

第5章

管理信息系统的开发

本章包括管理信息系统开发过程、管理信息系统的开发方法、管理信息系统的开发方式、管理信息系统开发工具、管理信息系统开发管理这些内容。

5.1 管理信息系统开发过程

本节介绍了系统开发的总体规划和阶段划分。

5.1.1 系统开发的总体规划

实践证明,在一个较大的单位中,试图开发一个单个、高度综合的总的管理信息系统,首要的问题是必须有一个信息系统的总体规划。在总体规划的指导下,按照优先次序分配资源,具体开发一个个子系统,才能保证这些子系统与其他子系统相适应。

系统的总体规划由以下的内容组成。

1. 确立信息系统的战略目标

采用战略集转化方法,把企业总的战略看成一个集合,转化成管理信息系统的目标与战略:确定企业的目标与战略,确立管理信息系统的任务,对环境进行分析评价,制定管理信息系统的目标与战略。

2. 分析企业的信息需求

可采用企业系统规划方法或战略数据规划方法对信息需求进行认真分析。工作的重点是:定义企业过程和数据模型,分析研究现行系统对企业的支持,研究管理部门对系统的要求,确定新信息系统的体系结构,确立新信息系统的实现优先顺序。

3. 制定资源分配计划

在总体规划的指导下,根据各子系统的优先次序,在具备了所需的资源后,就可以进行具体的项目(子系统)的开发了。当然对较小的企业,只要信息需求已经很清楚,也可以直接开发其信息系统。

5.1.2 系统开发阶段的划分

1. 系统准备规划阶段

当一个企业的现行系统因种种原因已不能适用发展的需要,用户提出企业管理信息系

统的开发请求后,就可为此成立一个开发机构负责对现行系统进行初步调查,研究当前企业存在的问题,以及存在的问题是否可以通过一个新的信息系统或修改现行系统就能解决。提出系统开发规划,对用户设想管理信息系统达到的目标、新系统的功能范围、基本工作过程以及对关键性的问题做出明确的描述。同时,还要进行可选方案的经济、技术和社会可行性分析研究,提出可行性分析报告。

2. 系统分析阶段

系统分析的目的是解决“做什么”的问题,它是在可行性分析的基础上,针对现行系统进行全面的调查,分析企业的业务流程,分析数据和数据流程,分析功能与数据之间的关系,并通过使用一系列的图表工具,构造出新系统的逻辑模型。

3. 系统设计阶段

系统设计阶段是解决“怎么做”的问题,它根据新系统的逻辑模型建立系统的物理模型,也就是根据新系统逻辑功能的要求,考虑系统的规模和复杂程度等实际条件,进行若干具体设计。系统设计包括模块设计、代码设计、输入输出设计、文件或数据库设计、可靠性设计等,最后确定系统的实施方案。

4. 系统实施阶段

系统实施是真正解决“具体做”的问题,它是新系统付诸实现的实施阶段。系统实施阶段是具体实现系统设计阶段的新系统的物理模型。它主要包括软、硬件准备,程序设计,数据收集与准备,人员培训,系统测试,系统转换(即新、旧系统的交接)等内容。

5. 系统维护与评价阶段

系统交付使用,投入运行后,需要不断进行维护,修改程序,增加系统功能以适应变化。系统运行一段时间后,要对系统的工作质量和经济效益进行综合评价,整理成系统评价报告,作为系统验收和改进质量的依据。

5.2 管理信息系统的开发方法

管理信息系统的开发方法包括生命周期法、原型法、面向对象的开发方法以及 CASE 开发方法。

5.2.1 生命周期法

生命周期法是建立信息系统最古老的方法,今天依然用于中型或者大型的复杂系统项目的开发上。该方法假设一个管理信息系统像任何生命机体一样,具有周期性,即有开始、成长、成熟、衰退、结束的过程。按照信息系统的建设和使用过程,把信息系统的生命周期划分为 5 个阶段:系统规划,也叫项目定义;系统分析;系统设计;系统实施;系统评价与运行管理。

生命周期法把系统开发分为 5 个阶段,在每个阶段都具有明确的阶段成果,作为和下一个阶段的分界点。系统规划阶段结束之后,形成规划报告;系统分析阶段形成系统分析报告;系统设计阶段形成设计报告;系统实施结束还有项目验收报告。各阶段成果必须经过严格的论证和验收,才能进行下一阶段的工作。整个项目工作必须在项目领导小组的指导和监督下,按计划进行。一个典型的中型开发项目一般需要两年的时间才能交付,还需要一个具有 3~8 年的期望使用寿命。

生命周期法在终端用户和信息系统专家之间有着明确的分工。技术专家(如系统分析员和程序设计员)负责信息系统的分析、设计、实施工作的大部分;终端用户需要给技术人员提供信息需求和审核技术人员设计的信息系统是否满足他们日常业务处理的要求。每当一个阶段成果结束后,技术人员和终端用户需要停下来,就某些关于开发的信息系统中有争议的问题进行协调。下面我们将就每一阶段的工作进行详细描述。

1. 系统规划阶段

系统规划阶段的主要工作是项目定义,项目定义的目的是回答以下问题:为什么我们需要一个新系统项目?我们要用新系统做什么?这个阶段需要确定该组织是否有问题,这些问题是否可以通过建立或改变现有的信息系统得到解决。如果确定此项目是必要的,这一阶段还需要确定系统开发的总目标,说明项目的范围,并制定进一步的工作计划。

2. 系统分析阶段

系统分析阶段将详细地分析现存的系统(这个系统可能是人工的或是自动化的)存在的问题,包括系统管理流程、组织结构、数据流程、数据处理方式等多方面的问题。通过分析,确定解决这些问题的方案,并详细地加以描述。然后,由管理层和项目专家一起审核各个方案的可行性。

系统分析阶段需要广泛地收集和调查系统的文档、报告和工作文件,观察系统是如何工作的,尽可能多地发现系统存在的问题。系统分析阶段所收集到的信息将被用来决定信息系统的的需求,决定下一步的工作。为了很好地完成这些工作,作为系统分析员要和用户很好地沟通,并能够理解用户的需求。因此,系统分析员需要具备很强的技术知识、管理知识和与人沟通的能力。

3. 系统设计阶段

系统设计阶段将把系统分析阶段的方案变成可以实施的逻辑和物理设计说明书,形成的是技术文档,这是程序设计的依据。这一阶段结束也需要和专家和用户一起审核。

4. 系统实施阶段

系统实施阶段包括程序设计和系统的安装、转换等工作。程序设计是严格按系统设计阶段的设计说明书的要求,完成程序的编写和调试。目前,一般使用面向对象的程序设计语言作为编程语言。一个大的系统包含成千上万条程序代码,因此,程序设计也需要程序员之间的合作。程序调试通过之后,新系统就要安装运行。系统安装的工作主要是系统测试,系统测试的目的是保证系统在技术上和功能业务方面正常执行,满足用户要求。系统转换是

