

第三章 系统和基础建设生命周期管理 15p

★ 必须的知识

- 1.收益管理实务(例如:可行性研究、业务案例)
- 2.项目治理机制,如:项目指导委员会、项目监督委员会
- 3.项目管理实务、工具和控制框架
- 4.用于项目管理上的风险管理实务
- 5.项目成功的原则和风险
- 6.涉及开发、维护系统(和/或体系)的配置、变更和(发布的)版本管理
- 7.确保 IT 系统应用的交易和数据的完整性、准确性、有效性和授权的控制目标和技术
- 8.关于数据、应用和技术的企业框架
- 9.需求分析和管理实务,如:需求验证、跟踪(可追溯)、差距分析
- 10.采购和合同管理程序,如:评估供应商、签合同准备、供应商管理、由第三方保存附带条件委付的契约(escrow).
- 11.系统开发方法和工具,以及它们的优缺点。例如,敏捷开发实务,原型、快速应用开发(RAD),面向对象的设计技术。
- 12.质量保证方法
- 13.测试流程的管理,如,测试战略、测试计划、测试环境、启用和关闭(测试)的标准。
- 14.数据转换工具、技术和程序。
- 15.系统(和/或体系)(报废、退役)的处置程序。
- 16.软件、硬件的认证和鉴定实务。
- 17.(现场)实施后的检查的目标和方法。如:项目关闭、收益实现、绩效的测定。
- 18.系统移植和体系开发实务。

★ 可能的考试重点

物理体系分析项目阶段 需要了解每个阶段的具体要求和审计师的关注

检查现有体系
分析和设计
起草功能需求
供应商和产品选择
编写功能需求
概念验证

应用控制 (原第七章) (重点) 各种控制方法的含义

输入控制 (授权输入、批控制、错误报告与处理、联机系统或数据库系统的批完整性)
处理程序与处理控制 (数据确认和编辑检查、处理控制、数据文件控制程序)
输出控制
业务流程控制保证 (在业务流程中实施应用控制)

★ 知识点摘要

业务运营的实现

组合项目/大项目管理

一方面,尽量使各项目((projects)独立运作,减少相互影响;另一方面,也要协调各项目(之间

的资源、进度)。这就需要专门的人项目组织(机制)。**大项目所有者(program owner role)**，与**项目所有人(Project owner role)**是不同的。

队签发**书面的委托书**。因为项目实施过程中可能出现新的人项目，为设定大项目的关联关系和边界而签署委托书是极为重要的。它也是**正规的管理授权(程序)**。

谈到大项目管理、项目组合管理和**项目管理**，**组织要有专门的良好设计的组织结构**，例如：智囊团、项目管理办公室、项目组合管理组。**项目管理办公室**的任务和活动仅限于**项目管理程序/标准**，而不是项目和大项目的具体工作内容。所谓**项目组合(管理)(Portfolio Management)**，是指指定时刻组织内**正在执行的所有项目**(的状态)，即：状态快照(snapshot)。相对于大项目管理要求所有相关项目的紧密配合，项目组合并无这样的要求。项目组合管理要求特定的项目组合管理报告。通常，该报告是项目组合条框图(bar chart)、收益与风险矩阵、项目组合推进图等等。

业务模式的制订和批准

任何 IT 项目的**首要考虑**，就是**业务模式(business case 或译作:业务案例、商业模式、商业案例)**。**业务收益推动项目(的投资和开发)**，是日益得到广泛认可的原则。业务模式为组织决定是否推进项目提供了(参考)信息。根据组织和通常的投资规模，**业务(模式/案例)的制订是项目启动的第一步**。最初的业务模式源自项目启动/项目计划的一部分任务的**可行性研究**。**业务模式还应该是贯穿任何项目生命周期中决策过程的关键组成部分**。在任何阶段都需要进行**正式和全面的检查**，业务模式一旦被认为(或被证实)不再有效，就会增加成本或降低预期收益，项目发起者((project sponsor)。决策点(decision points)，有时被称为“阶段门(stage gates)”或“杀点(kills points)” **业务模式变更**应该由部门计划和审批程序**重新(获得)批准**。

效益实现技术

业务效益日益通过技术革新来实现。在应用新技术之后实现。当**用户熟悉了新的程序和流程**时就会产生项目效益。收益(或效益)也很少能如计划那样地实现，**组织不得不经常检查并调整(发展)战略**。**效益实现是一个持续过程**，必须如业务过程一样受到**管理**。**阶段性评价(Gateway assessment)**包括评价商业模式和收益实现过程，是收益实现过程的关键组成部分。所有有关**项目的治理决策**都应该根据**业务模式**产生，并要定期检查**效益实现**情况。

项目管理结构

▲ 全局方面

信息系统项目可以由组织中的任何一部分启动，包括 IS 部门。

项目一般是一次性作业。**可以分成几个明显的阶段**(参考本章描述的软件开发生命周期)，每个阶段的目标都和项目**总体目标一致**，项目管理应该是**基于项目组织的业务过程**。

项目管理的复杂性要求精心而清晰的**设计项目管理流程** 就如同**业务流程设计**

▲ 项目范畴和项目环境

由于可能有几个项目同时运作，应该探究这些项目，**找出其为组织(战略服务)的公共目标、识别和管理风险、确定资源的关系**。比较常见的探究方法，是建立项目组合管理。

项目启动之前，有一些关键的活动和目标**必须纳入项目计划**的考虑。有必要关注它与其所处的社会环境的关系。这种**关系的协调与规划是项目管理的一项工作内容**。目标是确定项目相关的环境，为整体项目计划和项目成功打好基础。

▲ 项目组织形式

影响型项目组织(Influence project organization) 在影响型项目组织中，项目经理只是项目团队中的一员，**没有正式的管理授权**，他仅仅起到告知团队其他成员要完成的工作的作

用。

纯项目组织(Pure project organization) 在纯项目组织中，项目经理有正式的授权负责该项目，项目团队成员通常有专门的工作场地与其正常的办公场地相分离。

矩阵项目组织(Matrix project organization) 在矩阵项目组织中，项目经理与部门领导都有管理职权。

主要信息系统项目的申请要提交给**信息系统指导委员会审批**。

项目经理不一定是信息系统部门的员工，应该被授予对项目全面运营控制的权力和适当资源应用的权力。**IS 审计师可以作为控制专家被包括在项目小组内**；信息系统审计师也可以独于项目组，对项目提供独立、客观的审核，职责是确保内部控制是有效的，责任各方的参与程度是适当的

▲ 项目沟通和组织文化

确保项目团队内沟通顺畅的理想方法是**开项目推进研讨会(Project start workshops)**，以协调、沟通所有团队成员，并获得**利益相关者**(通常指股东、出资方和项目发起人)的参与和理解。这样会有助于勾勒出项目蓝图，形成项目组织文化。

▲ 项目目标

项目需要明确地定义其结果是确定的(specific)、可度量的(measurable)、可完成的(achievable)、中肯的(relevant)和符合既定时限的(time bound)。即 SMART 管理。兼顾不同的项目目标需要统揽全局的眼光。这些目标分为**主目标(main objectives)**、**附加目标(additional objectives)**和**非任务性目标(nonobjectives)**。主目标总是与业务成功直接相连、附加目标不直接相连但对项目成功有贡献、非任务性目标与主目标、附加目标相反，它用于增加项目范围的清晰度。

