CHAPTER 3

第 3 章

交付过程模型与项目管理控制

首先，祝贺你！哈哈，一不小心进入了大赛的第二轮。

从获得“入围”通知开始，为了保证项目过程平稳、顺利地进行，当然也是为了确保能够走到最后，要用规范的软件工程管理理念和工具，对开发过程进行控制和管理，这是我们与非软件工程专业学生的最大区别，也是实训课程的一个主要目标。

在入围的“兴奋”过去之后，团队接下来的事，一定是埋头“赶快把代码写出来！”管你什么需求、什么项目规划，有代码就好像家里有了“大米”，外面下多大的雪，心里也踏实。但是，大家知道，代码有可能都是“臭虫”(Bug)，什么用也没有。在这个时候，可能谁也没有项目经理承受的压力大。学生项目，即使失败了，多少也是学习的过程，损失还不大。在软件企业，上有老板盯着，下有用户催促，往往是不允许失败的(从个人来说，失败就走人，没有什么“人情”好讲的)。但是，在这样的完全不确定的混乱局面下，谁能保证按时、按质完成交付成果呢？如果不能完成，第一个承担责任的就是项目经理，他/她的压力当然大。

作为一个项目经理，他/她这个时候不但要关注交付成果，更要关注交付过程。有句话说：“有好的结果，不一定有好的过程；没有好的过程，一定没有好的结果。”就是这个意思。

3.1 交付过程模型与过程管理

通过学习《现代软件工程》可知：要按时、按质交付软件，面临着复杂的系统、分散的团队、不统一的流程和不兼容的工具等诸多的挑战，所有这些不确定性因素，导致了软件交付结果质量不可保证，交付过程的不可预期、不可管理和不可控制性。为此，软件人从发现“软件危机”开始，就与“不确定性”作斗争，希望有一天，软件开发就像在福特发明的汽车“流水线”上工作一样，谁去上个厕所，都需要协调一致；否则，大家都要停下来等他。

有专家总结软件工程发展趋势的所谓“软件四化”，就是软件开发

的组件专业化、构架平台化、编码自动化和管理工厂化。经过很多年的努力，上述“四化”的端倪已经逐渐显现。

“模块/组件”的专业化，是采用面向对象的方法，通过对行业事务逻辑的业务分解、层次化抽象、封装等技术手段、由特定的、专业化的软件企业（如建筑行业中专门生产预制混凝土楼板、钢制门窗的企业），生产出“标准组件”；其发布、部署和销售，更是通过现在所谓“开源”、开放式的服务组件的方式（云计算），提供应用。而他们自身的营利模式，已经不是卖产品，而是通过卖“服务”获得了。

另一方面，基于 SOA 技术，符合统一标准的模块，可以被有效地整合和重用进现有的系统构架中，从而可以实现各种业务的快速组装，并能满足应用系统的灵活性要求。这就是构架平台化。

最重要的是，在上述两项技术和应用的基础上，软件开发的工厂化管理的理想——既把“个体”的软件开发，变成集人、技术和流程“一体化”的“流水线”式作业和管理，可以得以实现。

我们的实训课程，也希望能让同学们初步体验一下在“软件工厂”中进行开发的过程。

3.1.1 过程模型的一般意义

提到工厂化管理，首先想到的是工厂生产的组织形式，有车间、流水线、工位、工序和按规定操作流程与程序操作的工人等等。这就涉及一个概念——过程。什么是过程？按照过程的一般定义，过程是指为了达到给定目的而执行的一系列活动的有序集合，包括活动的工具、方法、资料或人。例如，坐动车，出发地是上海，中间经过若干站，到达目的地北京。作为乘客，可能只关心几点到达，是否会晚点，最多还会感觉旅途是否舒适。而作为列车运行/服务提供者，则需要关注更多的内容，包括中间站停靠，只能停 2 分钟；否则，将不能保证准点。2 分钟要完成上下客等很多工作，其安排和组织，就不是乘客想得那么简单。

软件过程也是这样。软件过程是将用户的需求转化为有效的软件解决方案的一系列活动，是软件生产的“流水线”。软件工程包括过程，以及过程中所涉及的技术、方法和自动化工具、控制点……它是为软件“流水线”提供服务、支持、支撑和管理的所有东西。采用什么样的方式将这些东西全部组织在一起，在软件工程理论中，就是“软件交付过程模型”或按传统的说法是：软件生命周期模型（交付过程更关注组织内的生产环节和效率）。

