系统分析与设计

项目经理级的分析员

一、系统开发生命周期(SDLC)的五个阶段

1、计划阶段的主要目标是确定新系统的作用域、确保项目的可行性、制定进度表、分配资源并进行项目其余部分的预算。五个活动：

A、定义问题
B、确认项目的可行性
C、制定项目的进度表
D、为项目安排人员
E、启动项目

2、分析阶段的主要目标是了解新系统的商业需求和处理要求并制作书面文件。分析本质上就是一个发现过程，分析期间推动活动的关键词就是发现和理解。六个活动：

A、收集信息
B、确定系统需求
C、建立需求发现的原型
D、划分需求的优先级
E、产生并评估可替换方案
F、与管理人员一起审查建议

3、设计阶段的目标是设计解决方案的系统。七活动：

A、设计并集成网络
B、设计应用程序结构
C、设计用户界面
D、设计系统界面
E、设计并集成数据库
F、设计细节的原型化
G、设计并集成系统控制

4、实施阶段是建立、测试和安装最后的系统。这个阶段的目标不仅是要有一个可靠的工作信息系统，而且是要确保培训所有的用户并使商业受益。六活动：

A、构造软件部件
B、检验与测试
C、开发调整原型
D、转换数据
E、培训与文档
F、安装系统

5、支持阶段的目标是在系统初始安装后的几年里保持系统有效地运行。这一的活动分为两类：

A、提供对最终用户的支持。
B、维护和增强计算机系统。

二、项目管理：组织和指导其他人员在预先确定的进度表和预算内实现计划的结果。

三、里程碑：是完成项目主要部分的一个重大事件，它可以通过一个专门的文件或一个专门的状态审查会议来确定。

四、确认项目的可行性：评价可行性的目标是决定开发项目是否有合理的成功机会。有五个方面的风险：

A、经济可行性
B、组织上和文化上的可行性
C、技术可行性
D、进度表可行性
E、资源可行性

五、PERT 图和甘特图之间的区别：

PERT/CPM 图：基于单个任务或活动对项目进行规划的一种方法。
甘特图：以条形图代表项目进度表的任务和活动。

六、关键路径：表明项目完成最短周期的 PERT 图上的路径。

系统开发方法

一、方法、模型、工具和技术

1、方法

系统开发方法：提供完成系统开发生命周期每一步的详细指导，包括具体的模型、工具和技术。

2、模型：现实世界某些重要方面的表示。

(1)系统组件的一些模型：

A、流程图
B、数据流图（DFD）
C、实体-联系图（ERD）
D、结构图
E、用例图
F、类图
G、顺序图

（2）用于管理系统开发过程的一些模型：

A、PERT 图
B、甘特图
C、组织层次图
D、财务分析模型——NPV、POI

3、工具：帮助生成项目中所需模型或其他组件的软件支持。

CASE 工具：用来帮助系统分析员完成系统开发任务而设计的计算机辅助系统工程工具。

4、技术：帮助分析员完成系统开发活动或任务的一组方法。一些技术：

A、战略计划技术
B、项目管理技术
C、用户面谈技术

D、数据建模技术
E、关系型数据库设计技术
F、结构化分析技术
G、结构化编程技术
H、软件测试技术
I、面向对象分析和设计技术

二、系统开发的三种方法

结构化方法：使用结构化编程、结构化分析和结构化设计技术的系统开发方法。

1、结构化程序：具有一个开始和一个结束的程序或程序模块，并且在程序执行中的每一步由三个部分之一：顺序、选择或循环结构。

2、自顶向下程序设计：把更复杂的程序分解为程序模块的层次图。

3、结构化设计：它为确定 下列事物提供指导，即：程序集是什么，每一个程序应该实现哪些功能以及如何把这些程序组成一张层次图。

结构图：在结构化设计中生成的显示程序模块层次的图形模型。

4、结构化分析：它帮助开发人员定义系统需要做什么，系统需要存储和使用哪些数据，系统需要什么样的输入和输出以及如何把这些功能结合在一起来完成任务。

数据流图：显示在结构化分析中产生的系统的输入、处理、存储和输出的图形模型。

实体——联系图（ERD）：系统所需数据的图形模型，其中包括在结构化分析和信息工程阶段生成的存储信息的事物以及这些事物之间的关系。

面向对象方法：系统开发的一种方法，这种方法把信息系统看作是一起工作来完成某项任务的相互件作用的对象集合。

1、对象：计算机系统中可以对消息做出响应的事物。

2、面向对象分析（OOA）：定义在系统中工作的所有类型的对象，并显示这些对象如何通过相互作用来完成任务。

3、面向对象设计（OOD）：定义和系统中人机进行通讯所必须的所有类型对象，并对每一种类型的对象进行细化，以便一种具体的语言或环境来实现这些对象。