新系统替代旧系统的过程。系统通过测试之后,就要替代旧系统。首先,需要进行系统使用与维护的培训;其次,还要制定一个包括系统转换所有细节的正式的系统转换计划。

5. 系统评价与运行管理阶段

系统评价与运行管理阶段通常也叫系统的后实施阶段,包括系统安装之后和运行过程中对系统运行状态的日常记录、维护与评价。用户和技术专家通过一个正式的后期实施审计,判定新系统满足最初的目标的程度和是否需要进一步修改。经过一段时间,系统可能需要很多维护来满足用户的需求。随着环境的变化,用户的需求发生变化,系统将越来越不能满足用户的要求,新系统开发的要求将被提出来。这个过程就像生命体的一个生命周期。

生命周期法强调完备的文档和正式的文件交接,文档资料完备。但是,它也有缺点和局限性。生命周期法不适合所有的系统开发,比较适合结构稳定、需求定义准确的大型事务处理系统和管理信息系统的开发,复杂的技术系统的开发也适合用生命周期法。另外,由于生命周期法具有花费的时间长、不灵活的特点,因此,该方法排斥个性的发挥,需要过早地定义需求,而且不能改变。由于其开发周期长的特点,用户便不能及早地“看见”将要使用的信息系统,因此,可能导致开发过程中用户不能给予进一步的支持。该方法尤其不适合于用户界面的开发,这些系统缺少结构化特性,不同的用户又有个性化的要求。

5.2.2 原型法

针对生命周期法的缺点和不足,产生了原型法开发方法。原型法的基本做法是根据用户的需求,快速生成原型,然后把原型交给用户使用,让用户评价,在此基础上,再修改原型,逐渐达到用户要求。定义需求、设计原型、交给用户使用并评价,这个过程是不断反复的循环过程,直到用户满意为止。这个过程是设计、试验、改进的无数次反复。原型法和传统的生命周期法相比,具有反复循环的工作过程。具体可分为4个步骤。

(1) 确定用户基本需求。系统分析员初步调查用户的系统,找出用户的基本需求。用CASE工具或面向对象的分析设计方法描述系统需求。

(2) 设计原型。系统设计者使用面向对象的软件工具或CASE工具等,快速创建系统工作的原型。

(3) 使用原型。把原型交给用户使用。

(4) 用户评价。用户使用原型之后,就能够评价原型的好坏,是否满足用户的要求,从而启发用户的进一步需求。

设计员根据用户的新要求和修改意见,完善设计模型,然后交给用户使用,再让用户评价。即重复步骤(2)~(4),直到用户满意为止。原型法开发弥补了生命周期法的不足,具有开发周期短、用户满意度高的特点。但是,也有缺点和不足,该方法的使用也有其局限性。当需求和设计方案不能在设计开始就确定的时候,该方法最适合。例如,一个证券公司需要统一的信息系统来分析账户执行情况,但是,应该用什么指标评价账户执行情况是合理的?信息来自人工系统,还是需要从客户账单取得再合并在一起呢?应该比较报告中的哪些项目?最初,用户可能看不出系统应该如何工作,有的使用功能,甚至是系统开发人员想到了用户的前面。另外,原型法在开发用户界面系统方面尤其适合。该方法能使用户立刻对他们将要使用的系统部分做出反应,并提出进一步的需求。

但是,原型法不适合于大系统的开发,可以用于子系统的开发,这种方法往往与生命周期法一起使用。可以用生命周期法开发大系统,到各个子系统的开发中,再采用原型法。采用原型法开发子系统可以鼓励终端用户在系统开发生命周期的过程中积极参与。但是,由于原型法快速开发原型,可能不会形成系统开发过程的详细的文档资料,系统开发的关键步骤可能被掩盖,管理者可能看不到。

5.2.3 面向对象的开发方法

目前,面向对象的程序设计被广泛采用,因此,系统分析设计提供的设计说明书也应该是面向对象的工具图表,这样才能让程序员容易理解。面向对象的开发方法针对面向对象的程序设计的特点,为系统分析设计提供一套图表工具。面向对象的开发方法不同于传统的开发方法,它把业务处理和数据的独立建模转移到数据和处理相结合的对象上。采用类(对象的集合)、对象,以及它们之间的关系描述的方法,定义实体关系。对象像积木一样,被定义、描述、编程、执行和存储。因此,对象具有可以重用的特点。这种方法解决了重用性的问题,减少了代码编写的工作量和费用。采用原型法设计的系统原型,可以用面向对象的程序设计方法快速完成程序设计,缩短系统开发的时间。因此,只有采用面向对象的开发方法设计的原型,才能直接转换成面向对象的程序设计。另外,面向对象的开发方法也适合人的思维习惯和人脑处理问题的方式。因此,采用面向对象开发方法建立的系统分析和设计模型也能够让管理者理解。

采用面向对象的开发方法,在用户界面的详细定义上非常方便。采用这种分析方法,改变了我们过去习惯的面向过程的思维方式。在很多详细的模型设计中,我们可能不需要做处理过程的定义这样的工作了。但是,在系统开发的总体逻辑结构的定义上,系统分析员仍然需要采用生命周期法确定。

5.2.4 CASE 开发方法

计算机辅助软件工程(Computer Aided Software Engineering,CASE)方法是系统开发的自动化方法,是借助于软件开发工具完成系统开发过程中的系统分析和设计的。由于生命周期法开发周期长,开发工作具有不可重复性的特点,一旦系统分析或设计中有错误,或者用户需求发生变化,将导致系统开发工作成本剧增,开发时间增加。因此,系统开发专家探讨出一种用CAD技术设计系统模型的方法。该方法可以用自动化软件,快速完成系统分析和设计工作,给出系统模型的各种图表。如果用户和专家不满意,又可以按用户和专家的要求,重新完成设计模型。

目前,CASE工具很多,许多计算机辅助软件工程工具都提供基于个人计算机的、具有强大图形工具的能力。如产生图表的自动图形工具、屏幕和报告生成器、数据字典、扩充报告工具、分析和检查工具、代码生成器和文档生成器等。多数计算机辅助软件工程工具是基于一个或者多个流行的结构化方法和面向对象的方法的。通常CASE工具有以下6个特点:

- (1) 提供标准的开发方法和设计规则。
- (2) 能够加强用户和专家的沟通。

- (3) 能够完成组织和关联设计,并通过设计知识库,实现快速访问。
- (4) 在容易出错和冗长烦琐的部分,倾向于采用自动化。
- (5) 提供系统分析、设计、代码生成全过程的自动化。
- (6) 提供代码到设计、分析的逆向过程的自动化。

许多计算机辅助软件工具是按照它们是否支持系统开发过程的前端或者后端活动分类的。前端计算机辅助软件工具集中在对系统开发早期的系统分析和设计信息的获取,而后端计算机辅助软件工具解决的是编码的生成、测试和维护活动。计算机辅助软件工具能够自动把数据元素与其连接起来。如果数据流程发生了改变,数据字典中反映这些变化的数据元素也将自动地改变,相应的程序也自动改变。计算机辅助软件工具通过自动修正功能提供了建立原型的工具,支持从分析、设计到程序生成的循环修改工作,节省了开发的时间和人力。