先看一张在《现代软件工程》课程里已经很熟悉的图，如图 3-1 所示。

图 3-1 把软件交付过程的成熟度水平划分为 5 个层次。而从过程管理的角度看，要想提高软件组织的成熟度水平，首先要做的，就是建立一套有序的、可检查的过程模型——从无序到有管理。这是学习和实践软件工程的第一步，从“作坊”到规范化开发。

在很多软件工程的教科书中，一讲到规范的软件开发，就罗列大量的文档标准。其实，如果不理解过程的含义和过程管理的基本思想，仅仅套用模板，是没有用的。对于没有多少实践经验的学生而言，更是“如读天书”。因为，规范的文档只是结果，不理解软件过程管理的目标、要求，文档只能是累赘，很多软件企业的实际情况，确实是这样。

环顾目前的软件生命周期模型，有瀑布模型、迭代模型、螺旋模型、RUP 模型、敏捷模型、MSF 模型等，有人总结说，可能有 30 种之多。不同的软件企业，不同的项目、不同的人，可能会选用不同的模型。规范的软件企业有更复杂的过程模型。图 3-2 所示是林锐博士总结的一个软件企业诸多过程交汇的示意图。

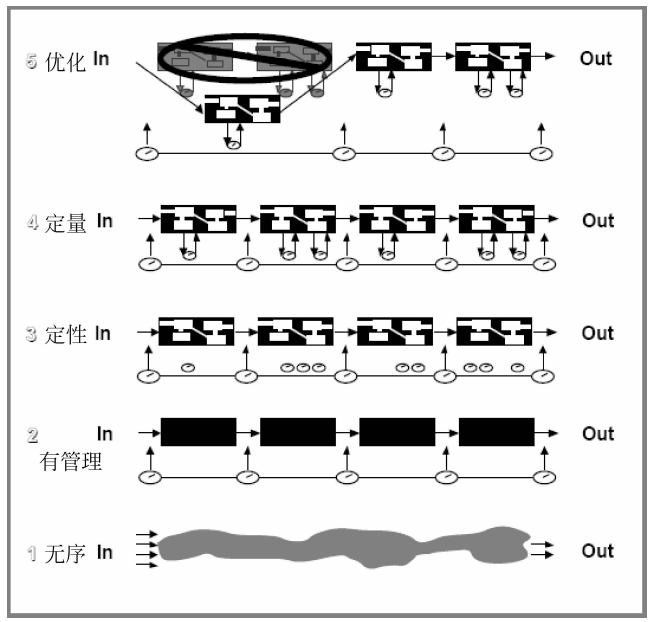

图 3-1　CMM 的 5 个层次

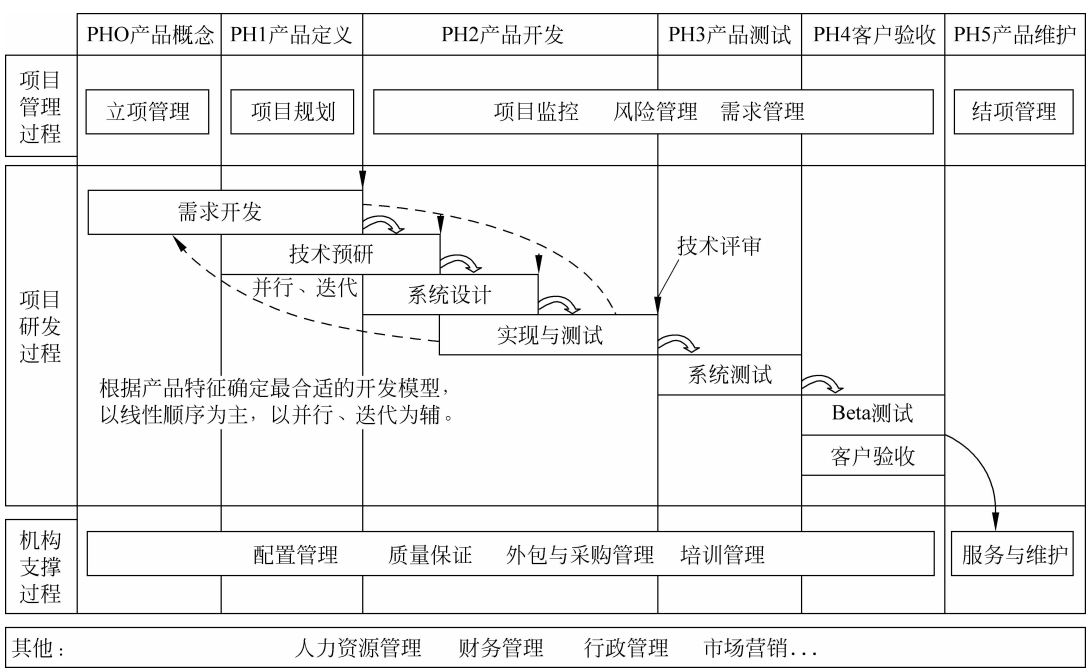

图 3-2　软件企业诸多过程交会示意图

模型真正要解决的问题，是过程的可视性，有了可视性，才可以实现可管理。在图 3-1 的 CMM 第一级中，软件过程是个黑盒，过程不可见，因而被称为是无序的。而 CMM 第二级，是在若干个黑盒的连接点上，实现“里程碑”可见。为此，项目管理人员需要花费大量时间和精力，确定项目的里程碑在哪里，目标要求是什么，以及如何检查当前的状况并与里程

碑要求进行比较，看是否达到了预定的目标，以实现控制（调整）。再往上（更高层次），则是实现更具体、更准确的细节控制。例如，更多的质量要求，量化的度量等。其追求的目标，是精准的过程控制，以导致项目结果的可预测性、可控制性和效能的提升。

现在，不管对这些模型评价如何，是否真的有效，但它终归是朝向克服“软件危机”的方向，迈出了有价值的一步。看一下同学们的项目过程，是否还处在 CMM 的第一级——一个无序的“作坊”式的开发过程？如何提高自己的过程管理能力，是实训项目的一个主要目标。希望能够克服“空对空”的软件工程学习方式，在适合学生项目的模型下，在一个真实的软件过程管理平台上，开发项目，学习软件工程的控制与管理。