4、面向对象编程（OOP）：用某种编程语言书写语句来定义各类对象的行为，包括对象间的消息传递。

调查系统需求

一、系统需求：是新系统必须完成的功能，是系统提
功能的详细定义。分为功能和技术需求：

A、 功能需求：是系统必须完成的活动，也就是系统将要投入的商业应用，描述系统必须支持的功能和过程的系统需求。

B、 技术需求：是和组织的环境、硬件和软件有关的所有操作目标，描述操作环境和性能目标的系统需求。

二、系统相关者：

（1）用户，即每天实际使用系统的人

（2）客户，即支付和拥有系统的人

（3）技术人员，即确保系统在组织的计算机环境下运行的人。

三、进行系统调查时能够提供指导的三个主要问题：

A、商业过程和动作是什么？（也就是提问用户，“你要干什么”）

B、商业过程应该怎样完成?（也就是提问用户，“怎样完成它”或“需要哪些步骤”）

C、需求信息是什么？（也就是提问用户，“为了实现系统你需要哪些信息”）

四、六种寻找事实的技术是：

A、向系统相关者分发和收集调查表

B、复查现有报表、表格和过程描述

C、主持与用户的面谈和讨论

D、观察商业过程和工作流

E、建立原型

F、主持 JAD 会议

五、原型：根本思想是更大、更复杂裸的一个最初的、可以运转的模型。

六、原型的下列特性将有助于项目成员开发出有效的原型：

A、可操作性：通常，一个原型应该是一个能运转的传奇，而重点是可运行性。

B、集中性：为了测试一个具体概念或者验证一种方法，一个原型应该集中于单一的餐目标。

C、快速性：我们需要一些诸如 CASE 的工具以便快速地建立和更改原型。

七、实体模型

最终产品的一个样例，这个样例只能进行观察发而不能实际执行。

八、联合应用设计

是用于加快系统需求调查的一种技术，即通过让所有相关的人一起参加某个单一会议来定义需求或设计系统。

九、结构化遍历

是指对调查结果和根据这些结果建立的原型进行复查。结构化遍历的目标是发现错误和问题。其基本思想是在理解系统需求的过程中建立需求文档，然后检查其中是否存在错误、遗漏及不一致之处。结构化遍历的四个要素：what、when、who 和 why。

A、what 和 when：在结构化遍历期间，需要复查的第一项是作为分析阶段的一部分而生成的文档。

B、who：遍历中包含的双方是自己的工作需要被复查的人和复查工作的人。

C、How：就像面谈一样，在结构遍历中，准备、执行、后续工作是必不可少的。

十、联合应用程序设计是这样一项技术，它通过和所有的关键参加者举行几次马拉松式的会议来加速系统需求调查进程。讨论立即导致需求定义和政策性决定，

而没有和分散的小组举行面谈和谋略消除分歧的延误。如果正确地使用这项技术，那么它将是一种强大有效的工具。

十一、业务流程再造（BPR）正在成为改善商业过程的一种广泛使用的技术。它要求对商业过程进行彻底地重新设计。使用 BPR，新的系统开发不仅仅是实现现有过程的自动化，而且要完全重新考虑整个设计过程。BPR 的目标是以一种新的方法使用 IT，从而实现效率和服务水平的巨大改进。由于系统分析员所且有的特殊的分析问题、解决问题和建模能力，因此他们通常在所有的 BPR 开发中都扮演着重要角色。

系统需求建模 事件和事物

一、建模在系统开发过程中重要的几个原因：

A、在建模过程中了解信息
B、通过抽象降低复杂性
C、有助于回忆所有的细节
D、有助于和其他开发小组成员进行交流
E、有助于和各种用户以及系统相关者进行交流
F、为以后的维护和升级提供了文档

二、模型的种类包括：数学模型、描述模型、图形模型

A、数学模型：描述系统技术方面的一系列公式。
B、描述模型：描述系统某一方面的描述性的备忘录、报表或列表。
C、图表和系统某些方面的示意性表示。

三、事件和系统需求

1、事件可以描述的、值得记录的在某一特定时间和地点发生的事情。
2、三种事件类型：外部事件、临时事件、状态事件
A、外部事件：系统之外发生的事件，通常都是由外部实体或动作参与者触发的。
B、临时事件：是由于达到某一时刻所发生的事件 。
C、状态事件：当系统内部发生了需要处理的情况时所引发的事件。
3、系统控制：为保证系统完整性而加入的防范和安全程序。
4、事件表：以各个事件为行、各个事件的关键信息为列。
A、触发器:用来通知系统某一事件发生了，这一事件可以是需要处理的数据到达了或到了一个时间点。
B、来源：为系统提供数据的外部实体或参与者。
C、动作：当某一事件发生时系统执行的操作。
D、响应：系统产生的一个输出结果，该结果将被选到某个目的地。
E、目的地：接收系统输出数据的外部实体或参与者。