计算机辅助软件工具信息库存储着系统分析员在分析过程中定义的所有信息,包括数据流程图、系统设计结构图、实体关系图、数据字典、处理逻辑定义、屏幕和输出报告格式、注释及测试结果等。另外,计算机辅助软件工具具有支持客户机/服务器计算模式、面向对象程序设计和企业业务流程重组等功能。

5.3 管理信息系统的开发方式

管理信息系统的开发方式有联合开发策略、应用软件包策略和外包策略这几种,下文将逐一介绍。

5.3.1 联合开发策略

联合开发是指组织的 IT 人员和开发公司的技术人员一起工作,完成开发任务。该策略适合于企业有一定的信息技术人员,但可能对信息系统开发规律不太了解,或者是整体优化能力较弱,希望通过信息系统的开发完善培养自己的技术队伍,便于后期的系统维护工作。

合作开发方式需要成立一个临时的项目开发小组,由企业业务骨干(甲方人员)与开发人员(乙方人员)共同组成,项目负责人可由甲方担任或由乙方担任,或者双方各出一位负责人,项目负责人直接对企业的“一把手”负责,紧紧围绕项目开发这一任务开展工作。该项目组是一个结构松散的组织,其人员与运作方式随着项目开发阶段的不同而不同,可根据需要随时增减人员与调整工作方式。

项目组应严格挑选与控制人员,经验告诉我们,在信息系统开发这种特殊的项目中随意增加人员,并不能加快系统开发的进程。该方式强调在开发过程中通过共同工作,逐步培养企业自身的人才。项目开发任务完成后,项目组一般会自行解散,后期的系统维护工作将主要由企业自身的人员承担。

另外,该方式还强调合作双方关系的重要性,建立一种诚信的、友好的合作关系对完成项目是至关重要的。

由于合作开发方式具有很强的针对性与灵活性,在我国被广泛采用,曾经是我国管理信

息系统项目开发中的主流开发方式。它的优点是相对于委托开发方式比较节约资金,可以培养、增强企业的技术力量,便于系统维护工作。缺点是双方在合作中易出现扯皮现象,需要双方及时达成共识,进行协调和检查。

5.3.2 应用软件包策略

应用软件包策略是通过购买应用软件包的办法建设本组织的信息系统,这是目前广泛采用的方法。应用软件包(Application Software Package, ASP)是由软件供应商提供预先编写好的应用软件以及相应的系统建设服务的方法。应用软件包的提供的范围可以是一个简单的任务,也可以是复杂的大型系统的全部管理业务。

应用软件包之所以被广泛采用,一是因为对于很多组织来说,都有共同的特性,如都包括财务管理、人事管理和库存管理等职能。实际上,很多组织都具有标准、统一的工作程序。二是应用软件包策略减少开发时间和费用。当存在一个适合的软件包时,组织就可以直接使用,这样减少了很多开发过程的浪费。三是软件供应商在提供软件的时候,一般都提供大量的持续的系统维护和支持,可以满足用户不断适应市场变化的需要。四是软件供应商提供了先进的工作流程。很多大的软件公司,如德国的 SAP 公司,都有一些高级人才从事流程设计,因此,体现在软件中的管理流程往往是最先进的。

对于组织的特殊要求,软件供应商还可以提供定制(Customization)服务。定制服务允许改变软件包来满足一个组织的特殊需求,而无需破坏该软件包的完整性。一些软件包采用组件开发思想,允许顾客从一组选项中仅仅选择他们所需要的处理功能的模块。

但是,没有一个软件包的方案是完美的。无论多么优秀的软件在解决具体企业的具体问题的时候,都会存在软件中找不到对应的功能部分的问题。因此,软件的二次开发是难免的。即都有定制的要求,大量的定制给项目建设带来风险和困难,因此,选择合适的软件是应用软件包策略首先要考虑的问题。其次,如果定制要求很多,系统建设费用将成倍增长。而这些费用属于隐藏费用,软件包最初的购买价格往往具有欺骗性。另外,项目建设中的有效管理是控制过程成本的有效途径。

5.3.3 外包策略

如果一个组织不想使用内部资源或者没有内部资源来建立信息系统或者运转信息系统,它可以雇佣专门从事这些服务的组织来做这些工作,这种把组织的计算中心操作、远程通信网络或者应用开发转给外部供应商的过程叫做外包。

外包是目前比较流行的信息系统建设方式。主要是因为多数组织认为外包是一种低成本建设信息系统的策略。尤其是对于那些业务波动的组织,外包策略提供给他们的是使用后付费方式,有效地降低了组织的成本。对于软件供应商来说,外包也使他们从规模经营中获得效益。通过提供具有竞争力的服务,外包软件的供应商获得稳定的收益。外包策略有很多优点:①降低成本。②获得标准流程的服务支持。③减少技术人员的需求。④降低信息系统建设的风险。

外包是组织资源外部化的一种方式,因此,外包常常引发一系列问题,如组织可能失去对信息系统的控制,甚至是组织关键资源的控制。当信息系统的控制转向外部的时候,往往

意味着组织商业秘密的外部化。如果组织不限制外包供应商为其竞争对手提供服务或开发软件的话,可能会给组织带来危害。因此,组织需要对外包信息系统进行管理,还需要建立一套评价外包供应商的评鉴标准,并建立相应的约束机制,如在合同中写明提供给其他客户类似的服务要征得该组织的同意等条款。认真设计外购合同是减少风险的一种有效办法。

5.4 管理信息系统开发工具

本节主要讨论了组件和对象、组件的构造、组件设计模式、架构,以及几种不同的组件模型。

5.4.1 基于组件技术的信息系统开发

面向对象技术衍生了组件技术,组件(Component)技术为软件开发提供了改良的方法,这些原理共同建立了一种主要的新的技术趋势。组件技术据称是近二十年来软件领域最大的成就。组件(或面向组件)包含了解决目前危急的软件问题的关键原理。组件具有两个特点:分布性(Distribution)和可重用性(Reusability)。组件原理包括组件的下层构造、软件模式、软件架构、基于组件的开发。

1. 组件与对象的对比

组件是软件系统中具有相对独立功能、接口由契约指定、和语境有明显依赖关系、可独立部署、可组装的软件实体。C++ Builder 中,一个组件就是一个从 TComponent 派生出来的特定对象。组件可以有自己的属性和方法。属性是组件数据的简单访问者。方法则是组件的一些简单而可见的功能。使用组件可以实现拖放式编程、快速的属性处理以及真正的面向对象的设计。

组件可以被认为是面向对象和其他软件技术的化身。区分组件和其他先前的技术有四个原则:封装(Encapsulation)、多态性(Polymorphism)、后期连接(Late Binding)和安全性(Safety)。这个列表与面向对象是重复的,除了删除了继承(Inheritance)这个重点。在组件思想中,继承是紧密耦合的、白盒(White-Box)关系,它对于大多数形式的包装和重复使用都是不适合的。组件通过调用其他的对象和组件重复使用功能,代替了从它们那儿继承。在组件术语中,这些调用称为委托(Delegations)。