3.1.2 微软公司的软件过程模型 MSF 与 VSTS

为了在整个软件开发生命周期中，将软件项目及其开发过程的成效最大化，微软公司开发了解决方案框架（Microsoft Solutions Framework，MSF）。通过 MSF，项目团队可以实现有效地规划、构建、部署和操作解决方案。这些框架模型，来源于微软公司内部进行大规模软件开发和服务项目过程中积累的经验、微软公司顾问专家的经验，以及在世界范围内软件行业里通行的最佳实践。

在预算范围内按期创建一个业务解决方案（如完成了一个软件产品/系统的开发并进行了发布/部署），需要一种经过检验的方法。MSF 为成功地规划、设计、开发和部署 IT 解决方案提供了经过检验的做法。同时 MSF 提供了一个可以伸缩的灵活框架，以满足任何规模的组织或者项目团队的需要。MSF 的核心是一个基础原理、两个模型和若干条用来管理人员、项目和技术元素的准则。

1. MSF 的基础原理

MSF 的核心有 8 个基础原理，包括推动开放式沟通、为共同的前景而工作、赋予团队成员权力、建立清晰的责任和共同的职责、关注交付业务价值、保持灵巧、预测变化、质量投资、学习所有的经验。这些原则的细节部分内容，将会在本书中提到。

这些原理共同传达了 MSF 的观点，构成了一种统一方法的基础，这一方法用来组织项目所需的人员和过程，以便交付技术解决方案。它们是 MSF 结构和应用的基础。尽管每个原理都已经显示出了自身的优势，而且是相互依存的，因为其中任何一个的应用都对另一个的成功起到了支持作用。在依次应用的时候，它们建立了一个稳固的基础，使得 MSF 能够很好地适用于规模、复杂程度和类型都不相同的多种项目。

2. 团队模型概述

为了使一个项目取得成功，必须实现 6 个关键的质量目标，这种理念是 MSF 的基础。这些质量目标驱动团队并定义了团队模型，如图 3-3 所示。虽然整个团队都对项目成功与否负责，团队模型还是将 6 个质量目标和分离的角色群联系起来，以确保义务分明和中心明确。

团队模型的 6 个角色群：产品管理、程序管理、开发、测试、用户体验以及发布管理。这些角色群定义了确定职能领域以及和他们相关联的职责的通用方式。角色群常常仅仅被看作多个角色。无论哪一种解释，这个概念是相同的：解决方案框架和团队模型是可伸缩的，以满足构建一个特别的解决方案的需要。一个角色或一个角色群，可能包含一个或许多人员，这取决于一个项目的大小和复杂程度，取决于为完成功能区内的职责而需要具备的各项技能。

