑结构设计和物理结构设计

* + 概念结构：描述系统最基础的数据结构
  + 逻辑结构：提供比较接近数据库内部构造的逻辑描述
  + 物理结构：数据库的物理数据模型

51. 

“学生”与“课程”的E-R图

52. 映射规则

* 将数据库概念结构中的“实体”映射为逻辑结构中的“数据表”，实体的属性可以用数据表中的字段来表示，实体的主关键字作为数据表的主键
* 数据库概念结构中的1:1联系可以映射为一个独立的数据表，也可以与跟它相连的任意一端或两端的实体合并组成数据表
* 数据库结构中的1:n联系可以映射为一个独立的数据表，也可以与跟它相连的n端实体合并组成数据表
* 数据库结构中的m:n关系可以映射为一个独立的数据表，各实体的主关键字的组合形成数据表的主键

53. 关系型数据库范式

* 第一范式：所有关系中的每一个分量都必须是不可分的数据项。第一范式是建立关系型数据表的最基本的要求
* 第二范式：满足第一范式的条件，并且每个非键属性都由整个键决定
* 第三范式：满足第二范式的条件，并且每个非键属性不依赖于其他非键属性值

54. 界面设计的“黄金原则”：

置用户于控制之下

减少用户的记忆负担

保持界面一致

55.界面设计的核心活动

创建系统功能的外部模型

确定为完成此系统功能人和计算机应分别完成的任务

考虑界面设计中的典型问题

借助CASE工具构造界面原型

实现设计模型

评估界面质量

56. 界面设计的4个模型 ：

* 由软件工程师创建的设计模型
* 由人机工程师（或软件工程师）创建的用户模型
* 终端用户对未来系统的假想
* 系统实现后得到的系统映象

57.界面设计中应该考虑的问题

* 系统响应时间
* 用户求助机制
* 出错信息
* 命令方式