定义项目目标的公认、通用方法是**目标分解结构(OBS, object breakdown structure)**或**工作分解结构(WBS)**再细分成**工作任务工作包(WPs, work packages)**，其结构也是分阶段面向(任务/作)流程的，每个工作包必须拥有唯一的所有人，常用形式**任务表(to-do lists)**，任务表可以帮助每个团队成员履行计划和相互协调一致。

▲ 团队和个人的角色与责任 P9

为了成功地完成和实施一个新系统，**信息系统审计师**应当积极参与到业务应用系统的开发中，**以促进充分的控制措施被设计和实施到新系统中**。

项目指导委员会(Project Steering Committee)—项目指导委员会对项目工作提供总体指导，并确保对项目有影响的各方都能参与到项目决策中来。项目指导委员会**对所有的成本和时间表负最终责任**，受项目影响较大的每一职能部门派出一个高级代表组成项目委员会，委员会中的每一个成员必须被授权，对影响其所代表部门的系统设计进行相关的决策。通常由项目发起人担任项目指导委员会主席。

项目发起人(Project Sponsor)—为项目提供基金，并与项目经理配合工作以定义项目成功的衡量标准。**建立量化的指标来衡量项目的成功是一种重要的项目管理方法**。数据和应用程序的所有权归属项目发起人，项目发起人通常是应用系统所要支持的主要业务部门的主管高级经理。

安全官(Security officer)基于公司安全政策对程序的数据分类，

质量保证(Quality Assurance)—在每一个开发阶段期间和阶段结束时，评价结果及其交付物，并确认与需求的符合性。**评价的切入点取决于所采用的系统开发的方法、系统结构和人小，以及潜在偏差的影响**。质量保证重点是要对面向过程的活动进行评价

理解系统开发、获取和维护所使用的方法，并确定潜在的脆弱性和需要控制的点，对于信息系统审计师来说很重要。如果缺乏必要的控制(可能是组织结构不合理造成的，或所使用软件的方法不当造成的)或处理过程无序，信息系统审计师有责任向项目小组和高级管理层及

时指出存在的问题。信息系统审计师介入开发和获取活动,建议有关人员实施适当的控制是非常必要的。

项目管理实务

项目管理知识和实务可以通过**项目启动、计划、执行、控制和结束**等各阶段的流程/程序构成来描述。成功的项目计划的总体特征是**基于风险**的管理流程/程序和它的深入应用。

项目有三个要素需要兼顾:

时间/时期—完成项目需要多长时间?

成本/资源—花费多少?

交付的成果(Deliverables)—完成了哪些工作?

一个重要内容是**项目的质量** 项目经理都必须给项目指导委员会、项目发起人和用户一个清晰的、书面的、正式的项目质量的预期。

▲项目启动

项目启动文档(PID, project initiation document)或项目请求书(PRD} project request document)的批准,是整个项目的开始。

▲项目计划

评估项目工作量的工具和技术:

软件规模评估 矩阵可以衡量开发工作的负担。其第一步是确定软件开发所需的资源。

源代码行数 衡量软件规模的传统方法,也是最简单的方式是计算源代码的行数(SLOC),也称为代码千行数(KLOC, kilo lines of code)。另外一个类似的概念是源指令数(KDSI, thousand delivered source instructions)。这种衡量方式已经被结构化编程语言(如: BASIC} }kCOBOL)是非常出色的。

功能点分析 20 世纪 70 年代后期,功能点分析((FPA} function point analysis)方法被提出,后来成为广泛应用的大型业务应用系统开发复杂性的估算工具。功能点(FP, Function Point)首先通过完成一个表来决定一个特殊的输入是否是简单、平均的或是复杂的。共定义了 5 个功能点: **用户输入数、用户输出数、文件数、用户请求数和外部接口数**。FPA 是一种更精确的、有用的五因素分析法,使用功能点,基于输入、输出、文件、接口和请求数量来间接地衡量软件规模。功能点分析法可以很好地估算**业务应用系统的开发**

FPA 特征点估算法 用于**网站应用**

▲★★项目执行

进度安排可以应用多种技术表示,如甘特图(Gantt charts)和项目评审技术((PERT Program Evaluation Review Technique,或译作:计划评审技术、流程评估检查技术)图。在项目的关键点。**关键路径方法**。路径是由项目开始到结束,贯穿上述网络结构的各条首尾衔接的所有任务组成的链。所有链之中,**时间消耗总数最大**的一条链,就是关键路径。关键路径上各任务是没有时间弹性的,反之,没有时间弹性的任务都在关键路径上。CPM 软件包提供了科学的资源平衡方法,并产生了满意的效果。使用 **PERT 技术也可以计算关键路径**,它就是网络中最一长的、唯一的那条路径。

甘特图可以结构化的帮助项目完成所需各任务的进度安排,显示出各任务的开始和结束时间,也显示了可以并行处理的任务以及各任务的完成顺序。甘特图也反映了分配给每个任务的资源。甘特图利用**基线**来帮助确认某项任务完成的早或晚会发生什么情况。

项目评审技术(PERT)是一种网络管理技术,常用于充分计划的系统开发项目。设计系统开发项目的 PERT 网络图时,**第一步是识别所有的任务、它们的相对次序**。

时间盒(Timebox)管理是一个新兴的项目管理技术,它是在一个相对短的、绝对的和不许变更的时间内以及有限资源的情况下,定义和部署软件交付品。在应用这项技术时,在软件质

量和在时间盒内完成所规定的工作之间需要达成平衡。项目管理层在定义需求范围方面有一定的弹性。**时间盒管理方法非常适合于原型法或者快速应用开发项目**，不适合传统的 SDLC 方法。

▲ 项目管理

可以通过自动化的软件来管理项目事项和估算成本，并对项目管理中的某些活动进行预测、报告。

▲ 项目的控制

项目的控制活动包括范围管理、资源使用和风险管理。

▲ 结束项目 P18

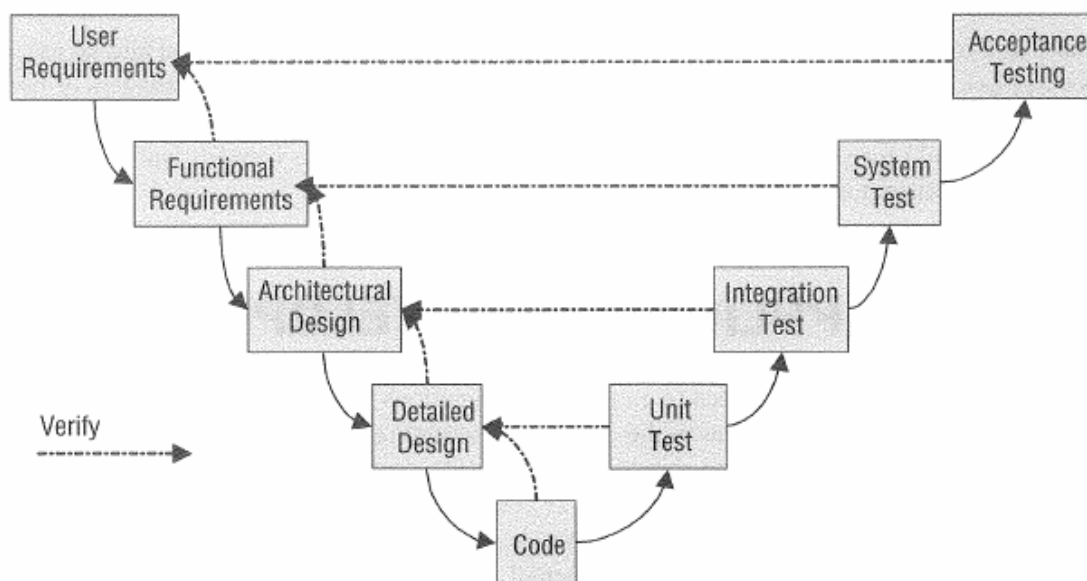
项目成果的**所有权**需要指定

3.5 业务应用系统的开发

为避免可能的风险，应当建立 IT 过程，对开发、实施、维护和废止各个阶段相关 IT 资源和其他类似活动进行管理和控制，这些过程应当被视为系统开发生命周期的组成部分。