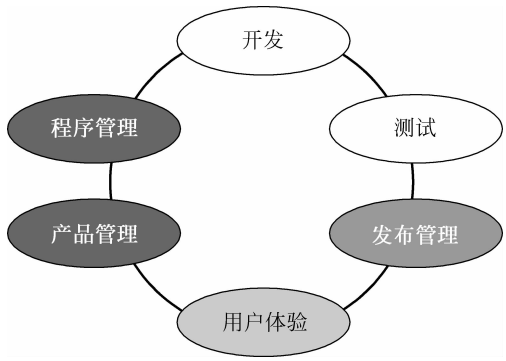

图 3-3 MSF 的团队模型

MSF 团队模型强调将各个角色群与各项业务需求相校准的重要性。角色分组和职能领域与各项职责相联系，职能领域和各项职责分别要求有不同的规则和重心。角色分组为一个协调良好的团队带来了动力。拥有一个清晰定义的目标将促进对各项职责的理解并且鼓励项目团队控制项目，这将最终带来一个更优质的产品。既然每个角色对项目的成功都有决定性作用，那么代表了这些目标的角色在决策时是平等的，具有均等的发言权。

注意：MSF 的这些角色群并不表示任何形式的组织机构示意图或是工作职位分布，因为这些角色群将随着组织和团队的变化而产生改变。更常见的是，角色将分布在 IT 组织内部的不同组群之间，有时还可能分布于业务用户社区或外部的咨询师和合作伙伴中。关键在于清晰的确定履行某一特定角色群的团队个体以及与之相关的有助于目标实现的各种功能、职责和分布。

3. MSF 的过程模型

每个项目都要经过一个生命周期，这是一个包含项目中所有活动的过程，而这些活动的发生，要到项目结束并过渡到操作状态才会结束。生命周期模型的主要功能是建立活动进行的顺序。正确的生命周期模型能够简化项目，并帮助确保每一个步骤都会让项目更加接近成功。图 3-4 所示是 MSF 过程模型生命周期的一个简图。

图 3-4 MSF 的过程模型简图

MSF 过程模型把来自传统的瀑布模型和螺旋模型的概念结合起来，并利用了两者各自的长处。过程模型把瀑布模型基于里程碑规划的优势与螺旋模型不断增加、迭代的项目交付内容的长处结合起来。

MSF 过程模型以阶段和里程碑为基础。在一个层次上，阶段能够被简单地看作是一段时间，只不过强调了为该阶段生产相关交付内容的特定活动。但是，MSF 阶段要比这复杂；每个阶段都有其自身的特色，每个阶段的结束都代表了项目进展和中心点的变化。里程碑是检查和同步点，用来确定阶段的目标是否已经实现。里程碑为团队提供了明确的机会，以调整项目的范围，反映客户或业务要求的变化，并解决项目过程中可能出现的实际风险和问题。此外，里程碑是每个阶段的结束，它按活动的职责进行转化，并鼓励团队以新的视角来看待下一阶段的目标。结束标志则用团队在每个阶段生产的实际交付内容，以及团队和客户对这些交付内容的评价意见说明。这个结束，以及相关的结果，将成为下一阶段的起始点。

MSF 过程模型允许团队响应客户的变更请求，并将需要变化反映到解决方案中。它允许团队先交付部分关键的解决方案，这要比以往的做法更快，因为它首先集中交付优先权最高的特性，然后转到不太重要的特性上，直到最终发布。过程模型是 MSF 的一个灵活组件，MSF 已经被用来成功地改善项目控制、将风险最小化、提高产品质量，以及加快开发速度。MSF 过程模型的 5 个阶段让其足以灵活地应付任何技术项目，无论是应用程序开发、基础结构部署，还是这两者的结合。

4. 用 VSTS 实现 MSF

VSTS(Visual Studio Team System)是微软公司开发的一套高生产力、集成的、可扩展的生命周期开发工具，它扩展了微软公司的 Visual Studio 产品线，增强了软件开发团队中的沟通与协作。利用 VSTS，开发团队能够在开发过程的早期以及在整个开发过程中确保更高的可预见性和更好的质量。

VSTS 很好地实现了微软公司的解决方案框架 MSF，以提供一套经过长期考验的软件开发过程，帮助开发团体交付企业级解决方案。在 MSDN 的 VSTS 主页上有更为详细的介绍：http://msdn.microsoft.com/vstudio/teamsystem/default.aspx。