四、事物和系统需求

1、数据实体：系统需要存储的有关信息系统传统开发方法的信息。
2、

需求的传统描述方法

一、传统方法

系统是过程的集合、过程与数据实体交互、过程接受输入并产生输出。

二、面向对象方法

系统是交互对象的集合、对象与人或其他对象交互对象发送与响应信息。

三、数据流程图（DFD）

是一种图形化的系统模型，它在一张图中展示信息系统的主要需求，即：输入、输出、过程和数据存储。

A、外部实体：在系统边界之外的个人或组织，它提供数据输入或接受数据输出。

B、过程：在 DFD 中的一个符号，它代表从数据输入转换到数据输出的算法或程序。

C、数据流：在 DFD 中的箭头，它表示在过程、数据存储和外部实体之间的数据移动。

D、数据存储：保存数据的地方，以便将来由一个或多个过程来访问这些数据。

四、抽象水平：能把系统分解成一个逐渐细化的分层集合的建模技术。

五、关联图

指描述系统高层结构的 DFD。所有的外部实体和进出系统的数据流都画在一张图中，并且整个系统被表示成一个过程。

六、DFD 片段

是为事件清单（扩展为事件表）中的每个事件创建的。用一个过程符号表示系统响应一个事件的 DFD。

七、事件划分的系统模型（0 层图）

一个为系统需求建立模型的 DFD，建模过程中对应于系统或子系统中每个事件使用单个过程。

八、评估 DFD 质量

1、复杂性最小化：人们对复杂的信息处理有局限性的。当太多的信息同时出现时，人们把这种现象叫做信息超量。

A、7+-2 规则：模型设计规则，它限制模型中组成元素个数或元素之间的连接数不能超过 9。

2、接口最小化：是与 7+-2 规则直接相关的。该原则通过使模型中各个元素之间的接口数或连接数最小化来达到简单的目的。

3、数据流一致性：分析员通过查找 DFD 中各种类型的不一致性可以发现错误或忽略的东西。以下是三个经常发生且易判别的一致性错误：

A、一个过程和它的过程分解在数据流内容中有差别

B、有数据流出但没有相应的数据流入。

C、有数据流入但没有相应的数据流出。

4、平衡：进出过程的数据流与进出过程分解 DFD 的数据流在数据内容上一致。

九、详细记录 DFD 部件

1、过程描述

A、结构化英语：一种收书写过程规范的方法，它将结构化编程技术和叙

述性英语结合起来。

B、决策表：一种处理逻辑的表格表示方法，其中包括决策变量、决策变量值、参与者或公式。

C、决策树：使用像树枝一样的线条对过程逻辑进行图形化的描述。

十、DFD 总结

传统分析模型的各个组成部分：实体-联系图、数据流图、过程定义图和数据定义。这四个组成部分构成了大多数系统一系列相互联锁的详细说明。数据流图提供对系统的最高层的考察，在这里结合了过程、外部实体、数据存储和数据流。其他的组成部分更详细地描述了数据流图的某些方面。

十一、信息工程模型

1、 四个阶段：系统规划、业务领域分析、系统构建、系统设计。

2、 过程分解和依赖模型

A、过程分解图：在不同的抽象层表示过程之间层次关系的一种模型。

B、过程依赖图：用存储实体描述过程顺序和交互的一中模型。

十二、工作流建模

工作流：是处理步骤的序列，这些事务步骤全面处理一个业务事务或客户请求。通过处理活动的控制流，它在人、组织、计算机程序和特定的处理步骤之间移动。

面向对象的需求描述方法

一、面向对象的需求

五个分离而又互相联系的面向对象的模型或图被用于从面向对象的角度出发来定义应用需求。这五种图是类图、用例图、协作图、顺序图、状态图。

A、 类图：目的是识别组成新系统的对象并进行分类。

B、 用例图：一种用以显示不同的用户角色和这些用户角色如何使用系统的图，目的是识别新系统的“使用”，或用例，换句话说，就是识别如何使用系统。

C、 协作图：一种用以显示对象如何被协调在一起以执行用例的图，目的是识别协作完成给定业务功能的对象。

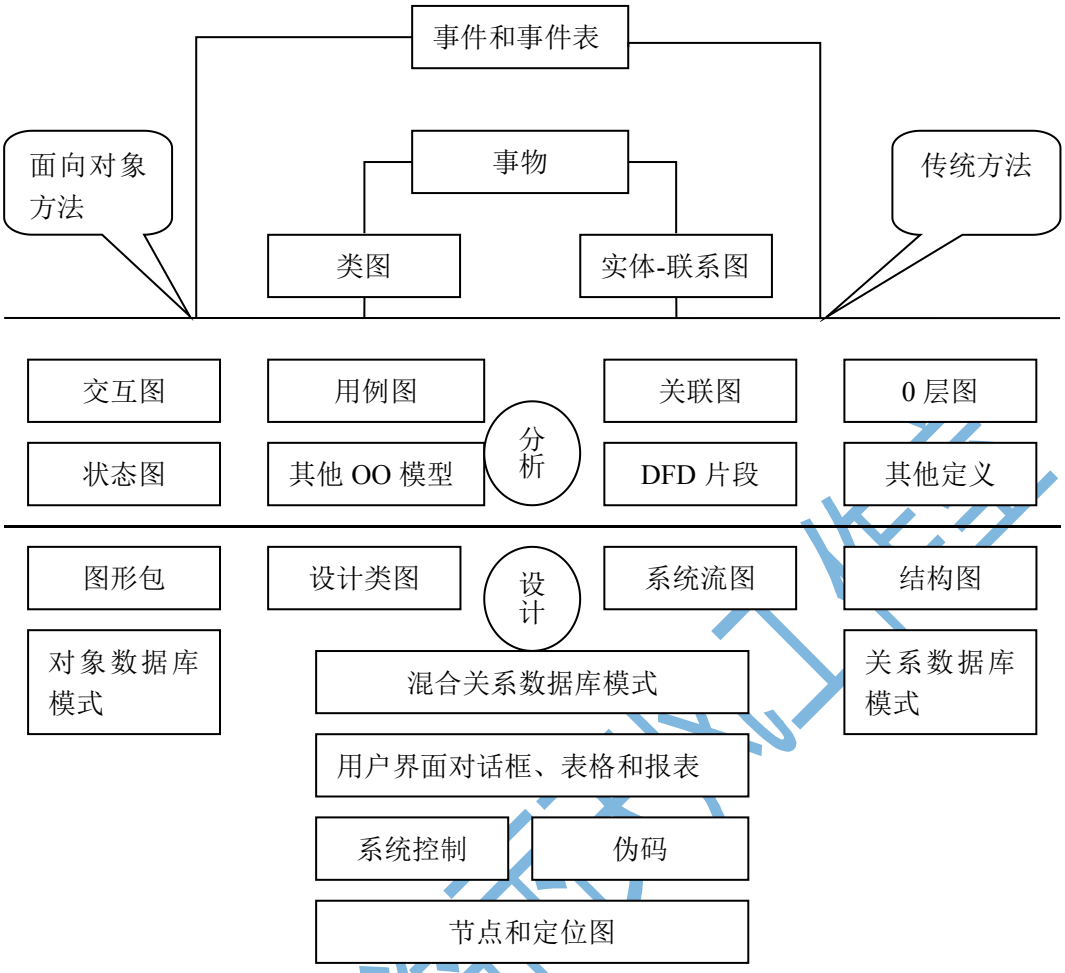
D、 顺序图：与协作图所表达的信息是一样的，只是显示方式有些差异，一种用以显示用例对象之间消息顺序的图。