组织开发业务应用系统的动力来源于**关键业务因素**紧密相连。所谓关键业务因素是指为达成**企业战略业务目标**而驱动业务职能行为和实施的业务职能属性。因此,所有**关键业务目标**(作为公司战略的分解)都**必须转换成关键业务因素**,以便于在 SDLC 项目过程中所有的相关团队都**参与到业务运营**中。使用这种方法的优点是所有的相关团队**对目标都有共同的、明确**的理解,也知晓如何提供业务支持。另外,相互冲突的关键业务因素(如,成本与功能)和相互依赖的关键业务因素可以在**SDLC 项目的早期被发现和解决**。

瀑布模型(和其变种模型)展示了在生命周期中的测试错误的方法,这种方法保证在每一个阶段对错误.加以验证与控制,而不是等到用户验收阶段再来测试错误。这种确认与验证错误的模型也称为 **V 模型**,它强调开发阶段与测试阶段的相互关系。一旦某一个程序单元的代码编写完成,就应当进行单元测试,这是一种最小粒度的测试。从这个模型中可以看出, **单元测试**是用于验证详细设计的正确性, **集成测试**是用于验证各单元集成后的正确性, **系统测试**用于验证系统架构的正确性, **用户接受测试**是用户为了验证软件是否实现了用户需求。



生命周期方法有助于信息系统审计师识别所选择评价的各个系统部分应当具有的属性并基于他们的技能和能力，可进一步深入技术层面去发现存在的问题。较深入地了解开发过程中的风险，独立地向管理层报告。

应用系统开发的分类：以**组织为中心的计算（加工数据 分享信息）**（MIS， ERP， CRM， SCM等）和以**最终用户为中心的计算（展示数据不同维度 绩效优化）**。

centric application takes an alternate development approach. The objective of an organization-centric application is to collect, collage, store, archive and share information to business users and various applicable support functions on a to-know basis. Thus, sales data are made available to accounts, administration, governmental levy payment department. Regulatory levy fulfillment is also addressed by the presence of organization-centric applications. The objective of a user-centric application is to provide different views of data for their performance optimization. This objective includes DSS, geographic information systems (GIS), techniques, etc. Most of these applications are developed using alternate development approaches.

3.5.1 传统的系统开发生命周期法(SDLC)

SDLC 是一种系统的、顺序式的软件开发方法，从**可行性研究**开始，通过**需求定义、设计**（设计也包括程序和数据库规格说明，以及一个安全计划 变动控制程序）、**开发、实施**（评价业务应用系统是否将风险减轻到一个适当的水平）和**实施后维护**等阶段而逐步发展。这一系列的步骤或阶段都有预定义的目标和活动，并建立了**相应的责任、预期的结果和完成目标**的日期。电子商务项目的关键成功因素的衡量包括生产率、质量、经济价值和客户服务等方面。在开发初期从用户那**获得一个完整的、清晰的需求是很困难的**。组织业务环境的改变能够引起用户需求的变化。

3.5.2 集成的资源管理系统

企业资源计划(ERP) 在实施前应当进行彻底的影响分析与风险评估。进行 ERP 的风险评估是组织的高级管理层的责任。

3.5.3 传统的 SDLC 各阶段描述

▲**可行性研究(Feasibility Study)** 与可行性研究紧密相关的工作是制订影响评估。

▲**需求定义** 需求定义阶段就是解决这类常被称为非功能性需求的问题。需求被描述为书面的需求规格说明书。需求定义阶段进行有效的需求检查。所有有责任的管理人员和用户部门必须积极地参与到需求定义阶段，以避免发生资源浪费。**实体关系图(ERD)**描述系统数据和它们之间的相互关系。实体的**主键(Primary Key)**可以唯一地标示实体的每个实例。**外键(Foreign Key)**。外键与主键不同，外键是关联属性的实体的主键的映射关系。面向对象方法中与 ERD 等价的**概念是类图**。**管理层的支持承诺、准确详细的需求定义和用户的积极参与**是软件开发成功的前提。

▲**软件获取** 请求建议书(RFP)或邀标书(ITT) 如果已经知道需要购买什么样的产品与相关服务，组织可以使用**邀标书**的方式，把所需的产品服务直接与价格结合在一起。例如，购买硬件、网络、数据库等产品时可直接采用邀标书;当需要供应商提供解决方案和相关的支持与维护服务时，组织需要对方更多地提供其能力、经验和方法方面的信息，这时适合采用请求**建议书**的方式。例如，购买 ERP 和 SCM 软件，这时请求建议书还应该涉及产品递交方式、源代码第三方保全协议等内容。组织在制定正式的请求建议书前，要先起草请求**提供信息(RFI,Request for Information)**文件，送给软件供应商，请其就组织的特定需求，提供产品与服务方面的初步咨询。。合同**法律顾问对合同条款进行详细审核**。 信息系统审计师要参与到软件获取过程中来，以保证组织与供应商达成任何协议前，**已充分考虑了安全控制的需要**。如果**软件中没有集成必要的安全控制**，那么就很难保证将来系统要处理的信息的完整性。商品化软件包相关的**风险有不完备的审计踪迹、口令控制和应用系统的整体安全**。

软件采购首先要满足两个条件：**符合业务目标 适当的机构进行授权**

▲**设计** 系统设计包括两个方面，首先是**总体结构**的设计，其次是把系统分解成一个个**具体**

模块与组件。在软件设计过程中, **用户过多参与**是不合适的,信息系统审计师主要关注系统设计对控制的影响。**基线(baseline):**基线标志软件开发过程的各个里程碑,任何一个软件配置项(如设计说明书、测试计划等),一旦形成文档并审核通过,即为一个基线。它标志开发过程中一个阶段的结束。对于已成为基线的软件配置项,虽然可以修改,但必须**按照一个特殊的、正式的变更管理过程进行评估**,在确认风险与成本的前提下,再进行修改。信息系统审计师还关注软件设计过程本身的有效性。是否将连续在线审计功能集成到系统中