VSTS 的用户存在于整个软件开发生命周期中，为软件开发项目流程中不同角色的人员提供相应的工具，并且最重要的是将这些工具很好地整合在一起。

所谓可扩展性就是它提供了一套标准的开发接口，开发公司都可以基于该接口开发第三方的组件，从而使该工具更加丰富，灵活而强大，实际上在 VSTS2005 发布后就有很多 ISV(独立软件供应商)发布了他们基于 2005 的插件。

如图 3-5 所示，VSTS 是以角色为基础的，包含项目开发中的各类角色成员：项目经理(Project Manager)、软件架构师(Software Architect)、开发工程师(Software Developer)、测试工程师(Software Tester)、解决方案构架师(Team Foundation Server)。各类角色成员通过使用 VSTS 而在项目开发过程中紧密地结合起来，及时有效地完成角色的任务。

图 3-6 所示为 VSTS 的产品功能，它在软件研发团队中的作用以及与各角色之间的关系。其中 VSTS 的一个重要组成部分就是解决方案构架。它是基于 TFS(Team Foundation Server)的，是团队协作的基础。

图 3-5　VSTS 的角色支持

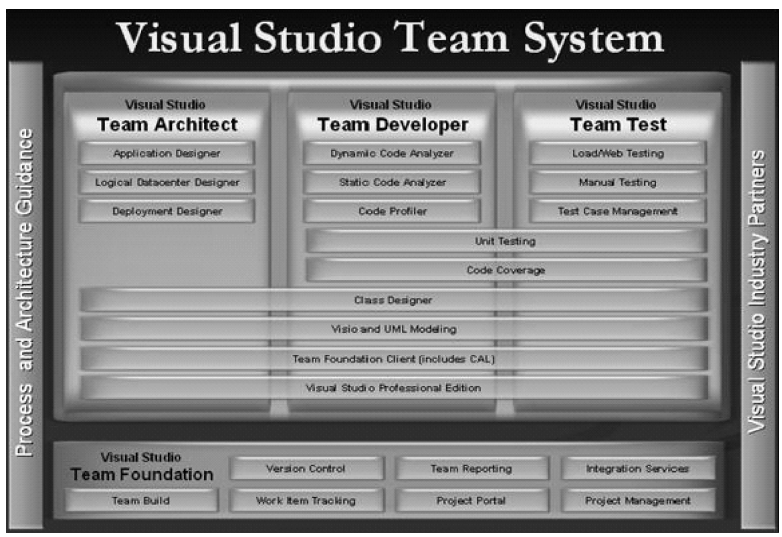

图 3-6　VSTS 的功能构成

VSTS 的产品集主要包括以下几个部分：VSTS2005 Team Edition for Software Architects、VSTS2005 Team Edition for Software Developers、VSTS2005 Team Edition for Software Testers、VSTS2005 Team Foundation Server、VSTS2005 Team Test Load Agent 和 VSTS2005 Team Suite。

Team Suite 是 VSTS 的主要组成部分，为核心软件开发团队中的每个角色成员提供了相应的工具，如软件设计、开发和测试，以及团队沟通和管理方面，使成员之间形成自由、无缝的连接。通过和 Visual Studio Team Foundation Server 的整合使用，Team Suite 帮助项目管理者在软件生命周期中的每个环节，更好地控制产品。

3.1.3　统一过程模型 RUP 与 IBM 的 Jazz

在《现代软件工程》中，介绍了 Rational 的统一过程模型(Rational Unified Process，RUP)，RUP 最早是由 Rational Software 公司提出来的(Rational 公司 2003 年 2 月被 IBM

收购)。因其与当前流行的Java、J2EE技术和面向对象的设计思想紧密地结合在一起,所以在大型的信息技术项目中得到了广泛的应用。与MSF一样,RUP也有6条原则:①使过程适应项目;②平衡相互竞争的涉众优先权;③跨团队合作;④迭代地证明价值;⑤提升抽象层次;⑥持续地关注质量。有兴趣的同学可以找来更多的资料了解RUP的这些原则和思想。