所有组件都拥有与它们的实现对应的规范。这种规范定义了组件的封装(例如它为其他组件提供的公共接口)。组件规范的重复使用是多态性的一种形式,它受到高度鼓励,组件技术是软件的可重用性的基础。理想情形是,组件规范是本地的或全局的标准,它在系统、企业或行业中被广泛地重复使用。

组件利用合成(Composition)来建立系统。在合成中,两个或多个组件集成到一起以建立一个更大的实体,而它可能是一个新组件、组件框架或整个系统。合成是组件的集成。结合的组件从要素组件中得到了联合的规范。

如果组件符合客户端调用和服务规范,那么不需要额外编写代码就能够实现交互操作(Interoperate)。这被称为即插即用(Plug-and-Play)集成。在运行时,这是后期执行阶段的

一种连接形似。例如,某个客户端组件可以通过在线目录发现组件服务器(类似 CORBA Trader 服务)。组件符合客户端和服务接口规范后,就能够建立彼此之间的运行时绑定,并通过组件的下部构造实现无缝连接。

理想的情形是,所有组件都将完全符合它们的规范,并且从所有的缺陷中解放了出来。组件的成功运行和交互操作依赖于很多内部和外部因素。安全性(Safety)属性可能是有用的,因为它可以最小化某个组件环境中的全部类的缺陷。随着社会日益依赖于软件技术,安全性已经成为一种重要的法定利害关系,并成为计算机科学研究中的最重要的课题之一。例如,Java 的垃圾收集(Garbage Collection)特性保证了内存的安全性,或者说从内存分配缺陷(在 C++ 程序中这是有问题的)中解放出来了。其他类型的安全性包括类型安全性(Type Safety,用于保证数据类型的兼容性)和模块安全性,它控制着软件扩展和组件合成的效果。

2. 组件的构造

主要的平台厂商都把自己的未来寄托在组件产品线上。特别地,Microsoft 公司、Sun 公司、IBM 公司和 CORBA 社团都已经通过大量的技术和市场投资建立了重要的组件下部构造。这些组件下部构造(Microsoft 公司的 .Net 和 Sun 公司的 Enterprise JavaBeans 包含了 CORBA)都是与行业中面向对象的企业应用程序开发部分竞争的主要的下部构造。这些技术通过共同支持的 XML、Web 服务和其他企业应用程序开发的标准把互相之间的交互操作进行了很大的扩充。

在很多了解 Internet 的组织中 Java 应用程序服务器已经取代了 CORBA 的角色。CORBA 缺乏的是对可伸缩性、可靠性和可维护性的直接的支持。现在这些能力都是大多数 Java 应用程序服务器支持的标准特征了。

组件的下部构造对于软件开发有重要的影响。在很多方面,这些下部构造正在向成为主流开发平台的方向前进。因为它们都变成了可以交互操作的(通过 CORBA IIOP),下部构造模型之间的关系都很好理解了。它们的相似点远远大于它们的专利的差异。

下部构造的选择是讨论得最多的一个问题,但是对于组件的实现而言,其重要性却最低。对于共同开发者来说,最重要的问题会在选择下部构造之后遇到。这些问题包括如何掌握使用这种技术进行设计、如何架构系统、如何协调彼此之间的开发工作。

3. 组件软件的设计模式

软件设计模式由能够应用于所有组件下部构造的软件知识的共同主体组成。软件设计模式是检验过的用于解决特定类别的结构上的软件问题的设计方法,它们已经备案以供其他开发者重复使用。其他重要的模式类别还包括分析模式和 AntiPatterns。分析模式定义了对业务信息进行建模的检验过的方法,它可以直接应用于新软件系统和数据库的建模。

软件设计模式是组件的一个必要的元素。新的、可以重复使用的组件的开发要求的设计、规范和实现都有专家级的质量。经过检验的设计方案对于建立成功的应用程序系列的组件架构和框架是有必要的。通常,在没有检验过的设计观念中偶然性变数太多了。

软件设计模式的流行可以看作是对面向对象在实践中的缺点的反映。AntiPatterns 解释了人们开发面向对象软件系统(以及其他类型的系统)的时候通常犯的错误。想要建立成

功的系统不仅需要基础的面向对象的原理。设计模式解释了有效的软件设计所需要的额外的、改善的想法。分析模式提出了概念和数据的有效建模所必须包含的改善的想法。

在软件开发中彻底改造设计思想仍然是很普遍的,这导致了“试验-错误”实验法的风险和延迟。实际上,大多数软件方法鼓励彻底地改造,把它作为开发的一般模式。给定了需求的改变、技术革新和分布式计算的挑战压力之后,彻底改造对于很多环境都是不必要的风险。这种意见特别适用于组件的开发,在这种情形中缺陷和重新设计的代价可能影响多个系统。

总而言之,软件设计模式可以用知识的重复使用(Knowledge Reuse)来描述。有趣的是大多数模式都被认为像专家级开发者的常识一样简单。但是,对于大多数开发者来说,模式是技术训练中必要的部分,它可以帮助开发者获得世界级的结果。

4. 组件软件的架构

软件的架构涉及从早期的系统概念到开发和操作范围内的系统结构的计划和维护。好的架构是稳定的系统结构,它可以适应需求和技术的改变。好的架构确保了系统的生命周期中人们需求(例如质量)的持续的满意度。可以重复使用的组件是良好的架构的例子。它们支持稳定的接口规范,可以适应改变,是在很多系统环境中重复使用的结果。

软件架构在组件的设计、规范和使用中扮演着重要角色。软件的架构提供了组件设计和重复使用的设计环境。组件在软件架构的预定义方面扮演一定的角色。例如,组件框架就可能预定义了系统的某个重要部分的架构。

5.4.2 组件模型

组件模型定义组件的制作方法、运行方式以及相互作用机制。本节介绍目前流行的两种组件模型:COM(以及建立在COM基础之上的DCOM)组件模型、CORBA组件模型,最后引入一种便于提高组件构造效率的、更真实地反映现实世界的源码级组件模型——KCOM组件模型。

1. COM/DCOM 组件模型

COM(Component Object Model)是一种平台独立的、分布式、面向对象的系统,用于创建可交互操作的二进制软件组件。COM是Microsoft公司的OLE(复合文档)、ActiveX(可用于Internet的组件),以及许多其他技术的基础。COM最初是由Microsoft公司提出并独立发展,现在,它已交给一个独立的组织管理。

为了理解COM以及所有建立在COM基础之上的技术,有一点必须铭记在心,那就是COM不是一种面向对象语言,而是一种标准。COM也不指定一个应用程序应该如何构建。语言、结构,以及实现细节都留给应用开发人员。COM指定一种组件模型以及COM组件(或对象)之间相互作用所需具备的编程要求。COM组件可以分布在同一个进程内,或者在不同的进程间,甚至是远端的机器上。COM组件可以用不同的语言编写,结构也可以有很大的不同。COM被认为是一种二进制标准,因为它是一种当程序被编译成二进制机器码后才起作用的标准。