▲**开发** **编程标准是一项基本控制**,因为它是系统开发过程中编程团队成员之间、编程团队与用户之间交流的方法。编程标准减小了系统开发过程中由于人员轮换带来的返工,提高了系统的开发效率,方便了维护和修改。**结构化的应用程序**更易于开发、理解和维护。**集成开发环境**,集成开发环境又称为在线编程设施。风险:非授权访问。**程序库集中安装在服务器上存在的风险:多个程序版本的存在引起混淆、非授权访问与更改程序代码的可能性增加**,危及了系统与程序的完整性、对程序代码的合法修改可能被其他人员的修改覆盖。**调试程序**是为了在生成正式软件产品前,检测出程序中的异常中止和程序编码错误。调试工具一般分为二种类型:**逻辑路径监测<Logic Path Monitors>**—通过对程序运行过程中的一系列事件的报告,向编程人员提供逻辑错误的线索;**内存转储(Memory Dumps)**—通过对计算机内存中某一时间点的内容映像进行输出。一般在**程序异常中断点**提供内存转储,以供编程人员发现数据或参数的不一致。内存转储的另一种形式称为追踪(Trace); **输出分析(Output Analyzers)**—帮助检查程序运行的输出结果的精确性,一般通过比较预期的结果与实际结果而实现。

▲**测试** 一般来说,许多大型应用系统的测试是应用自底向上的方法,并涉及了不同层次的各种测试,例如单元测试、子系统测试/集成测试、系统测试等。**系统测试**的测试用例应根据需求分析规格说明来设计,并在**实际使用环境下来运行**。验收测试:质量保证测试(QAT 技术)和用户验收测试(UAT 功能)。最后一步通常是认证和鉴定程序。

测试方法:**导航测试(Pilot testing)**对系统的某些特定的或预定义的内容进行预备性的测试;**白盒测试(White Box Testing)**允许测试人员利用程序内部的逻辑结构及有关信息,设计或选择测试用;**黑盒测试(Black Box Testing)**黑盒测试意味着测试要在软件的接口处进行,测试人员完全不考虑程序内部的逻辑结构和内部特性, **是一种动态分析工具**。**回归测试(Regression Testing)**对测试计划或测试场景的某些内容重新进行一测试,目的是为了保证最近对软件所做的变更和修改没有引入新的错误。回归测试中所用数据要与以前的测试数据一致。**功能/确认测试(Function/Validation Testing)**功能测试又称确认测试,它的任务是验证软件的有效性;**社会性测试(Sociability Testing)**这种测试是为了证实新系统在目标环境中运行时,不会给其他系统带来负面影响。

▲**实施** 实施计划应该在实施前制订, **正式的实施计划早在设计阶段就建立了**,并随着开发过程不断完善。创建生产环境的每一步都必须保证:有专人负责、步骤如何验证、出现问题时的返工流程。实施新项目的公司常常在新的系统框架下同时运营和支持现有的系统和新的系统。现有的支持流程和组织单元需要为新的平台和旧的平台提供适当的支持。因此,对**现有系统和新系统实施创建、集成、移植和切换等各阶段的管理是主要的挑战**。

实施技术的阶段:**阶段 1:制订支持结构**:差异分析、角色定义(运营经理);**阶段 2:建立支持功能**:服务等级协议(SLA)、实施计划/知识传授计划、培训计划、最终用户培训

数据转换 数据转换和数据转储是类似的活动 另一个重要目标是减少风险。因此,了解业务单元日常工作的重要性并理解按次序执行。

回滚方案 最佳实务提倡的分阶段的开发应用系统的方法能够减少关键任务的应用系统的风险,因此需要逆向数据转换的组件。

数据在转换前应该备份。**未授权的拷贝或副本**过多会导致系统中数据的误用、滥用或失窃。

认证/鉴定 认证是一个程序,它通过认证机构的评审员或审核师检查组织符合认证某些要

求的程度。

▲**实施后检查** 在**项目开始阶段**就应该考虑到实施后检查和需要检查的信息；目标是评估和度量项目**对业务的价值**(商业价值实现)；实施检查的 IS 审计师应该独立系统开发过程。而**提供咨询服务的 IS 审计师不能参与检查**。系统应该定期检查以确保系统始终符合业务目标，符合**成本—效益原则**，一直保持着**控制的完整性**。

3.5.4 与软件开发相关的风险

风险之一就是系统**不符合用户的业务需要**，不能达到用户的期望值。另外一个风险就是**项目风险**，设计和开发系统的项目活动如果超出了必要的财务资源限制。

有效的软件**过程管理**与**项目管理**，SDLC 的活动才能被控制和改善。仅仅遵循一个 SDLC 管理方法并不能确保一个开发项目的成功完成。

3.5.5 结构化的分析、设计和开发技术的使用

一种从**抽象到具体**的实现框架，通过在不同的抽象水平使用不同的图形符号来表示应用程序的数据和程序组件，直到其达到一种能够使程序员可以编码的抽象水平。

是否被良好定义、记录和遵循

3.6 其他的系统开发方法

增量开发或渐进开发(Incremental or Progressive Development)—系统内嵌到各阶段或版本中，而不是在开发完成后整体交付。这种开发也存在不同的版本，其操作起来就像不同的子项目。通常的做法是在第一个版本中交付基本的系统结构，随后的版本在系统的功能、用户范围和使用定位等方面不断进行扩充。

迭代开发(Iterative Development)—这种方法涉及使用迭代或增量来构建系统。也就是当出现一个需求增量之后，产生反馈以促进项目计划和软件开发产品的**重新调整**。迭代开发现今被认为是一种最佳实践，迭代开发是处理软件开发项目中的复杂性和相关风险的最恰当的方法。变种：**进化式开发(Evolutionary Development)**—原型—设计是用于构建一个**可工作的模型**，用于确认需求和寻找设计中存在的问题。**螺旋式开发(Spiral Development)**—对于复杂的大型软件，开发一个原型往往达不到要求，因此在详细的设计、构建和测试点，使用**一系列的原型**来开发一个解决方案。**敏捷开发(Agile Development)**—项目被分解成相对较短的时间段(time-boxed)进行迭代开发。从最早期的迭代开始，开发重点就是快速生产出可工作的功能性产品，然后对这些模块进行迭代，功能性从用户接口开始扩展，通过中间媒介层，再到存储的数据，然后**返回下一个迭代**。

3.7 其他形式的软件项目组织方式

3.7.1 敏捷开发(Agile Development) P50

敏捷开发(Agile Development)是指支持非传统方式开发复杂系统的一系列开发程序。将计划和指导任务的完成从项目经理转移到项目团队中的成员，项目经理可以由时间解决开发过程中的障碍问题，从而达成开发目标。敏捷开发方法对需求的适应方法**不强调管理需求基线**；敏捷开发方法的**重点是通过建立实际的功能与正式的定义早期软件**，快速地验证系统结构和数据结构；敏捷开发方法采取限制过失测试，但**通过建立频繁的测试循环来验证功能**，并纠正下一子项目的问题，避免再发生更多的时间和成本。敏捷开发方法**不强调已定义好的、可重复的过程**，而是**通过频繁的检查来实施和适应程序的开发**。

3.7.2 原型法 prototyping

原型法(Prototyping)也称作启发式或者演化式开发方法。用户首先提出开发要求，开发人员

识别和归纳用户要求，根据识别、归纳的结构，构造出一个原型，然后和用户一起评价原型，如果不行，则重新构造原型，如果不满意，则修改原型，直到用户满意为止。**完成的系统在控制机制方面一般较弱。变更控制也变得更加复杂（需求变动比较多）。**会显著节约时间和成本。