1. RUP的生命周期模型

已知,RUP最重要的三大特点是:软件开发是一个迭代过程、由用例驱动和以构架设计(Architectural Design)为中心。

与MSF一样,RUP强调软件开发是一个迭代模型(Iterative Model),如图3-7所示。RUP定义了4个阶段(Phase):初始化(Inception),详细化(Elaboration),构造(Construction)和移交(Transition)。其中每个阶段都有可能经历处理工作流和支持工作流的各个步骤,只是每个步骤的高峰期会发生在迭代的不同阶段。例如开发实现的高峰期主要发生在构造阶段。实际上这样的一个开发方法论是一个二维模型。这种迭代模型的实现,在很大程度上提供了及早发现隐患和错误的机会。而这个二维模型,与微软公司的MSF模型是何其相似。

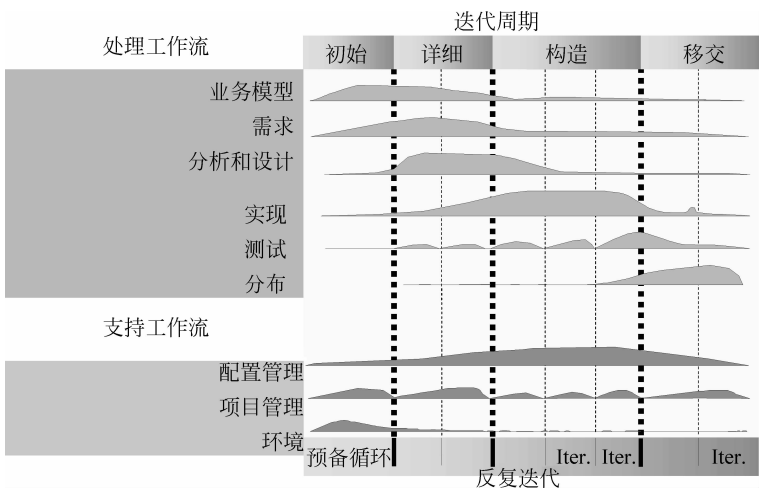

图3-7 RUP的生命周期迭代模型

2. IBM的Jazz平台

Jazz是IBM Rational面向软件交付技术的下一代协作平台。Jazz平台专门面向全球化和跨地域团队开发,通过这一全新的平台,地理上分隔的开发人员将能互相协作,共同构建软件。从而使得软件交付实现更加协作化、高效率和无缝衔接。可以把Jazz技术看成是一个可扩展的框架,可以动态集成和同步与软件开发项目相关联的人力资源、开发过程以及其他资产。

Jazz是一个技术平台,而不是一个具体的产品。基于Jazz平台构建的产品将能为团队软件开发和交付提供一个丰富的功能集合。Rational Team Concert产品家族,如Rational Team Concert Express-C,Rational Team Concert Express以及Rational Team Concert Standard Editions将会是第一个基于Jazz技术构建的产品工具集。

Jazz 使用一种名为“开放商业软件开发”的新形式进行开发。在传统商业开发流程中，新产品或新版本发布前，客户基本上无法了解产品的情况。与此不同的是，Jazz 的开发工作在 Jazz.net 以开放的方式进行。这种开放性和透明性的好处在于，它允许客户成为持续反馈循环的一部分，以便推动开发决策。您可以通过 Jazz.net 了解开发工作的进展情况，并可以下载 Jazz 最新的构建版本，亲自体验 Jazz 带给您及您团队的协作开发新体验。

3. 基于 RUP 的 IBM Rational Team Concert（RTC）产品

IBM Rational Team Concert（RTC）是构建在 IBM Rational 面向软件交付技术的下一代协作平台 Jazz 上的第一个商用产品、一个协作式的软件开发环境，它包含了集成的源代码控制、工作项管理和构建管理等功能。

RTC 是一个可实时相互协作的软件交付环境，可以帮助跨地域分布的开发团队简化协作开发过程，并使其软件交付过程实现自动化管理，如基本的软件版本控制、工作空间管理和平行开发支持。目前，RTC 有 Standard、Express 和 Express-C 版本，适用于小型或中型开发团队，可以帮助项目团队简化、自动化和监管软件交付流程。