E、状态图：一种用以显示对象在生命周期和转换期情况的图。

F、消息：用例内部对象之间的通信。

G、 交互图：显示对象之间交互的图，它或者是一个协作图，或者是一个顺序图。

二、OO 需求=事件表+类图+用例图+交互图（协作图/顺序图）+状态图表。



三、类图

类图说明系统的组成部分是什么，而其他图说明这些组成部分干什么。类不仅定义了数据库的结构，也定义了计算机程序的结构。

四、系统行为：面向对象的用例/场景视图

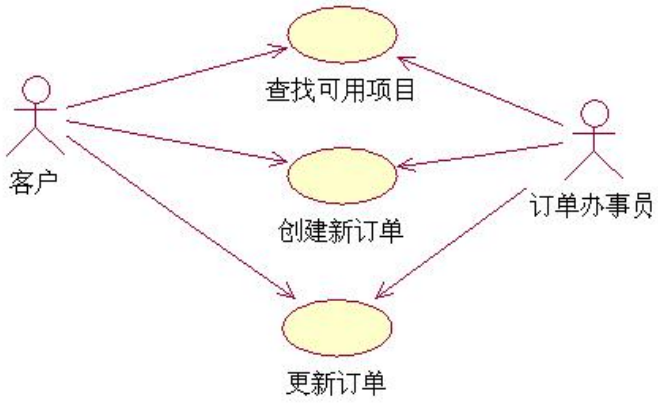
1、用例：描述系统在对事件做出响应时所采取的行动。由系统为使用该系统的用户完成的一个单一用途或功能。

2、参与者：系统用户扮演的一个角色。

3、场景：在用例中活动的一个特定顺序；一个用例可能有多个不同的场景。

五、用例图

用例图是概括有关参与者和用例信息的一个图形化模型。为了对用例进行分析，我们把系统作为一个整体，并且设法辨别系统的所有主要用途。



六、用例图与结构化技术的比较:

用例图的目标是提供一个系统的概览，包括使用这个系统的参与者和这个系统执行的功能。从这个意义上来说，它定义了系统的范围，因此用例图与关联图类似。但是，单个用例在标识系统支持的功能方面来说更象是一个 DFD 片段。

结构化建模和面向对象建模的一个主要差别在于开发一个最初定义自动化边界的用例图的思维过程。在 DFD 的开发中，自动化边界只在所有的过程已被细化后才确定。所以，在 DFD 中外部实体总是信息的源和目的，而且它可以不必是一个与系统交互的实体。

