目录

[什么是Rational统一过程 2](#_Toc494560282)

[6个最佳实践的有效部署 2](#_Toc494560283)

[迭代地开发软件 2](#_Toc494560284)

[需求管理 2](#_Toc494560285)

[基于构件的体系结构 3](#_Toc494560286)

[可视化软件建模 3](#_Toc494560287)

[验证软件质量 3](#_Toc494560288)

[控制软件的变更 3](#_Toc494560289)

[过程简介 3](#_Toc494560290)

[二维结构 3](#_Toc494560291)

[阶段和迭代——时间轴 4](#_Toc494560292)

[初始阶段 4](#_Toc494560293)

[细化阶段 6](#_Toc494560294)

[构建阶段 7](#_Toc494560295)

[交付阶段 8](#_Toc494560296)

[迭代过程 10](#_Toc494560297)

<https://www.ibm.com/developerworks/cn/rational/theme/rational-rup/rup.html>

<https://www.ibm.com/developerworks/cn/rational/r-rupbp/>

# 什么是Rational统一过程

RUP采用了许多现代软件开发过程的最佳实践经验，使得开发团队能有纪律的分派任务和承担责任，规定开发团队在制定需求、设计、和用户交互的时候，能让顾客、合作伙伴参与到开发中，并采用UML来说明解释需求，对于关键的开发活动，能够提供统一的准则、模板和工具，为大部分开发过程提供自动化的工具。它适用于不同规模的项目或开发团队。

RUP是什么？它由什么东西组成？执行过程如何？

地址：

[https://www.ibm.com/developerworks/cn/rational/r-rupbp/#1](https://www.ibm.com/developerworks/cn/rational/r-rupbp/" \l "1)

# 6个最佳实践的有效部署

Rational Unified Process 描述了如何为软件开发团队有效的部署经过商业化验证的软件开发方法。这些方法被称为“最佳实践“，已经被许多团队和公司运用。RUP从下面六个过程为每个团队成员提供了必要准则，模板和工具指导。

1. 迭代地开发软件
2. 需求管理
3. 使用基于构建的体系结构
4. 可视化软件建模
5. 验证软件质量
6. 控制软件变更

## 迭代地开发软件

迭代，就是通过一系列细化，若干个渐进的反复过程而生产有效解决方案的方法。Rational Unified Process专注于处理每个阶段最高风险，减少项目的风险性。因为每次迭代以可执行版本告终，所以用户可以察看每次迭代的版本，提出反馈意见，确保开发人员的需求是正确的。开发人员也可以根据用户要求变更软件的需求。

## 需求管理

Rational Unified Process提供一系列方法来获取、组织需求，并将其文档化，同时还可以跟踪需求的变更，给开发和发布系统提供了连续的可跟踪的线索。在需求的变更中，创建用例来驱动设计、实现和测试，令最终系统更加能满足用户的需求。

## 基于构件的体系结构

*构建是实现清晰功能的模块，子系统。基于构件的模块*，这个过程描述了如何设计**灵活的、易于修改的、直观且便于理解的、可重用的**构件。RUP提供了使用已有构建或者计划中将被实现的构建系统化方法。

## 可视化软件建模

使用可视化软件建模，可以构建软件的体系结构，模块的架构和行为，并且允许你在描述系统的时隐藏细节。他可以：

1. 帮助沟通软件的不同方面
2. 观察各个构件是如何配合在一起的
3. 确保构建模块的行为和代码的行为一致

在RUP中，使用Unified Modeling Language (UML)，它是由Rational软件公司创建的工业级标准。

## 验证软件质量

要验证一个软件的质量，应该从该软件的可靠性、功能、性能，根据需求来验证。Rational Unified Process帮助开发者计划、设计、实现、执行各种测试。质量评估跟随这整个开发过程，对于每个开发活动、每位开发人员，使用客观的测试标准；而非开发完毕后单独进行质量检验。

## 控制软件的变更

开发者应该具有管理软件变更的能力，确定每个变更都是可以接受的、能够被跟踪的，且是必要的。这个过程描述了开发者是如何控制、跟踪、监控修改，来保证开发顺利完成，它同时知道如何通过**隔离修改**和**控制软件的产物**，为每个开发者建立安全的工作区。另外，它通过描述**如何进行自动化集成**和**建立管理**使开发小组如同单个单元来工作。

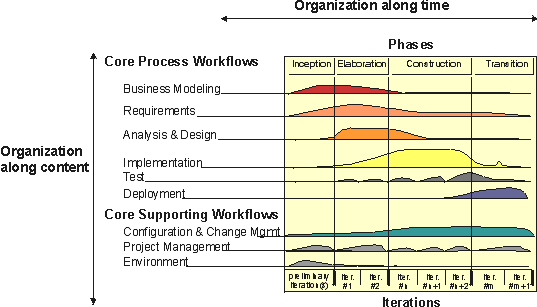
<https://www.ibm.com/developerworks/cn/rational/r-rupbp/#2>

# 过程简介

### 二维结构

开发过程可以用一张二维表来表示：

* 横轴代表了制订开发过程时的时间，体现了过程的动态结构。它以术语周期（cycle）、阶段(phase)、迭代（iteration）和里程碑(milestone)来表达。
* 纵轴表现了过程的静态结构：如何用术语活动（activity）、产物(artifact)、 角色(worker)和工作流(workflow)来描述。