RTC 提供的功能如下：

（1）自动化数据收集和报表的功能，减轻了传统软件交付管理上的过度管理问题。

（2）实时监控功能，使得软件项目的监管更加有效。

（3）动态的项目配置（Dynamic Project Provisioning）功能，增强了团队在立项前期的生产力。

（4）实时协作功能，可显著降低资源浪费和返工。

RTC 通过提供整合的工作项目、版本构建、软件配置管理和 Jazz Team Server 提供的协作基础设施，如图 3-8 所示，增强了团队开发的能力。

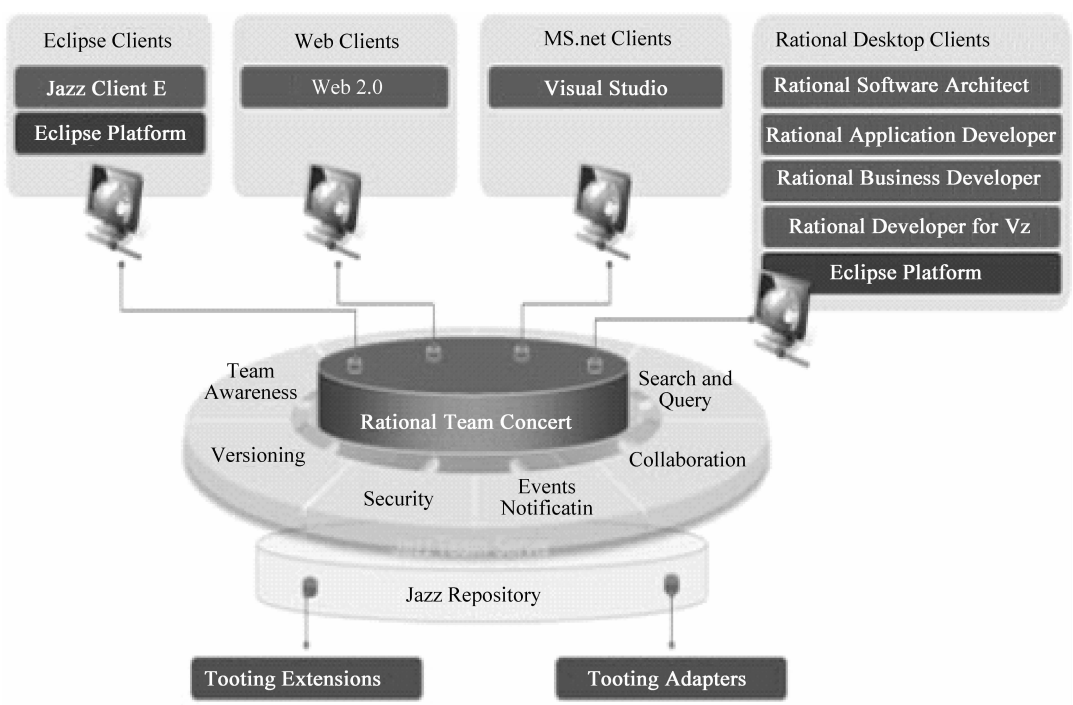

图 3-8　Rational Team Concert 架构图

3.1.4 MSF 与 RUP 的比较

MSF 将过程模型分为两个不同却相互叠加的模型——“团队模型”和“过程模型”，分别描述了软件生命周期中的团队和活动。“团队模型”定义了在项目中工作的角色及其各自的活动和职责，而 MSF 过程模型则将生命周期分为构想、计划、开发、稳定和部署 5 个阶段，每个阶段都描述了一组副产品和应该达到的里程碑。每次经历完 5 个阶段后，便发布一个版本，称为一次迭代。两个模型的叠加，就是 MSF。

RUP 用一个二维结构描述开发过程。横轴代表了 RUP 的动态结构，用迭代的 4 个阶段表示软件开发的生命周期。纵轴代表了 RUP 的静态结构，即每次迭代都包含商业建模、需求、分析与设计、实现、测试、部署、配置和变更管理、项目管理、管理环境 9 个流程，每个流程都包含角色、活动、任务和工件等元素。在迭代的末尾，会得到这次迭代的里程碑，而在每个阶段的末尾，会得到阶段里程碑。阶段、迭代和 9 个流程的集合组成了完整的基于 RUP 的软件开发过程框架。

不论是从各自的原则、流程与活动，还是从模型的角色、过程这两个维度看，MSF 与 RUP 在本质上是十分相似的，所不同的仅仅是细微的语言描述方法而已。相同或相似并不奇怪，因为这两家大公司不论其市场、产品或企业文化有多么的不同，但其软件开发过程，都必须遵循软件生产所特有的规律。两者之间的差异分析，此处不再赘述。