COM对语言的唯一要求就是必须能创建结构指针,并且能显式地或隐式地通过指针

调用函数。面向对象语言,如 C++、Smalltalk 等,都提供简化 COM 组件实现的编程机制。其他语言环境,如 C、Pascal、Ada、Java、BASIC 等也都能创建和使用 COM 组件。

COM 定义了 COM 组件的本质特征。一般来说,软件是由一组数据以及操纵这些数据的函数构成的。COM 组件通过一个或多个相关函数集来存取组件的数据,这些函数集称为接口,而接口的函数称为方法。COM 组件通过接口指针调用接口的方法。

除了指定二进制组件标准,COM 还定义一些基本的接口以提供一些所有基于 COM 组件的技术所公有的函数。此外,COM 还提供一些所有组件所需要的 API 函数。

DCOM(Distributed Component Object Model,扩展 COM)用以支持不同计算机之间的对象通信,这些计算机可以位于局域网、广域网,甚至是互联网。

DCOM 是 COM 这一世界流行的组件技术的自然发展,因此可以在充分利用已有的基于 COM 的应用程序、组件、工具、知识等的基础之上转向分布式计算。DCOM 使用户能将重点放在真正的商业应用上,而不必关心太多的网络协议细节。

DCOM 位于应用程序的组件之间,将组件以不可见的方式胶合在一起组成具有完整功能的应用程序。

2. CORBA 组件模型

CORBA(Common Object Request Broker Architecture)是 OMG(Object Management Group)制定的开放的、独立于开发者的体系结构和基础构造,通过 CORBA,计算机应用程序可以通过网络协同运作。因为使用标准的 IIOP(Internet Inter-ORB Protocol)协议,两个基于 CORBA 的程序,不管是来自任何开发商、运行在何种机器、采用哪种操作系统和编程语言、通过不同的网络,都可以相互操作。

OMG 成立于 1989 年,是开发商,开发人员,以及最终用户的联盟。OMG 的目标是鼓励一种对象管理体系结构(OMA)的发展和标准化,这种对象管理体系结构提供在异种网络上面向对象的分布式组件的广泛互操作性。CORBA 规范构成 OMA 体系结构的核心。

CORBA 的体系结构是基于面向对象技术的,并且是围绕着三个关键成分构建的:OMG 的接口定义语言(IDL)、对象请求代理(ORB)、标准协议(IIOP)。在对一个分布式系统进行初始的分析和设计之后,紧接着就是定义对象的接口。对象接口表达客户端和服务端应用程序之间的协议。在 CORBA 系统中,使用 IDL(Interfaces Definition Language)来描述接口。IDL 是独立于语言的,它的语法和 C++ 相似,包括模块结构、接口、操作、属性、用户自定义类型、异常等。IDL 并不指明接口的实现方式。不同的 CORBA 产品有不同的处理实现描述的方法。所有的 CORBA 产品都生成基于 IDL 定义的指定语言的绑定。C 语言是第一个由 CORBA 指定的映射语言,此外,OMG 还认可 C++、Java、Smalltalk 等语言绑定。IDL 定义可以储存在一个接口仓库中,客户端可以使用接口仓库进行类型检查以及执行动态接口操作。

ORB(Object Request Broker)是 CORBA 系统的核心。一个 ORB 将客户端应用程序的请求递送到服务器端应用程序。客户端的接口完全独立于对象的实现。对分布式系统开发人员来说,底层的 ORB 实现方式是不重要的。重要的是 ORB 的接口和对象的接口是否有足够好的定义以提供跨整个分布式环境的统一的框架,以及通过 ORB 建立起来的应用程序在不同平台之间是否能很好地沟通。ORB 提供一种将客户端请求传达到目标对象实

现的机制。ORB 通过降低客户端对方法调用细节的了解程度而简化分布式开发,它使得客户端的请求看上去像是本地过程调用。当客户端调用一个方法,ORB 负责找到对象的实现,必要时激活对象,将请求递送给对象,并将任何响应结果返回给调用者。

IIOP 定义一整套数据格式化规则,这些规则称为 CDR,并且是按 CORBA 的接口定义语言所支持的类型来制定的。IIOP 规范还通过使用 CDR 数据格式化规则定义了一套消息类型,这些消息类型支持 CORBA 核心规范中所有的 ORB 语义。CDR 格式化规则和消息类型共同构成一个抽象的协议——GIOP(General Inter-ORB Protocol),GIOP 消息可以通过任何传输协议发送,如 TCP/IP、SPX 协议、SNA 协议等。但是,为了确保 ORB 产品之间的互操作性,IIOP 规范要求 ORBS 发送 GIOP 消息时要通过 TCP/IP,因为 TCP/IP 是 Internet 上的面向连接的标准传输协议。简单来说,GIOP 加上 TCP/IP 就是 IIOP。

CORBA 引人注目的特征包括:①极广泛的平台和编程语言支持;②面向对象方法完全兼容,以保持分布式应用程序的灵活性和可升级性;③具有广泛的开发商、开发人员和用户支持;④开放的、可互操作的体系结构;⑤包含一种灵活的对象间通信的互联网协议标准;⑥支持面向对象的分布式应用程序;⑦成熟的对象概念;⑧接口和实现之间的良好分隔。

CORBA 组件能运行于任何一种平台之上,特别是在 UNIX 平台上,它已经成为编写分布式应用程序以及中间件的标准组件模型。CORBA 从一开始就是为了成为一种跨平台的分布式组件标准,经过多年的发展,已经成为一种成熟、完备的组件模型。CORBA 和 COM/DCOM 之间一直是一种竞争关系,因为 Microsoft 公司只极力推崇自己定义的 COM/DCOM,因此在 Microsoft 公司所提供的流行的开发工具,如 Visual C++、Visual Basic 等,都不支持 CORBA 组件的编程,这对 CORBA 在 Windows 这一最主要的操作系统平台上的普及造成了极大的障碍。

3. KCOM 组件模型

无论是 COM/DCOM,还是 CORBA,它们都烙有很深的计算机软硬件系统的印记。人们构建组件模型的目的是提供描述、反映、模拟现实世界的方法和工具。虽然计算机是实现工具,但是人们的组件模型应该尽量避免受到计算机软硬件系统的影响,想办法构建最接近现实世界的组件模型。此外,一个好的组件模型应该成为普通人的日常工具,而不应该只是计算机专家的专有工具。下面介绍一种全新的组件模型——KCOM 组件模型。

KCOM Space 是一种组件开发及运行平台,KCOM 组件模型是 KCOM Space 的最重要的组成部分。KCOM 这个名称由 K 和 COM 组成。K 是 Knowledge 的首字母,代表知识和智能;COM 是 Component 的缩写,也是 Commerce 的缩写,代表组件以及商务。KCOM 既代表 KCOM 组件模型,也表示它是一种用于实现和运行电子商务的知识化、智能化的工具平台。