3.7.3 快速程序开发 Rapid applications development

RAD 是一种方法论，**快速开发具有战略性的信息系统，减少开发成本和提高维护质量。**通过一系列已经证明的应用开发技术，采用清楚定义的方法论。**RAD 支持单独系统的开发和实施**，但不支持整个企业信息需求或者某个主要业务领域的信息需求的规划与分析。通过对系统开发设立严格的时间框架，采用重用组件，RAD 提供一种快速开发系统的思路。

3.8 其他开发技术

▲**面向数据的开发方法**是通过数据和数据结构来表达软件需求的一种开发方法

▲**面向对象开发方法(OOSD)** 对象之间的相互依赖性很小，因此可以独立地被其他各系统选用 当一个类足够在以后的应用程序中重用，就将其加入标准类库。

▲**基于组件的开发方法** 基于组件的开发方法被认为是面向对象法的拓展，基于组件开发意味着通过定义的接口集成可执行的软件包，从而完成一定的服务。COM 是 ActiveX 技术的基础，ActiveX 控件是应用最广泛的组件。**COM/DOOM、CORBA**，它们允许对象在分布式平台下相互作用。这些技术也被称为**中间件**。通过程序、对象、组件之间的相互作用提供运行服务的软件。规避和控制这部分风险的办法是，在项目进行的过程中，**不断地对该阶段进行风险评估**，并设定有效的里程碑。

▲**基于 Web 应用开发方法** **Web Services** 是目前最流行的、基于 Web 应用开发的一个全新技术它基于互联网标准和 XML 技术建立，用以描述、发布到网络供其他应用程序调用。Web 服务的远景是由各种**以 XML 为基础的科技**所共同组成的，包括:SOAP,WSDL 以及 UDDI。**XML** (eXtensible Markup Language, **可延伸标记语言**)是一项用来将结构化资料以文字格式进行定义与组织的标准规范。((**XML 纲要》(XML Schemas)**...Web 服务的核心技术是《**简易对象存取协议**》(Simple Object Access Protocol **SOAP**) **发掘与整合(Universal Description, Discovery, and Integration, UDDI)**、**Web 服务描述语言(Web Services Description Language. WSDL)**提供一种 Web 服务功能的抽象化描述方式。WSDL 的职责在于告诉其它机器如何将请求与响应讯息进行格式化。

SOAP WSDL 以及 UDDI 彼此合作，共同组成一套三面向的服务架构。服务提供者将它们的服务内容透过 UDDI 目录公告周知，以便让其它有需要的机构知道这些服务的存在。需要服务的机构透过 UDDI 目录找到所需的服务之后，便可以藉由 SOAP 交易来呼叫这项服务，而该项服务的 WSDL 接口则清楚地描述了它所支持的请求与响应方式。

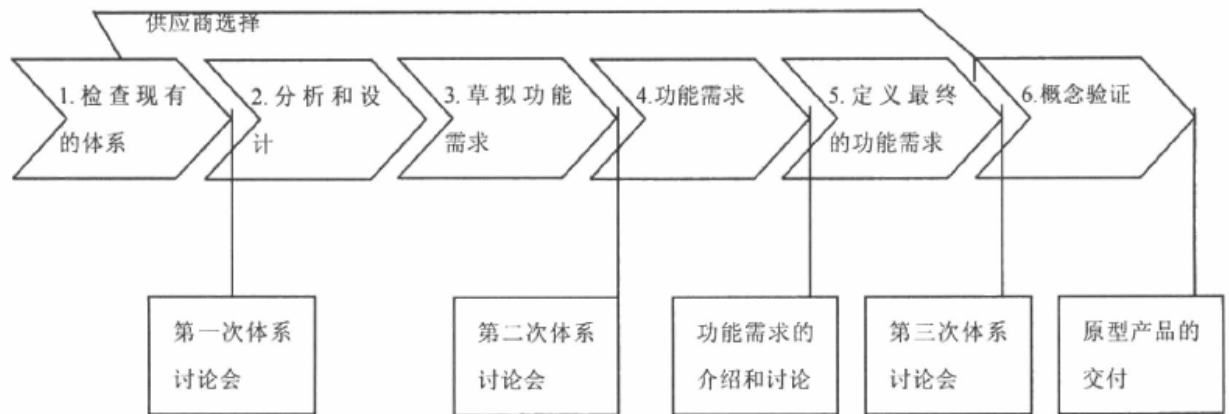
▲软件的再次开发 软件重组

▲**逆向工程法** 就是分析已有的程序，寻求比源代码更高级的抽象表现形式。

3.9 体系开发和获取实务

需求需要使用概念验证(a proof of concept)的方式验证。概念验证是物理体系的试验台实施。来证明选定的硬件和软件能够满足所有预期要求。

3.9.1 物理体系 physical architecture 分析项目阶段



3.9.2 实施体系的计划 P55 eP66

3.9.5 硬件的获取

吞吐量—单位时间内系统的有效工作量。吞吐量的衡量指标可以是每秒执行的指令数或其他性能单位。对于数据传输，吞吐量为有效的传输速率，通常用 Kbps, Mbps 或 Gbps 表示。

3.9.7 系统软件变更控制程序

变更控制程序用来**保证变更已经得到授权**、IS 管理层和相关人员清楚并参与到系统软件的变更过程中、而且变更不会破坏现有处理流程。变更控制程序应保证变更对生产系统的影响已经得到适当的评估(尤其是在安装失败时)失败，能使其影响最小化。软件打补丁前需要先进行风险评估。

3.10 信息系统维护实务

信息系统维护实务主要是指**管理应用系统变更**的过程，其目的是保证软件产品源代码与可执行代码的完整性。

▲变更管理程序 系统开发和维护人员在某些情况下需要提出变更，需要从适当的管理层获得授权。程序员不能写、修改和删除产品数据的访问。在进行任何变更之前，**程序员必须接到授权**。除了管理层批准变更之外，管理层还要安装**自动变更控制软件**，防止**未经授权的**的程序变更。变更申请应该有报批流程并文档化;**变更潜在影响应予以评估**。。在审计期间，可**抽查程序变化的一个样本追踪到维护表格**，决定变化是否进行了授权。**紧急变更**的用户需要监控

▲配置管理 更优地维护变更控制 管理层对软件配置管理的支持是非常重要的。配置管理工具能够支持变更管理和版本管理，并且自动化地执行。

3.11 系统开发工具和生产辅助工具

代码产生器 是基于系统分析员和数据流程图定义的流程产生程序代码的一种工具，通常结合 CASE 产品使用。

计算机辅助软件工程(Computer-Aided Software Engineering, CASE)技术，是一种先进的软件开发技术，可望解决长期以来困扰系统开发人员的软件生产效率低的问题。**高级 CASE (UDper Case)**是用于描述和记录业务和应用需求的工具。**中级 case(Middle Case)**—是用于制订详细设计的工具。**低级 CASE (Lower Case)**是生成程序代码和实施数据库定义的工具。