另一个重要的差别是用例图不显示数据流。流进和流出系统的信息只有到下一层建模的交互图中才得以体现。

八、对象交互：顺序图与协作图

1、顺序图：顺序图展示对象之间的交互顺序，这些交互是指在场景或用例的事件流中发生的。

开发顺序图的一个有效方法其步骤如下：

A、 确定所有与场景有关的对象和参与者。

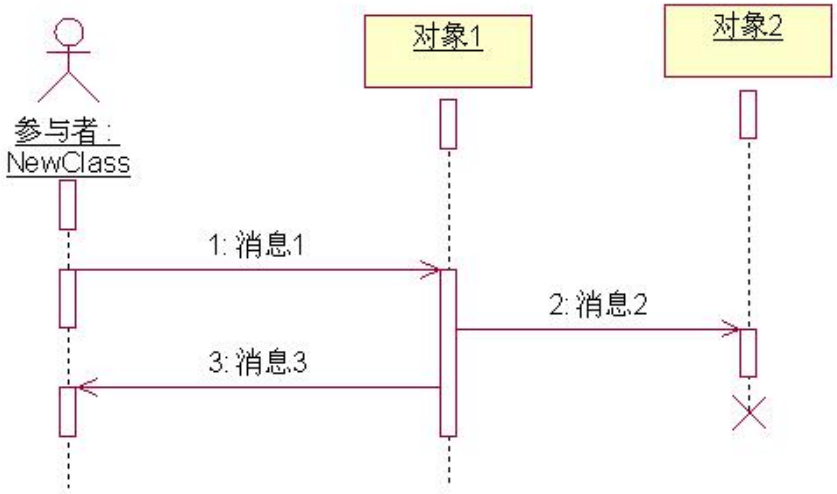
B、基于活动流，确定每一个需要用于完成场景的消息。

C、下一步决定是否总是发送还是有条件的发送每一个消息。

D、正确地为这些消息排序并把它们附在合适的参与者或对象的生命线上。

E、给消息加上形式化的语法以描述条件、消息名和要传递的参数。

F、如果你愿意，加上响应消息和通信以使顺序图完整。



2、协作图：主要应用是快速浏览相互协作，用来支持一个特定场景的所有对象。

九、对象行为：状态、状态转换和状态图表

1、状态：是指它在生命期中满足某些标准、执行一些行为、或等待一个事件时的存在条件。状态是对象的半永久条件。

2、动作：在一个特定状态下对象执行的行为。

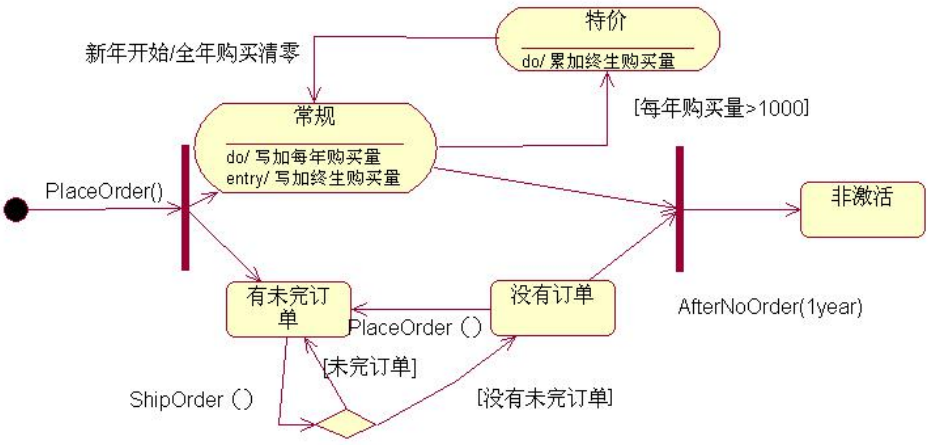
3、转换：状态图中的一个组成部分，它标示从一个状态到另一个状态的移动。

4、消息事件：转换的触发器，这个转换由一个有事件属性的消息组成。

5、行动表达式：对一个转换的陈述，以描述执行的动作。

6、内部转换：在一个状态内的转换，它不会从这个状态移开对象。

状态图：



十、帮助你开发状态图的一些步骤：

1、检查类图并选择出需要状态图的类。
2、标识所选类的全部顺序图的所有输入消息。
3、对一组中的每个所选择的类，为你能辨别的所有所有状态画一个列表。
4、建立状态图片段并把这些片段按正确的顺序排列。
5、回顾路径并查找独立、并行的路径。
6、使用适当的消息事件、保护条件和行动表达式扩展每一个转换。
7、检查和测试每一个状态图。

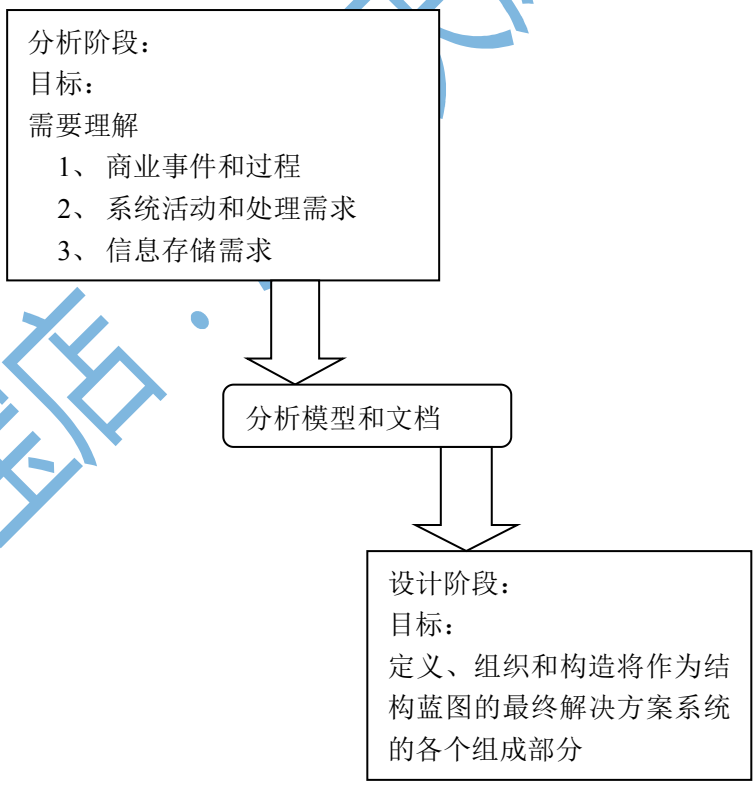
系统设计

一、理解设计要素

系统设计是一个描述、组织、构造系统部件的过程。这个过程分为两个层次：一是结构设计，一是细节设计。

1、输入：从分析到设计

不管对哪种分析方法而言，设计阶段的输入都是一系列已建好的文档和模型。



2、设计阶段主要组成部分和层次

A、 构架设计首先决定了整个结构，即高层设计，也叫总体设计或概念设计。

B、细节设计是低层设计，包括具体的程序细节设计。

3、输出：结构化模型和面向对象模型

设计的原始定义表明设计包含描述 、组织和构造，设计活动的输出是一系列满足目标的图和文档。这些图就是系统解决方案的各个方面的模型及其相应文档。

二、应用程序结构设计：结构化方法

1、模块：一个计算机程序的可标识部分，用来完成一个具体给定的功能。

2、计算机程序：由一系列模块组成的可执行的实体。

3、系统流程图：描述一个系统内计算机程序之间所有控制流的图。

4、伪码：与结构化编程类似的语句，它描述了模块的逻辑。

三、自动化系统边界

自动化系统边界将数据流程图处理划分成手工处理部分和系统处理部分。数据流可以在系统内部、外部或穿过系统边界和程序边界。穿过系统边界的数据流是格外重要的，它们代表了系统的输入和输出。换句话说，程序接口的设计（包括用户界面设计与其他系统的接口设计）是由穿过边界的数据流定义的。

四、系统流程图

系统流程图是对一些计算机程序、文件、数据库以及相关手工过程的计算机系统表达。

五、结构图

结构图：用来展示一个计算机程序模块间关系的层次图。结构图建立的规则和方针是：程序是分层的，且模块按高内聚、低耦合的方式组织在一起。

1、程序调用：控制从一个模块转换到下一层模块以便执行一个需要的服务。

2、数据耦合：在一个程序调用中，模块间传递的各个单独的数据项。

3、两种方法开发结构图：事务分析和转换分析

A、事务分析：基于 DFD 的结构图开发，用来描述多种事务类型的处理过程。

B、转换分析：基于 DFD 的结构图开发，用来描述输入-处理-输出数据流。

六、应用程序结构设计：面向对象方法

1、包图：是一个高层图，用以标识系统中的主要部件。

2、设计类：是类图的一变体，设计类图是带某些符号的类图，这些符号在类中描述了设计部件。

3、设计类图的开发：

A、决定需要设计的类。

B、找到属于这个类的所有方法。

C、详细描述带有逻辑的方法。

七、 结构化方法设计系统，输入的主要是数据流图。最初，数据流程图通过加入系统边界得到加强。系统边界指 0 层图，它用以显示一个完整的系统。它也是 DFD 片段的构架，在更低层显示程序边界。在数据流程图中，系统边界用转换分析或事务分析转换成结构图。事务分析用在高层 DFD 中，并用以收集大量的系统转换信息。转换分析可以在一个单独转换基础上开发一个结构图，该转换从它的输入到一个输出。