3.2 交付过程模型的结构与关键行为

真正把上述模型变成实际行为的是模型所规定的任务、工作产品、角色与活动，而把这些组织在一起的是流程。图 3-9 所示为 MSF 模型的主要行为结构。

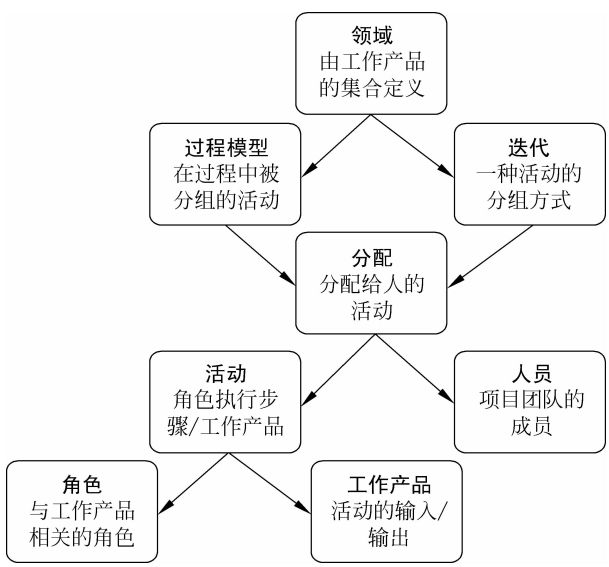

图 3-9 MSF 模型的概念、结构和行为

3.2.1 工作项与工作产品

在《现代软件工程》中，介绍了“配置管理”。实际上，VSTS 与 Jazz 本质上都是软件过程的“配置管理”工具，是配置管理在各自平台上的具体实现。

要实现一个软件过程的配置管理，首要的问题是管理什么？即如何定义配置管理系统中的“配置项”？尽管在各自的系统中，对“配置项”——这个最小的过程管理单位的名称定义有所不同，但作为过程管理工具，其本质并没有太大的区别。

知道了配置项，才能理解 VSTS 或 Jazz 是如何通过对“配置项”的定义、跟踪，进行过程管理的。

在 VSTS 中，配置项被称为“工作项”，与配置管理中定义配置项的 4 大类型不同，VSTS 工作项的范围更广，可以是任务、缺陷、场景、风险、服务质量、需求等，所有你认为是一个“工作”的项。对于初学者来说，这一点未免有点难于把握。

工作项在某个时间点上可能会产生一个结果，这在配置管理中称为“中间制品”，在 VSTS 中是“可交付的产品”，关键可交付产品构成了里程碑的基线。而 RUP 定义了三种不同类型的“工作产品”：可交付的产品、工件和结果。在 RUP 中，工作产品总是任务的结果，其中包含任务将如何执行的细节，而对于 VSTS，工件是阶段的结果。这只是分别定义了不同阶段的工作成果，并给它们取个不同的名字而已。

3.2.2 角色

MSF 的 8 个基础原则之一是：清晰的责任，共同的职责。MSF 将工作进行中需要共同承担的职责和确保工作如期完成需明确的工作责任结合起来。MSF 团队模型基于这样一个前提，即团队中的每个角色都代表了对项目的一种独一无二的观点，但是没有哪个个人能够完全代表所有的不同质量目标。为了解决这一问题，MSF 团队模型把对各种利益相关人的清晰角色职责与实现这个项目成功的整个团队的责任结合起来了。

MSF 的团队模型中有 6 个角色，通称“团队角色”，是基于多组活动的，而 RUP 中所描述的角色是基于职责的，RUP 针对每个流程的活动和任务，都定义了合适的角色执行。因此，RUP 的角色种类繁多，远远超过 6 种。尽管有这些差别，但执行两个框架之间的角色映射也是可能的。可以将 MSF 的一个角色映射为 RUP 中的多个角色。所以，这不存在本质的区别。

3.2.3 流程

流程是对软件过程的进一步细化的划分。对于 MSF 而言，流程按“粒度”由粗到细地定义，包括生命周期阶段、里程碑、状态和基线。并根据任务方面的不同，进一步划分为项目管理流程、风险管理流程和就绪管理流程等。由于这些流程并不单纯地只是一些时间和阶段的定义，还包括角色责任、里程碑目标、交付成果规定等内容，因此有些教科书上把流程也称为规程，意为有规定内容和要求的流程。RUP 将流程与开发过程的“步骤”相关联，并被分为不同的主题，包括开发流程（处理业务流）和管理流程（支撑业务流）两大部分。开发流程有商业建模需求、分析与设计、实现、测试、部署等核心开发流程。管理流程有配置和变更管理、项目管理、环境 3 大核心管理流程。