Is 审计师也可以成为 CASE 工具的用户，因为 **CASE 的一些功能能够协助审计程序**。数据流程图可以由高级和中级 CASE 工具生成，可以取代其他流程图技术。IS 审计师可以利用 CASE 工具生成的审计文档，有经验的审计师可以利用 cASE 工具来创建审计文档。CASE 工具也可以用于开发审核软件和嵌入式审计模块。组织 CASE 工具和其功能列表可以用于

了解系统并检查开发过程中的控制。

第四代编程语言(C 4GLs) 事件驱动的, 并广泛使用面向对象的编程概念: 使用应用生成器的不是最终用户, 而是数据流开发人员。

3.12 流程改进实务

▲**业务流程重组和流程变更项目** 业务流程重组(BPR) (PER, Package-enabled Reengineering)

新的业务流程不可避免地**涉及到业务方式的改变**, 并可能影响到组织的财务、经营理念、员工、业务伙伴和客户。

在 BPR 流程中, 需要关注的问题是:**一些关键控制可能会在业务重组过程中没有被纳入新流程中去**。信息系统审计师的任务就是要确定组织中现存的关键控制, 并评价移除这些控制对组织的负面影响。如果控制是关键的**预防控制**, 信息系统审计师必须确认管理层已意识到控制被移除, 并愿意接受没有这些控制所带来的潜在的重要风险。

对流程进行**标杆管理(Benchmarking Process)** 每一个流程与组织目标、组织战略改进联系在一起。**组织的变更(重组)工作与组织的整个文化和战略计划相一致**。**标杆管理(Benchmarking Process) 步骤:**

★★**计划**。在计划阶段, 基准团队首先要在标杆管理实践中确定关键流程; **研究**。团队应该收集流程的基准数据, 然后再收集其他相关流程的基准数据。**观察**。收集资料并访问标杆对象; **分析**。总结和解释所收集的资料, 并分析组织流程与标杆对象的流程之间的差异。**调整**。调整标杆比对的结果; **改进**。在标杆管理过程中, 持续改进是一个关键因素。

▲**ISO 9126** (软件产品质量评价特性及应用指南)给出了简洁有效的软件质量模型。

▲**软件能力成熟度模型(CMM Capability Maturity Model for Software) P71 P86 CMM** 的核心是把软件开发视为一个过程, 并根据这一原则**对软件开发和维护进行过程监控和研究**, 以使其更加科学化、标准化, 使企业能够更好地**实现商业目标**。它侧重于对软件过程开发的管理及软件工程能力的改进与评估。

第 1 级 初始级

第 2 级 可重复级 建立基本的项目管理控制

第 3 级 已定义级 既关注项目问题, 也关注组织问题

第 4 级 已管理级 对软件开发过程和软件产品都有一个**定量**的理解

第 5 级 优化级 不论组织还是项目必须**追求持续的、可度量的**过程改进

▲**CMMI** 全称是 Capability Maturity Model Integration 强调了对需求的管理 加强了对工程过程的重视 更加强了对风险的管理

▲**ISO 15504 SPICE** 类似于 CMMI, 是过程改进、基准测定和平衡采用最佳实务的标准 eP85

3.13 应用控制 ★★ P74

应用控制是针对输入、处理和输出功能的控制。**应用控制目的:**

- .在计算机系统中仅有完整的、准确的和有效的数据被输入和更新
- .处理过程完成了正确的任务
- .处理结果与预期目标相符合
- .数据得到了维护

3.13.1 输入/源头控制

只有**合法的、经授权**的信息才能被输入, 而且只对这些**事务处理**一次

▲**输入授权** 有充分的控制**使授权数据不被改变**

在线访问控制(Online Access Controls)保证只有经授权的人可以访问数据或执行敏感的功能。

终端或客户工作站的识别(Terminal or Client Workstation Identification)

源文件(Source Documents) 用于记录数据的表单

▲批控制和批平衡

批控制对输入事务进行分组以提供控制总计

总金额(Total Monetary Amount)—确认被系统处理的项目总金额等于处理前批文件中项目金额之和

总项目数(Total Items)—确认在本批次中的每一个文件中的项目总数与被处理的项目总数一致。

总文件数(Total Documents)确认一个批次中文件总数等于被处理的文件总数。

哈希汇总(Hash Totals, 也译作:杂选汇总、杂选总计)—确认一个批次中的所有文件中的数值类字段的总和(虽然不同类型字段加起来的值并没有实际意义)与系统计算出来的总和相等。例如, 在输入一张发票前, 把发票中的单价、数量、重量、总价等数值型字段的值加起来, 与输入系统后计算机算出来的值进行比较, 看有没有输入错误出现。

批平衡(Batch Balancing)能够通过人工或自动的对账(Reconciliation)来执行。

批登记(Batch Registers)

控制账户(Control Accounts)—使用控制账户是通过一个初始编辑文件来确定批总计。数据被处理并先存入主文件, 然后把主文件中的批总计与初始编辑文件中批总计进行比较。

计算机一致(Computer Agreement)

▲错误报告和错误处理方法

输入控制技术包括:

.事务日志(Transaction Log) 事务日志包含数据更新的详细清单

数据对账(Reconciliation of Data)

源文件取消(Cancellation of Source Documents) 取消输入源文件的程序 避免重复输入

▲联机系统或数据库系统中的批输入完整性 监督人应当先检查联机批输入, 然后才提交给系统进行处理。

3.13.2 处理程序和控制 P77

处理程序和控制用于保证应用程序处理的可靠性

▲数据确认和编辑检查程序

如果可以绕开数据的确认和编辑控制, 系统日志应当自动记录下这种行为; 数据确认可以鉴别错误的数据、不完整的数据或遗失的数据, 及相关的数据库项目之间的不一致性。如果使用智能终端, 就能够执行前端(front-end)数据的编辑与和确认。编辑控制是在应用程序中使用的预防性控制, 发生在数据处理之前。

顺序检查(Sequence Check)。通过自然顺序号进行控制记数

极限检查(Limit Check)。数据不应当超过一个预定义的数量。

范围检查(Range Check)。数据应当在一个预定义的值的范围内。

有效性检查(Validity Check)。按照预定义的标准进行的数据确认的一种程序化检查

合理性检查(Reasonableness Check)。例如, 在大多数情况下, 一个小输入的数据与预定义的合理极限或发生率相匹配。配件制造商通常所接受的订单不超过 20 个

击键校验(Key Verification)。通过计算机来记录并比较录入人员击键顺序与次数的差异来发现录入过程中的错误。

校验数位(Check Digit)。一个经计算而附加到数据上的数值, 确保原始数据不被改变或者是被一个实际上不正确但可以被计算机接受的值所替换。

逻辑关系检查(Logical Relationship Check)

存在性检查(Existence Check)。数据被正确的输入并与有效的预定义的标准一致。例如，一个有效的事务代码必须在事务代码区域输入。

完整性检查(Completeness Check)。一个字段应当总是包含非零和非空数据，要对该字段的每一个字节都要进行检查，以检测数据是否满足非零和非空的条件。

▲处理控制程序

处理控制保证计算数据的**完整性和准确性**。这类控制保证数据在文件或数据库中的完整和准确，只有**授权的处理或修改程序**才能对数据进行更改。

编辑(Editing)。编辑检查可以是一个程序指令或一个子程序，它用来测试数据的准确性、完整性和有效性。也可以用于**控制输入或其后的数据处理**。

运行到运行的总计(Run-to-run Totals)。提供了一种在**数据处理各阶段中验证数据值**的能力，它保证被计算机读取的数据被处理系统正常接受了，并应用于更新处理过程。