八、 面向对象设计使用三种分析图作为输入：类图、协作图和状态图。类图

可以扩展为包含了描述各个类及其方法中的变量的设计类图。设计过程从协作图中提取消息，并将它们加入到方法中。状态图用来为每个方法建立内部逻辑。

输入/输出和控制的设计

一、完整性控制

完整性控制是系统内部的机制和程序，用来保护系统和系统内的信息。

1、完整性控制的目标：

A、确保只有一个合适并正确的商业交易发生。

B、确保交易被正确地记录和处理。

C、保护组织的资产（包括信息）。

2、系统访问控制：确定谁有权访问系统及其数据的完整性控制。

3、输入完整性控制：是用来减少输入错误数据的一种补充的验证方法。

4、输出完整性控制：目的是确保输出到达正确的目的地，并且这些输出是正确的、精确的、通用的和完整的。

A、目的地控制：确保将输出住处输送到正确接受者的完整性控制。

B、完整性、精确性和正确性控制：输出信息的完整笥、精确性和正确性主要是系统内部处理功能，而非任何一组控制。

二、系统输入设计

当设计系统输入的时候，系统开发人员必须完成四个任务：

A、 确定将要用作输入的设备和机制。

B、确定所有的系统输入，并制定一个包括所有数据内容的列表。

C、对于每个系统输入，确定哪些控制是必需的。

D、 设计和规范电子表单（用户使用的窗口）及其他输入。

三、系统输出设计

系统输出的主要目的是在正确的地点和时间为正确的人提供相关信息。输出设计的任务实现下列四个目标：

A、 确定每个输出类型。

B、为应用设计所要求特定输出做一个列表。

C、提供必要的控制来保护输出中提供的信息。

D、 设计和规范报表格式。

1、确定输出类型

四种报表类型：详细报表、汇总报表、异常报表和决策报表。

A、详细报表：用来记录每天的商业处理过程，它们包含商业交易中的详细信息。

B、汇总报表：通常用来对周期性活动进行扼要重述，对一段时间内或某些种类的信息细节进行摘要或汇总的报表。

C、异常报表：仅包含非标准或异常、条件信息的报表。

D、决策报表：从通常用于战略决策的各种信息源得到的汇总报表。

2、内部输出：为了组织内部的使用而生成的打印报表和文件。

3、外部输出：为组织外部的使用而生成的打印文档，诸如声明、通知、标准信

函和法律文档。

4、返回文档：一个外部输出，其中包含作为输入返回系统的一部分。

5、信息过载：对用户提供大多信息发明奖没有提供组织和搜索信息的技术所造成的问题。

四、 设计者使用在分析和应用程序设计活动中建立的图表建立系统的一个输入列表。对结构化方法，可以使用 DFD、数据流定义和结构图。对于面向对象方法，顺序图是信息的主要来源。设计类图用来保证能提供产生输出的正确数据字段和正确方法。

五、 设计系统输出的过程与设计系统输入的过程有相同的四个步骤。对于输出设计，DFD 和顺序图被用来确定输出系统的数据流和信息。新的技术提供了许多用图表、图形和多媒体表示输出的方法。在确定一个输出媒体之前，设计者要仔细考虑使用系统的对象和输出目的。

快速应用开发和基于组件的开发

一、开发过程缓慢的原因

三大类原因：返工、需求的变化以及开发工具和技术的不足或不正确。

1、避免返工需要做两件事：

A、好的软件（并且只有合适的软件）必须是可构造或可获得的。

B、开发过程生产的软件必须至少满足最低的质量标准。

3、保证构造好的软件的惟一方法是保证在设计和构造开始之前就要充分地了解系统需求和全面的设计限制条件。

二、快速应用开发（RAD）：已被证实了的在某些情况下可以缩短开发进程的开发方法、工具以及技术的总和。

三、快速开发方法

1、最古老的软件开发方式是完全顺序化的。这种软件开发方式有其时代背景：

A、系统相对简单并且彼此独立。

B、计算机硬件资源非常昂贵。

C、软件开发工具相对原始。

D、在现在有较老的系统开发方式后面隐藏着如下假定：系统相对简单、容易分析和建模，因此有理由相信在设计和构造活动开始之前，需求能够被充分和正确地界定。

2、重新构建正确开发方式需要依靠：

A、项目规模。

B、在项目开始的时候或者可行性（不）确定的程度。

C、在项目周期中用户需求的预期变化率。

D、开发者对提出的实现技术所具有的经验和信心。

四、原型化开发方法

原型化是构建一个可以模仿真实系统的部分或者全部功能的系统模型的过程。一个原型是一个自我独立的系统。但是它可能不完善或者不能完全符合用户的最后要求。有两类原形系统

A、发现型原型：通常用在分析阶段，偶尔也会用在设计阶段，用来发现或

精练系统需求或者系统参数的原型系统。

B、发展型原型：被反复开发直到成为最终系统的原型系统。

1、什么时候使用原型化方式：

A、 一部分需求不能独立于体系结构或者详细设计而被完全确定。

B、一些系统功能的技术可行性不可知或者不确定。

C、原型化开发工具有足够的能力去构造一个功能完备的系统。

2、原型化设计的工具要求：成功的进行原型化设计需要系统开发者利用功能强大，灵活而且高效的工具。成功的原型化工具使用高度交互的方式进行应用程序开发，其他典型的技术和能力包括：