KCOM 组件模型建立在这样的观念之上:组件可以层层包裹,同一层次的组件相互之间不直接作用,而是通过父(外层)组件间接作用,父组件相对子(内层)组件而言起了胶合作用。父组件和子组件之间通过消息通信,父组件通过子组件的名称向子组件发送消息。

子组件因为无法知道父组件的名称而只能向外广播消息,隐含的接受者就是父组件。一个组件接收到的消息来自两方面:父组件和子组件。组件的事件与方法本质上是一样

的,从父组件发给子组件的消息称为方法,从子组件发给父组件的消息称为事件。

KCOM 组件的子组件的物理位置可以分布在网络的不同地方,只要能通过某种方式传递消息,父组件和子组件就可以看成一个统一的整体。KCOM 组件的这一特性表现为分布式组件计算。KCOM 组件所包含的子组件可以是动态的,每时每刻都可以有新的子组件加入进来,也可以有子组件离开。一个最明显的例子可以很好地说明动态组件的概念:如果将商店看成一个父组件,而顾客视为子组件,那么顾客和商店之间就构成这样一种动态关系。

由于 KCOM 组件之间是包含的关系,子组件相对父组件是一种局部的关系,因此 KCOM 组件的命名规则就很简单了,只要在同一层组件之间名称不重复就可以了,而每一个 KCOM 组件所包含的子组件的数目是有限的,这样对 KCOM 组件来说,就不必要有像 GUID 这样的全局的标识。对于位于网络不同位置的 KCOM 组件,可以通过网络地址加上组件名称来标识。

KCOM 组件模型的目标就是最真实地反映现实世界。现实世界就是由大大小小的组件构成,大到宇宙,小到原子,组件层层包裹。物质世界如此,人类社会的组成也是这样,从作为个体的人,到集体,到整个社会,都可以用组件加以描述。

5.5 管理信息系统开发管理

本节主要包括管理信息系统开发项目管理的流程、组织机构、基本内容与步骤这些方面。

5.5.1 管理信息系统开发项目管理流程

管理信息系统的开发是一项涉及面广、技术难度大的综合性系统工程,需要投入大量的人力、财力、物力、时间等资源,对整个企业组织的改革与发展会产生很大的影响。只有对企业管理信息系统的整个开发过程按照系统的观点使用现代项目管理的科学理念和方法进行控制,才可能以较小的投入,取得较为理想的效果。

1. 项目的概念

项目管理是指在一定资源如时间、资金、人力、设备、材料、能源、动力等约束条件下,为了高效率地实现项目的既定目标(即到项目竣工时计划达到的质量、投资、进度),按照项目的内在规律和程序,对项目的全过程进行有效的计划、组织、协调、领导和控制的系统管理活动。项目是具有明确目标的一次性任务,具有明显的生命周期,阶段性强。项目管理是面向所有工程项目的管理,是运用系统科学的原理对工程项目进行计划、组织与控制的系统管理方法。项目管理要解决的基本问题就是如何按所选择的研制方法,对开发项目进行有效的计划、组织、协调、领导、控制。

项目的管理是“一把手”工程,领导参与是关键。项目经理必须合理配置项目参加人员,制定项目建设的有关规范,及时产生详尽的报表,正确评价项目的进展情况。

近年来,世界各国都开始对信息系统的建设实施项目管理,不少软件开发商还提供了项

目管理软件,如美国微软公司的 Project 2000、美国 Primavera 公司的 Project Planner P3 和 OS/23.0、我国北京梦龙公司的 PERT 3.0 等。这些软件主要用于编排项目的进度计划,通过资源的分析和成本管理,合理配置资源使计划进度更为合理,同时按计划来安排工程进度,并对进度进行动态跟踪与控制等。

2. 加强信息系统开发项目管理的重要意义

已经在国民经济诸多领域中成功运用的项目管理方法,也完全可以用于信息系统开发项目的管理。管理信息系统开发是一项长期的任务,必须根据企业组织的改革、发展的需要和可能,分成若干项目,分步进行开发。信息系统的“开发项目”包含信息系统分析、设计和实施的整个过程。它由项目负责人(项目经理)负责,利用可获得的资源为用户组织系统的建设。根据系统科学的观点,小项目可以构成一个大项目,一个大项目可以分解成若干个小项目。项目管理实质上是保证整个系统开发项目顺利、高效地完成的一种过程管理技术,贯穿于系统开发的整个生命周期。信息系统开发也是一项系统工程项目,如同其他工程项目一样,研制开发一个信息系统也需要在给定的时间内计划、协调和合理使用配置各种资源。对信息系统进行项目管理的重要性有以下四点:①可以进行系统的思考,进行切合实际的全局性安排;②可为项目人力资源的需求提供确切的依据;③通过合理的计划安排对项目进行最优化控制;④能够提供准确、一致、标准的文档数据。

5.5.2 企业信息系统开发项目管理的组织机构

要想保证信息系统开发工作的顺利启动,首先要建立项目的组织机构——项目组。项目组可以由负责项目管理和开发的不同方面的人员组成,项目组由项目组长或项目经理来领导。一般来说可以根据项目经费的多少和系统的大小来确定相应的项目组。项目组根据工作需要可设若干小组,小组的数目和每个小组的任务可以根据项目规模、复杂程度和周期长短来确定,可以设立的小组有过程管理小组、项目支持小组、质量保证小组、系统工程小组、开发与测试小组、系统集成与安装调试小组等。一个好的项目组不一定能保证项目的成功,但一个差的管理组将肯定会导致项目的失败。因此,在建立项目组时要充分利用项目组每个成员的特长,坚持将正确的开发方法贯穿始终。

1. 项目经理(项目组长)

项目经理(项目组长)是整个项目的领导者,其任务是保证整个开发项目的顺利进行,负责协调开发人员之间、各级最终用户之间、开发人员和广大用户之间的关系。同时他拥有资金的支配权,可以把资金作为强有力的工具来进行项目管理,对项目经理的资金运用情况可采用定期向上级汇报等方法进行合理监督。

项目经理在实施项目领导工作时,要时刻注意所开发的系统是否符合最初制定的目标;在开发工作中是否运用了预先选择的正确的开发方法;哪些人适合于做哪些工作等。只有目的明确、技术手段适合、用人得当,才能保证系统开发的顺利进行。

对于小型项目,项目经理可以独立进行工作,直接管理各类开发技术人员,必要时可以求得外部机构的支持;对于中型项目,应划分出各个任务的界限,由不同的人去管理,项目经理通过这些人来实施各项管理工作;对于大型项目,应有专门的管理机构进行辅助管理,