▲**数据文件控制程序** 文件控制应当确保对于存储的数据只有授权的程序才能进行处理。

奇偶校验检查(Parity Check) 错误检测方法采用一个校验位和冗余传输就足够了。**冗余校验**是一个通用错误检测程序，系统检查传输数据块(包含一个或多个记录或信息)中的特征数字或位模式，如果这个数字或位模式与预定义的参数不一致，接受设备忽略这个传输数据并通知用户重新传输。

数据文件或数据库中的表，通常分为以下四类:系统控制参数(System Control Parameters) 常备数据(Standing Data) 主数据或平衡数据(Master Data/Balance Data) 事务文件(Transaction Files)

3.13.3 输出控制

平衡和核对(Balancing and Reconciling)。数据处理应用程序的输出应当和控制总计平衡。

报告接收确认(Verification of Receipt of Reports)。

3.13.4 对业务流程控制的鉴证

需要在**业务流程和业务活动**这二个层次上对控制进行评价 对业务流程所有者特有的控制进行评价，这些控制包括:**建立安全措施及职责分工机制，对访问权限进行阶段性的审核与批准，在业务流程中建立应用控制**等。在流程中评估业务风险;对最佳实践进行标杆管理

3.14 应用控制的审计

审核应用系统文件，以加深对应用程序的功能的理解。

风险评估可以提供**应用程序的固有风险**的信息。

▲**观察和测试用户操作程序** 访问控制列表提供了个人访问级别的信息。访问应当基于岗位职责说明，**用户活动报告(Activity Report)**提供用户的详细的活动量和时间，**违例报告(Violation Report)**描述了任何不成功的访问，或未经授权的访问企图。

▲**数据完整性测试** 数据完整性测试是一种实质性测试程序，它检查保留在系统中的当前数据的准确性、完整性、一致性和授权。当检查是针对授权的源文件时，通常每次仅检查文件的一部分。由于这个文件是**定期的循环检查**，因此，这种控制技术通常也称为**循环校验(Cyclical Checking)**。

关系完整性测试(Relational Integrity Tests) 数据元素和记录)和**参照完整性测试(Referential Integrity Tests)** 实体之间)

▲**联机事务处理系统中的数据完整性** 具备容错能力(Fault Tolerance)

原子性(Atomicity) 一个事务在整体上要么是完整的 要不提交要不回滚

一致性(Consistency) 一致性确保事务处理之前和之后数据库始终处于合理的逻辑状态。

隔离性(Isolation)。每一个事务与其他事务相互隔离

持久性(Durability)。如果给用户的事务报告是完整的, 则对数据的任何变化都能恢复

▲测试应用系统

快照 记录通过程序逻辑路径的指定事务流

在生产系统运行过程中, 把特定的软件嵌入到主机应用程序中, 过滤和筛选审计所需要的输入事务和生成事务。通常是作为软件开发的一个部分, 有两种类型:一是**系统控制审计检查文件(SCARF)**。由审计师决定把合理的测试嵌入正常的处理流程中, 为将来的审核提供信息。二是**样本审计检查文件(SCRF)**。随机的选择事务以提供代表性的文件进行分析。

为了便于**对应用程序系统测试的评价**, 信息系统审计师可以使用**通用审计软件**(Generalized Audit Software, 这在发现了特定的应用程序控制弱点时特别有用

▲连续在线审计

连续在线审计是一种重要的信息系统审计工具, 特别是当其用于多用户分时系统环境, **处理大量的事务但几乎没有留下纸质轨迹的时候**。通过允许信息系统审计师在连续的基础上又不中断组织的正常业务运营来评价运行控制, 提高系统的安全性。需要保证测试生产数据隔离。**快照(Snapshots)**。这种方式记录一个事务从输入到输出各阶段的**处理轨迹**。在使用该技术时, 通过对输入数据使用标识符来对事务作标签, 并跟踪该记录的处理轨迹, 供信息系统审计师后续检查。

审计钩(Hooks)。该技术在应用系统中内嵌程序“钩”, 象标识符那样起作用, 在错误或不规范事务失去控制之前, 提醒信息系统审计师采取行动。

持续和间歇性模拟(CIS)。在一个事务的处理运行期间, 计算机系统模拟应用程序指令的执行。在每一个事务输入时, 模拟器就确定该事务是否符合预定义的标准

集成测试 ITF 在审计对象应用系统的生产文件中设置虚构的事务, 信息系统审计师可以使这些为特定测试目的而**虚构的事务与真实事务**一起进入应用系统中运行, 并让这些事务更新虚构实体的记录。然后, 审计师用经独立计算的数据来比较虚构事务的处理输出, 以确认计算机处理数据的正确性。

3.15 对系统开发过程的审计

信息系统审计师应判定系统的主要风险及其等级, 以便选择适当的控制方法;

▲项目管理审计 信息系统审计师要评估 SDLC 每个阶段的相关风险, **确认已建立适当的控制机制**, 并且避免**成本**太人的控制。

▲详细设计和编码阶段 信息系统审计师可以抽查一部分程序, 看是否遵循了标准, 程序是否符合系统设计。此外, **需要审查可能的控制漏洞, 每个设计的控制是否进行**。如果控制需要确定, 信息系统审计师要提出建议, 确保控制的有效性。 **这个阶段的检查可以判断用户的需求规格说明是否实现**。

3.16 业务应用系统

▲**电子商务★★** 应用服务器支持一个特定的组件模型并提供如数据管理、安全和事务管理之类的服务。最重要的风险要素是: 机密性(Confidentiality)、身份鉴别和抗抵赖(Authentication and Nonrepudiation)保证交易双方对已经完成的交易不能否认、

能对电子商务的**参与者身份进行唯一和正确的鉴别**的过程。(例如, 使用公钥、私钥对的加密和验证过程。)

数字签名。通过数字签名, 电子商务交易的发起者能被唯一地确定, 并由独立的第三方提供公证。数字签名的属性包括:

- 》数字签名对于使用它的人来说是唯一的
- 》能够被验证
- 》产生和附加这个签名的机制处于使用人的唯一控制之下

》数字签名与数据紧密连接在一起，一旦数据被改变，数字签名就可被验证为无效。

认证授权中心(CA)。也就是通常所说的数字证书认证中心

电子商务环境中对安全进行审计和评价的正式程序，以保证电子商务环境 and 应用系统中控制措施的有效性。

建立适当的保护，防止**所收集的个人信息**在没有得到用户同意的情况下，被暴露或用于其他目的。

▲电子数据交换系统(EDI)

EDI 接口可以**产生和发送回执(Acknowledgement)**，识别和验证商业伙伴的**身份**，通过检查商业伙伴的主文件的传输信息以验证交易的**正确性**。回执功能是标准的 EDI 事务，它告诉商业伙伴其电子文件已经收到。不同类型的回执功能提供了各个层次的细节情况，因此可以作为 EDI 交易的一个审计踪迹。