A、 WYSIWYG（所见即所得）的形式开发用户界面组件。

B、可以依据图形化模型或应用模板产生完整的应用程序、程序框架和数据库设计。

C、软件库或者组件的快速定制。

D、 复杂精密的纠错和调试能力。

3、原型化对每个项目都必须满足的条件,包括：

A、 对系统使用的技术环境的适应性。

B、实现所有系统要求的能力。

C、具有与用其他工具开发的软件相连接的能力。

五、螺旋形开发方法

螺旋形开发方法是一种循环反复的开发方法，其中每一次循环都由规划、分析、设计或实现几步构成。

1、在每一个循环中实现哪些特性是一个重要的决定。开发者应该怎样选择这些特性要依据一定的准则，包括：

A、 用户的优先级。

B、不确定的需求。

C、功能重用。

D、 实施风险。

2、螺旋形开发的益处与风险：螺旋形开发方法相对传统的开发方法和原型化开发方法有许多的益处，包括：

A、 高度的并行性：在各个原型循环内和循环之间，有许多进行重叠活动的机会。

B、 高度的用户参与性：使用者可以参与每一个计划、分析和测试阶段。

C、 循序渐进的资源负担：使用螺旋形的开发循环将可以使资源的消耗更平缓。

D、 频繁的产品交付：每一个原型都是一个可以独立工作的系统。

3、一个采用螺旋形设计方法的设计者要冒更大的风险进行决策，因为这种决策可能是全局次优的。这些问题是否显著依赖于许多因素，包括：

A、 系统功能的相互依赖性。

B、系统预期的生命期。

C、设计队伍的经验和技能。

D、 运气。

六、快速开发技术

1、风险管理：是辩识和减轻软件开发风险的系统过程。

A、如果对它们多加小心，大多数的风险是可以辩识的。
B、风险是随着系统的开发进程而出现、消失、增长或者减小的。
C、小的风险应该受到监控，大的风险应该被主动地降低。

2、基于工具的开发：选择与系统需求最匹配的工具进行开发的方法，不开发这些工具不容易实现的任何需求。

3、软件重用：指为一个目的开发的软件可以应用于另外一个目的的机制，叫软件重用，也叫做代码重用。

七、对象框架

对象框架：被指定设计用来在各种程序和系统中进行重用的类的集合。
基类：对象框架中的类。

1、对象框架的类型：
A、 用户界面类。
B、通用数据结构类。
C、关系数据库接口类。
D、 用于特定应用领域的类。

2、对象框架对于设计和实现任务的影响：
A、 框架必须在细节设计开始之前选定。
B、系统设计必须符合框架关于应用程序结构和操作的特殊假定。
C、设计和开发人员必须接受培训以便有效地使用框架。
D、 可能需要多个框架，需要尽早进行兼容性和集成性测试。

八、组件

组件：一个标准可以互换的、已经装配完成可以立即使用的软件模块。

1、基于组件的设计的构造与众不同的之处如下：
A、 组件是二进制的代码单元，重用源代码的方法是结构化设计和面向对象的继承。
B、组件作为可执行对象，公布一个公共接口，对其他的组件隐藏它们方法的实现。

软件包及企业资源计划

一、 集成：融合已有的独立的产品组件和子系统，并解决它们之间交互问题的过程。

二、 软件包

1、软件包的实施和支持
A、 有三方面的考虑对于软件包来说比较独特：专用性、集成性以及将来的升级。
B、 一个软件包作为一个大系统的一部分必须能够从其他的系统中获得数据，不管这个系统是专用的还是软件包，也要向其他的系统提供数据。
C、 毫无疑问，软件包将要有被软件供应商在生命周期中进行一次升级的可能。