项目经理应能保证其思想的实施,并通过管理机构对开发技术人员的工作实施管理,同时注意对其产品的审核。

2. 过程管理小组

过程管理小组的任务是负责整个项目的成本及进度控制、进行配置管理、安装调试、技术报告的出版、培训支持等任务,这是一个综合性的机构,用以保证整个开发项目的顺利进行。

3. 项目支持小组

项目支持小组的任务是保障后勤支持,它要及时提供系统开发所需要的设备、材料;负责进行项目开发的成本核算;负责合同管理、安全保证等,特别是对大型项目而言,由于其涉及的资金巨大、开发人员众多、材料消耗也多,尤其要进行科学的管理。

4. 质量保证小组

质量保证小组的任务是及时发现影响系统开发质量的问题并给予解决。问题发现越早,对整个项目的影晌越小,项目成功的把握就越大。

5. 系统工程小组

由于信息系统开发是一项系统工程,因此可以按照工程的一般特性,用系统的观点制定出系统开发各个阶段的任务,这是系统工程小组的工作职责,即将整个开发过程按阶段划分出若干个任务,规定好每个任务的负责人、任务的目标、检验标准、完成任务的时间等。只有明确每一项任务的责、权、利,才能使得开发工作顺利进行。

6. 开发与测试小组

开发与测试小组的任务是充分利用系统开发的一些关键技术、开发模型以及一些成熟的商品软件从事各子系统的开发与集成,并对各子系统进行测试。这是整个开发项目的关键,因此要组织好该小组的成员,并采用统一的方法和标准进行工作。

7. 系统集成与安装调试小组

系统集成是对整个信息系统进行综合的过程,该小组成员在充分注意软件、硬件产品与所开发的信息系统之间的结合,注意最大限度地保证系统可靠性及发挥系统的最高效率的前提下,完成信息系统的软件、硬件等各方面的集成,并做好整个系统的测试与安装调试工作。

5.5.3 管理信息系统开发项目管理的基本内容与步骤

在具体实施管理信息系统开发项目管理时,可按下面 5 个步骤来进行。

1. 任务分解

任务分解(Work Breakdown Structure, WBS)又叫任务划分或工作分解结构,是把整个

信息系统的开发工作定义为一组任务的集合,这组任务又可以进一步划分成若干个子任务,进而形成具有层次结构的任务群。使任务责任到人,落实到位,运行高效。任务划分是实现项目管理科学化的基础,虽然进行任务划分要花费一定的时间和精力,但是在整个系统开发过程中将会越来越显示出它的优越性。

任务划分包括的内容有任务设置、资金划分、任务计划时间表、协同过程与保证完成任务的条件。

任务设置是在统一文档格式的基础上详细说明每项任务的内容、应该完成的文档资料、任务的检验标准等;资金划分是根据任务的大小、复杂程度,所需的硬件、软件、技术等多种因素确定完成这项任务所需的资金及分配情况;任务计划时间表是根据所设置的任务确定完成的时间;协同过程与保证完成任务的条件是指在任务划分时要考虑为了完成该项任务所需要的外部 and 内部条件,即哪些人需要协助、参与该项任务,保证任务按时完成的人员、设备、技术支持、后勤支持是什么等。在进行了任务划分之后,将这些任务落实到具体的人,并建立一张任务划分表,在这张表中标明任务编号、任务名称、完成任务的责任人,其中任务编号是按照任务的层次对任务进行编码,最高层的任务为 1,2,3,⋯,对任务 1 的分解为 1.1,1.2,1.3,⋯,对任务 2 的分解为 2.1,2.2,2.3,⋯,依此类推。

任务分解的主要方法有以下三种:

1) 按系统开发项目的结构和功能进行划分

即可以将整个开发系统分为硬件系统、系统软件、应用软件系统。硬件系统可分为服务器、工作站、计算机网络环境等,考虑这些硬件的选型方案、购置计划、购置管理、检验标准、安装调试计划等内容,制定相应的任务;系统软件可划分为网络操作系统软件、后台数据库管理系统、前台开发平台等,考虑这些软件的选型、配件、购置、安装调试等内容,并制定相应的任务;对于应用软件可将其划分为输入、显示、查询、打印、处理等功能,考虑对系统进行需求分析、总体设计、详细设计、编程、测试、检验标准、质量保证、审查等内容,并制定相应的任务。

2) 按系统开发阶段进行划分

即按照系统开发中的系统分析、系统设计、系统实施及系统实施中的编程、系统测试、系统安装调试、系统试运行、系统运行等各个阶段划分出每个阶段应该完成的任务、技术要求、软硬件系统的支持、完成的标准、人员的组织及责任、质量保证、检验及审查等项内容,同时还可根据完成各阶段任务所需的步骤将这些任务进行更细一级的划分。

3) 将 1) 和 2) 结合起来进行划分

采用这种方法主要是从实际应用考虑,兼顾两种方法的不同特点而进行。

在进行任务划分过程中应特别注意以下两点。

一是划分任务的数量不宜过多,但也不能过少。过多会引起项目管理的复杂性与系统集成的难度;过少会对项目组成员,特别是任务负责人有较高的要求,而影响整个开发。因此应该注意任务划分的恰当性。

二是在任务划分后应该对任务负责人赋予一定的职权,明确责任人的任务、界限、对其他任务的依赖程度、确定约束机制和管理规则。

2. 计划安排

依据任务划分即可制定出整个开发及项目管理计划,并产生任务时间计划表。开发计

划可以划分为配置计划、应用软件开发计划、测试和评估计划、验收计划、质量保证计划、系统工程管理计划和项目管理计划等。

计算机硬件系统、系统软件配置计划包括：

- (1) 建立系统基准。
- (2) 配置、选型、购置、安装调试过程。
- (3) 在变化的情况下如何保持系统基准的稳定。
- (4) 最终产品的文档。

应用软件开发计划包括：

- (1) 将用户需求转化为相应的项目。
- (2) 软件开发过程。
- (3) 集成软件的过程。
- (4) 测试软件的过程。

测试和评估计划包括：

- (1) 整个系统的集成。
- (2) 整个系统的测试。
- (3) 给用户展示系统的工作情况。
- (4) 准备给用户使用系统。

验收计划包括：

- (1) 准备验收文档。
- (2) 如何将最终系统提供给用户。

质量保证计划包括：

- (1) 验证开发质量。
- (2) 确定外部产品质量。

系统工程管理计划包括：

- (1) 管理全部系统开发任务。
- (2) 跟踪用户对系统开发的需求。

项目管理计划包括：

- (1) 何时及如何完成任务。
- (2) 建立完成的策略和标准。
- (3) 各种计划的协调。

计划安排还包括培训计划、安装计划、安全性保证计划等。当这些计划制定出来后,可以画出任务时间计划表,标明任务的开始时间、结束时间,标明任务之间的相互依赖程度。这个任务时间计划表可以按照任务的层次形成多张表,系统开发的子任务可以形成一张表,它是所有子任务时间计划表建立的基础。这些表是所有报告的基础,同时还可以帮助对整个计划实施监控。任务时间计划表的建立可以有多种方法,它可以采用表格形式,也可以使用图形来表达,也可以使用软件工具,其表达方式取决于实际的应用需求。