EDI 独有的风险中，最重要就是来自**交易授权的风险**。由于贸易双方的互动是电子化的，传统的“面对面”的身份验证方式不再起作用了。

安全控制措施：

在商业伙伴之间应当存在直接的或专用的传输通道，以降低传输线路被搭线窃听的风险。

应当对数据使用商业伙伴协商一致的算法来加密。

在传输中使用电子签名以识别传输起点和传输目的地。

应当使用信息鉴别，确保发送内容与接收内容一致。

应当建立控制确保所有的 **EDI 进站交易 inbound** 都被完全、正确地接收(通讯阶段)、转换(转换阶段)和准确地传送到应用程序(应用程序接口阶段)中，所有的进站 EDI 交易都仅仅被处理一次。

应当建立控制以确保只有恰当授权的**出站交易**才能被处理，包括以下控制目标:出站交易是**基于授权而被启动**;出站交易包含了唯一的预先授权的交易类型;出站交易只能被发送到合法的商业伙伴那里。由发送者将**分段计算总计(Segment Count Totals)**内建到交易中以设置追踪器。

专家系统。 基于判断规则，系统能够自动决定这种交易的重要性，

▲电子邮件 ★★ 重点加解密题目的理解

如果邮件是被发送到网络外部的某人，可能要穿越组织设立的防火墙计算机系统**防火墙**可以把内部网络与 Internet 隔离开来，以**防止非授权人员入侵组织的内部防火墙**可以追踪进出内部网络的数据与信息，以及来自于 Internet 的数据和信息，防火墙可以通过设置规则，来限制某些信息包来穿越防火墙。**网关**使用 TCP 协议把 IP 信息包重新生成一个完整的信息 网关**执行电子邮件格式转换** 在邮件服务器和电子邮件客户端之间敏感的、未加密的信息在传输时，可能会被**中途拦截**。**数字签名**用于鉴别来自于不信任网络的用户的**身份**。数字签名是**公钥加密技术**的另一种应用。接收者一端，用户需要**发送者的公钥**来解密信息和数字签名，在使用数字签名保护电子邮件信息安全的时候，有两种不同的加密技术可以用来加密信息。一种是对称密钥管理系统(**私有密钥**)，使用 DES 算法;另一种是用**公钥管理系统(非对称)**，使用 RSA 算法。若以公钥作为加密密钥，以用户专用密钥(私钥)作为解密密钥，则可实现多个用户加密的信息只能由一个用户解读;反之，以用户私钥作为加密密钥而以公钥作为解密密钥，则可实现由一个用户加密的信息而多个用户解读。前者可用于数字加密，后者可用于数字签名。 组织应当通过使用**防火墙、路由器和入侵检测系统**等网络安全技术来保护邮件服务器。

▲POS (Point-Of-Sale System)系统

▲电子银行 (electronic banking)

电子银行活动不会提高在传统银行业中并不存在的风险，但是会增加和更改那些传统的风险。电子银行的主要问题是**战略、经营和声誉上的风险**。

风险管理是董事会和高级管理层的责任 实施技术是信息技术高级管理层的责任 测量和监控风险是经营管理层的责任

Internet 有效地放入了**访问控制、用户身份鉴别、数据保护、审计踪迹和用户隐私保护**等安全控制措施的重要性。

董事会和管理层的监督

》对电子银行活动的有效管理监督

》建立综合的安全控制程序

》对外包关系和其他的第三方的独立性履行全面的尽职调查和管理监督程序

电子银行交易的**抗抵赖(Nonrepudiation)**和**可追踪性(Accountability)**控制、在电子银行系统、数据库和应用程序之中的**适当授权控制**、电子银行交易、记录和信息的**数据完整性**

▲电子金融

▲支付系统(Payment Systems)

电子现金系统的目标是模拟真实的现金，发行者通过创建数字证书来实现这个过程。

电子转账系统是三种支付模式中最简单的。付款人简单地创建一个转账支付指令，通过**数字签名**将其发送给发行者，发行者随后校验在这个支付请求上的签名并执行转账。

▲集成制造系统(IMS, Integrated Manufacturing System)

MRP (制造资源计划) 是目前流行的企业资源计划 ERP 中的典型模块，ERP 中一般还集成了客户关系管理模块(CRM)和供应链管理模块(SCM)。

▲电子资金转账(EFT, Electronic Funds Transfer) 如网银支付

EFT (EFT, Electronic Funds Transfer)是经由远程通讯而进行的货币交换，但实际上并没有进行真正的现金换手操作。

明确规定在电子数据传输和接收过程中，双方要遵循的强制性的法律要求。

EFT 是在买主和卖主及其各自的金融机构之间进行的电子的资金转移。EFT 是将一笔货币以电子方式从一个账户的金融交易。EFT 允许当事人从一个账户向另外一个账户转移资金，填写和现金收款程序。

▲综合客户文件(Integrated Customer File)

▲ATM 对于客户和设备本身而言，必须提供高级别的**逻辑和物理安全措施**加以保护。

ATM 体系机构有一个物理网络层、一个交换层和一个连接各种各样的 ATM 终端的通讯层。

▲图像处理系统 存储、恢复和处理图形资料，像图像、图片、图表，以取代文本数据，或作为文本数据的附加。图像处理系统对存储能力的要求很高，绝大部分图像系统都使用光盘存储。**新的控制措施到自动化处理过程中**，以确保图像文件不能够被改变、删除或丢失。

▲人工智能和专家系统 P112 eP122

知识库对专家系统来说是最关键的，它包含了特定的信息或与特定的主题事件相关的事实模型以及解释这些事实的规则。

决策树(Decision Trees)。使用调查问卷引导用户通过对一系列问题进行选择，直到得出结论。

规则(Rules)。通过 if-then 的使用来表达陈述性知识。

语义网络(Semantic Notes)。由包含代表实际的或概念上的目标节点和描述节点之间关系的曲线组成。**语义网络类似于一个数据流程图**，通过使用其内部机制来预防数据的重复。

审计师应当确保在开发基本的假设和模型时，使用了适当水平的专家经验。

专家系统好处：

在职员离开组织前获得他们的经验和知识。**最重要的好处就是可以收集组织中所有人的知**

识。

在特别的专业技术领域共享知识和经验。

促进一致的和有效的质量决策

提高个人的生产力和业绩

使高度重复性的一工作自动执行(例如，帮助台、信用评分等)

▲商业智能(BI, Business Intelligence)

风险管理—通过确定不正常的交易和积累事件与损失的统计资料来实施

泳道图 Swim — lane Diagrams 描述的是按组织职能部门分布的活动。

▲决策支持系统(DSS)

决策支持系统是一种交互式系统，它给用户提供了一种从大范围资料中访问决策模型和数据的简易方式，以这种半结构化的决策支持模式为组织实现业务目标服务。它是一种信息应用程序，用来帮助组织在基于智能工具提供数据的基础上作出决策。实施的风险：不能确定用户模型。

决策支持系统可以以图示方式提供信息，也可以包括专家系统或人工智能。

▲客户关系管理(CRM)

▲供应链管理(SCM)supply chain management

供应链管理是关于相关企业之间业务流程的连接，买卖双方处在供应链的两端，SCM 可以连接所有的领域，如供应商、消费者、仓库、批发商/零售商发行人之间的物流管理、信息交换、服务和货物交换;制造商的货物制造等。

从题目中看需要进一步通过外部知识强化的知识点：

▲数据仓库

保证数据安全不被修改

原始数据的正确性

元数据 Metadata 的质量