2、软件包定制：由于工业和商业功能的实践以及大量的选项和特性往往已经内嵌其中，所以软件包通常被设计以后需要一点儿或一点也不需要定

制的形式。有三种类型的定制：配置、修改和增强。

A、 配置：指在一系列选项中进行选择，例如把软件模块装配成一个完整的系统或者设定软件包中的参数值。

B、 修改：修改程序代码以改变现有的进程形式。

C、 增强：通过给软件增加程序代码或者程序模块提供附加的功能。

3、软件包的集成

这里有两种基本的集成应用程序的方法：

（1） 使用中间件联接独立的软件包；

（2） 购买一个已经内嵌有中间件的综合 ERP 软件包。

企业应用集成（EAI）：为了对各个商业单位和 IT 系统的信息流的支持以连接应用程序的过程。

4、软件包的升级：软件升级的结果往往是提供附加功能或者提高易用性。

三、 企业资源计划

1、什么是 ERP：ERP 使用集成应用软件以提高企业效能和效率，最终增加企业竞争力的过程。

A、 价值链：在机构的内部所有增加产品和服务价值的动作顺序。

2、为什么要考虑 ERP

A、 通过减少人工操作以及重复性劳动，减少在核心事务处理中的工作强度。

B、 支持全球商业动作。

C、 通过替代分离的系统获得规模效益。

D、 通过使用软件包的解决方案减少信息系统的开发人员。

E、 通过更好的基于更加完善和便捷的后勤信息供应来改善客户服务。

F、 通过公用数据库提高数据一致性。

G、 通过及时的报告和传达能力发送决策支持。

3、实施 ERP

影响成功实施最关键的因素是：

A、 有力的高级管理层支持。

B、一个商业主管对项目的集中式管理。

C、有力的 IT 管理和人员保障。

D、 在项目的管理和实施中，用户要充分参与。

E、 业务流程再造的标准是要依照软件的能力而不是为业已存在的商业过程定制软件。

F、在 ERP 的实施和维护中要对现有系统的开发人员进行再培训。

G、 对新系统的终端用户进行额外培训。

H、 依靠咨询顾问去领导实施和培训。

I、 在组织内部，ERP 得到尊重和有效的拥护。

J、 参与 ERP 项目的各部分之间保持有效的不断交流。

K、 有丰富的商业知识和技术能力的顶级系统分析员。

L、在项目中要留住这些顶级分析员。

M、关注用户对新系统的抵触。

4、ERP 系统开发方法论

A、 第一阶段：计划 要建立一个监督委员会来启动项目。

B、第二阶段：分析 一旦分析阶段开始，监督委员会就要处理方案，有

可能选择一种实施方法，然而此时还不会选定供应商。

C、第三阶段：设计 当 ERP 的供应商最后敲定以后，设计阶段就开始了。

D、 第四阶段：实施 如果选择定制软件的方式，它的子阶段叫做构造。

E、第五阶段：支持 这个阶段的目的是确保新系统短期和长期的成功。

5、ERP 开发的特殊问题

A、 第一步：初步评估

（1） 主要功能。

（2） 软件包的前期费用，可以广泛的在供应商中询价，而且供应商经常很愿意协商。

（3） 软件包的运行支持费用。

（4） 供应商的长期支持，供应商可以存在多久。

（5） 软件包的商业模型和目前公司采用的实际商业动作之间的适应程度。

（6） 满意度，承诺的实施时间

（7） 一定规模的安装数量，以使公司还要成为试验品

（8） 有保障的本地支持服务。

（9） 有在国际范围的成功事例。

（10）能否得到集成第三方软件和旧有软件的定制工具。

（11）与公司的管理风格进行最好地融合。

B、第二步：细节评估

（1） 可按照不同的标准设定价格。

（2） 支持多种货币的财务信息。

（3） 控制销售代表的佣金

（4） 支持时限生产。

C、第三步：供应商展示

D、 第四步：实地考察

E、最后决定

制作可操作的系统（实施、转化和支持）

一、程序开发

1、系统实施的顺序

A、输入、处理、输出

B、自顶向下

C、自低向上

2、输入、处理、输出（IPO）的开发方法：这种顺序是基于一个系统或程序的数据流，首先开发包含外部输入的程序或模块，再开发处理这些输入数据的程序或模块，最后开发产生输出结果的程序或模块。

二、框架开发

构建一个大型的面向对象系统时，通常的做法是建立一个对象框架（或一系列的基类）它包括大多数或全部的应用程序中所包含的特定的类。基类

可以在系统的许多部分和不同的应用程序中被重用。正因为这样，它们是系统的关键组成部分，基类中的错误将会影响到系统的每一个程序，另外，后期对基类的更改将会使整个系统发生重大变化。

三、基于小组的程序开发

程序开发通常是一个小组的程序员一起工。使许多程序员同时开发系统的不同部分，可以缩短开发周期。然而，成组开发项目也会带来一些管理上的问题，如下所示：

A、编程组如何组织

B、特定的小组或成员如何分派任务

C、成员与小组间的交流与协作

这里有许多开发小组的组织方法，一些常用的组织模式包括：

A、同等合作小组：由有着大体相同技术、经验和相似专业背景的人员组成的开发小组。

B、首席开发者小组：小组只有一个领导者，所有的重大决策都由他做出。

C、协作专家小组：小组成员在技术和经验方面差异很大，很少有重叠。

四、源代码的控制

源代码控制系统（SCCS）：一种自动工具，用来跟踪记录源代码文件并的定制对这些文件的改动。

五、版本

1、α版本：是一个未完成的但是已准备好了接受严格测试的系统。

2、ß版本：是一个足够稳定的系统，可以接受终端用户的测试。

六、质量保证

质量保证是保证信息系统满足用户、技术人员和管理人员最低质量标准的过程。

七、技术复审

技术复审可以分开设计和构建的过程，为别人提供发现错误和提出建设性意见的机会；技术复审存在的范围不一，可以在组织和组织之间进行，也可以 在同一组织的不同项目之间；技术复审的形式有正式的，也有非正式的。

八、检查：一组开发人员对设计和构建细节的正式复审每个开发者都有明确的任务。

九、测试

测试是一个对产品进行检验以确定其存在哪些缺陷的过程。

1、测试实例：对开始状态、软件必须响应的一个或多个事件、期望得到的响应或结束状态等内容的正式描述。

2、测试数据：用于测试一个模块、一组模块或整个系统的一组开始状态和事件。

3、单元测试：这是在与其他模块进行集成测试之前，对单个代码模块进行测试的过程。

4、驱动程序：一种为单元而开发的模块，用来模仿尚未开发的模块的调用行为。

5、占位程序：一种为了测试而开发的模块，可模仿一个尚未瓦特 模块的执行或行为。

6、集成测试：是测试一组模块或方法的性能。集成测试的目的是要发现

单元测试不能发现的错误：

A、 接口不兼容。

B、参数值

C、运行例外。

D、 意外的状态交互。

7、系统测试：用来测试整个系统或独立子系统的性能。

8、验收测试：一种系统测试，以确定系统是否满足用户需求。

9、测试伙伴：负责测试另外一个程序员编写的程序代码的程序员。

十、安装

最常用的安装途径包括：

1、直接安装：一种使系统迅速运行起来的安装方法。

2、并行安装：是一种使新旧两个系统在很长的一段时间内同时运行的安装方法。

3、阶段安装：经过一系列的步骤和阶段使新系统投入运行的安装方法。

十一、文档

1、系统文档：描述系统功能 、结构和构造细节，用户是维护人员和未来的开发人员。

2、用户文档：描述如何使用和维护系统，用户是终端用户和系统操作员。

十二、维护

软件维护：在软件发布之后为了纠错、改善软件性能或属性或使软件产品适应环境变化而对软件进行的修改。

维护活动包括：

A、 跟踪修改请求和错误报告

B、实施一些改动

C、监视系统性能，并实施一些改动以提高系统的工作能力或改善系统性能。

D、 升级硬件和系统软件。

E、更新文档以反映出维护活动所作的改动。