3. 项目经费管理

项目经费管理是信息系统开发项目管理的关键因素,项目经理可以运用经济杠杆来有

效控制整个开发工作,达到事半功倍的效果。在项目管理中,赋予任务负责人一定职责的同时,还要赋予其相应的支配权,也要对其进行适当的控制。

在经费管理中要制定两个重要的计划,即经费开支计划和预测计划。

经费开支计划包括:

- (1) 完成任务所需的资金分配。
- (2) 确认任务的责权和考虑可能的超支情况。
- (3) 系统开发时间表及相应的经费开支。
- (4) 如果需要变动,及早通知项目经理。

预测开支计划包括:

- (1) 估计在不同的时间所需的经费情况。
- (2) 了解项目完成的百分比。
- (3) 与经费开支计划相比较。
- (4) 允许项目经理做有计划的经费调整。

4. 项目审计与控制

项目审计与控制是整个项目管理的重要部分,它对于整个系统开发能否在预算的范围内按照任务时间表来完成相应的任务起着关键的作用。相应的管理内容和步骤如下:

- (1) 制定系统开发的工作制度。按照所采用的开发方法,针对每一类开发人员制定出其工作过程中的责任、义务、完成任务的质量标准等。
- (2) 制定审计计划。按照总体目标和工作标准制定出进行审计的计划。
- (3) 分析审计结果。按计划对每项任务进行审计,分析执行任务计划表和经费的变化情况,确定需要调整、变化的部分。
- (4) 控制。即根据任务时间计划表和审计结果,掌握项目进展情况,及时处理开发过程中出现的问题,及时修正开发工作中出现的偏差,保证系统开发工作的顺利进行。

对于系统开发中出现的变化情况,项目经理要及时与用户和主管部门联系,取得他们的理解和支持,及时针对变化情况采取相应的对策。

5. 项目风险管理

信息系统开发项目实施过程中,尽管经过前期的可行性研究以及一系列管理措施的控制,但其效果一般来说还不能过早地确定,它与风险联系着,可能达不到预期的效果,费用可能比计划的高,实现时间可能比预期的长,而且,硬件和软件的性能可能比预期的低等。因此,任何一个系统开发项目都应具有风险管理,这样才能充分体现出成本分析的优点。

1) 风险管理注意事项

在风险管理中应注意的是:

- (1) 技术方面必须满足需求,应尽量采用商品化技术,这样可以降低系统开发的风险。
- (2) 开销应尽量控制在预算范围之内。
- (3) 开发进度应尽量控制在计划之内。
- (4) 应尽量与用户沟通,不要做用户不知道的事情。
- (5) 充分估计到可能出现的风险,注意倾听其他开发人员的意见。

(6) 及时采纳减少风险的建议。

总之,风险管理也是项目管理的重要内容,是项目经理的特别职责。

2) 风险管理过程

风险管理过程可以划分为以下几个步骤:

(1) 风险辨识。首先列出一个潜在问题表,然后再考虑其中有哪些问题会出现风险。风险的确定应听取技术专家和广大用户的意见。潜在的风险源包括:①在总体规划和系统分析阶段所进行的需求分析不完全、不清楚、不稳定、不可行,最终影响软件集成和系统集成。设计结果的可用性、可实施性、可测试性较差,影响系统的后续开发工作。②在程序设计过程中,可能出现的非一致性或系统的支持较差。③在整个开发过程中,遇到困难和问题时,开发人员可能出现的矛盾和不协调性将影响系统开发的质量和开发进度。④在实施项目管理过程中,计划的准确性、可监控性、经费运用及分配情况等都将对整个开发工作产生影响。

(2) 风险分析。对辨识出的风险进行进一步的确认后分析风险概况,即假设某一风险出现后,分析是否会有其他风险出现,或是假设这一风险不出现,分析它将会产生什么情况,然后确定主要风险出现最坏情况后,如何将此风险的影响降低到最小,同时确定主要风险出现的个数及时间。

(3) 风险缓和。通过对风险的分析确定出风险的等级,对高级的风险要制定出相应的对策,采取特殊的措施予以处理,并指定专人负责重要风险项目的实施,同时在风险管理计划中进行专门的说明。

(4) 风险跟踪。对辨识后的风险在系统设计开发过程中进行跟踪管理,确定还会有哪些变化,以便及时修正计划。具体内容包括:①实施对重要风险的跟踪;②每月对风险进行一次跟踪;③风险跟踪应与项目管理中的整体跟踪管理相一致;④风险的内容和对项目开发的影响应随着时间的不同而相应地变化。

因此,在项目实施管理过程中,随时研究项目的风险并做出相应的对策是管理工作不可缺少的。通常影响项目内在风险的因素有三个:项目的规模、业务的结构化程度以及项目的技术难度。

3) 风险管理方法

项目管理中,通常根据项目风险水平进行组织和管理。为了搞好项目,管理可采用四种措施和技术:

(1) 项目组与用户结合的外部结合措施和技术。如用户项目管理组织、用户参加的项目小组和用户指导委员会。

(2) 项目组协调工作的内部结合措施和技术。如项目评审会、备忘录和项目组参与决策。

(3) 任务结构化、条理化、规范的计划措施和技术。如关键路线图、抓重大事件以及项目审批程序等。

(4) 估计项目进程的规范化控制措施和技术。如具有差异分析的一系列正式的状态报告。

通常,任务的结构化程度越低,越需要外部与用户的高度结合。采用难度大的高技术项目通常借助于高度的内部项目结合和规范化很低的计划和控制。规范化高的计划和控制对

技术难度低而规模大的项目最为有用。

如果一个风险高的项目获得成功,将能得到最大的期望效益。当冒着某种风险去实现规模大、非结构化的高技术项目时,把具有不同风险和不同项目组织管理的一些项目结合起来,可以使企业获得令人满意的结果。

对信息系统的建设来说,项目管理中风险管理十分重要,因其涉及方方面面的开发人员和广大的最终用户。为了保证系统开发的顺利进行,除了要建立一整套的管理职责和规范,坚持将一种正确的开发方法贯穿始终外,还要做好各类人员的思想沟通,使开发项目组的全体人员自始至终都能保持一个声音说话。

习题

一、选择题

1. 系统可行性研究不包括()。
 - A. 经济可行性分析
 - B. 操作可行性分析
 - C. 社会可行性分析
 - D. 技术可行性分析
2. 外包策略优点有()。
 - A. 降低成本
 - B. 对技术人员数量要求少
 - C. 培养自己的技术人才
 - D. 降低系统建设风险
3. 关于管理信息系统开发项目管理描述正确的是()。
 - A. 任务分解可以按照开发阶段划分
 - B. 对无风险项目无须进行风险管理
 - C. 项目组成员内部沟通比用户沟通更重要
 - D. 不要采纳存在风险的建议

二、名词解释

1. 生命周期
2. 原型法
3. 外包
4. 组件
5. CASE

三、简答题

1. 原型法有什么优缺点?
2. 简述系统分析的目的。
3. 什么是项目管理?
4. 生命周期有哪几个阶段?