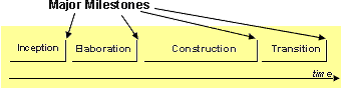


## 阶段和迭代——时间轴

软件生命周期被分解为周期，每一个周期工作在产品新的一代上。Rational Unified Process将周期又划分为四个连续的阶段。

* 初始阶段
* 细化阶段
* 构造阶段
* 交付阶段

每个阶段终结于良好定义的里程碑——某些关键决策必须做出的时间点，此时关键的目标必须被达到。



### 初始阶段

https://www.ibm.com/developerworks/cn/rational/r-rupbp/image003.gif

*初始阶段的目标是为系统建立商业案例和确定项目的边界。*

为了达到该目的必须识别所有与系统交互的外部实体，在较高层次上定义交互的特性。它包括识别所有用例和描述一些重要的用例。商业案例包括*验收规范、风险评估、所需资源估计、体现主要里程碑日期的阶段计划。*

本阶段具有非常重要的意义，在这个阶段中，关注的是整个项目进行工程中的业务和需求方面的主要风险。对于建立在原有系统基础上的开发项目来说，初始阶段的时间可能很短。

本阶段的主要目标如下：

1. 明确软件系统的范围和边界条件，包括从功能角度的前景分析、产品验收标准和哪些做与哪些不做的相关决定
2. 明确区分系统的关键用例（Use-case） 和主要的功能场景
3. 展现或者演示至少一种符合主要场景要求的候选软件体系结构
4. 对整个项目做最初的项目成本和日程估计(更详细的估计将在随后的细化阶段中做出)
5. 估计出潜在的风险(主要指各种不确定因素造成的潜在风险)
6. 准备好项目的支持环境

初始阶段的产出是：

1. 蓝图文档核心项目需求关键特色主要约束的总体蓝图
2. 原始用例模型(完成10%～20%)
3. 原始项目术语表(可能部分表达为业务模型)
4. 原始商业案例,包括业务的上下文、验收规范（年度映射、市场认可等等），成本预计
5. 原始的风险评估
6. 一个或多个原型

#### 里程碑：生命周期的目标

https://www.ibm.com/developerworks/cn/rational/r-rupbp/image004.gif

初始阶段结束时是第一个重要的里程碑：生命周期目标里程碑。

初始阶段的评审标准：

1. 风险承担者就范围定义成本日程估计达成共识
2. 以客观的主要用例证实对需求的理解
3. 成本/日程、优先级、风险和开发过程的可信度
4. 被开发体系结构原型的深度和广度
5. 实际开支与计划开支的比较
6. 如果无法通过这些里程碑，则项目可能被取消或仔细地重新考虑。

### 细化阶段

https://www.ibm.com/developerworks/cn/rational/r-rupbp/image005.gif

细化阶段的目标是分析问题领域，建立健全的体系结构基础，编制项目计划，淘汰项目中最高风险的元素。

为了达到该目的，必须对系统具有"英里宽和英寸深"的观察。体系结构的决策必须在理解整个系统的基础上作出：它的范围，主要功能和非功能性需求。

在细化阶段，可执行的结构原形在一个或多个迭代过程中建立，依赖于项目的范围、规模、风险和先进程度。工作量必须至少处理初始阶段中识别的关键用例，关键用例典型揭示了项目主要技术的风险。通常我们的目标是一个由产品质量级别构件组成的可进化的原型，但这并不排除开发一个或多个探索性、可抛弃的原型来减少如：设计/需求折衷，构件可行性研究，或者给投资者、顾客即最终用户演示版本等特定的风险。

本阶段的主要目标如下：

1. 确保软件结构、需求、计划足够稳定；确保项目风险已经降低到能够预计完成整个项目的成本和日程的程度。
2. 针对项目的软件结构上的主要风险已经解决或处理完成。
3. 通过完成软件结构上的主要场景建立软件体系结构的基线。
4. 建立一个包含高质量组件的可演化的产品原型。
5. 说明基线化的软件体系结构可以保障系统需求可以控制在合理的成本和时间范围内。
6. 建立好产品的支持环境。

初始阶段的产出是：

1. 用例模型（完成至少80％）-- 所有用例均被识别，大多数用例描述被开发
2. 补充捕获非功能性要求和非关联于特定用例要求的需求
3. 软件体系结构描述\_可执行的软件原型
4. 经修订过的风险清单和商业案例
5. 总体项目的开发计划，包括纹理较粗糙的项目计划，显示迭代过程和对应的审核标准
6. 指明被使用过程的更新过的开发用例
7. 用户手册的初始版本（可选）

#### 里程碑：生命周期的结构

https://www.ibm.com/developerworks/cn/rational/r-rupbp/image006.gif

细化阶段结束是第二个重要的里程碑：生命周期的结构里程碑。此刻，应检验详细的系统目标，系统界限、结构的选择，主要风险的解决方案。 主要的审核标准包括回答以下的问题：

1. 产品的蓝图是否稳定？
2. 体系结构是否稳定？
3. 可执行的演示版是否显示风险要素已被处理和可靠的解决
4. 构建阶段的计划是否足够详细和精确？是否被可靠的审核基础支持？
5. 如果当前计划在现有的体系结构环境中被执行而开发出完整系统，是否所有的风险承担人同意该蓝图是可实现的？
6. 实际的费用开支与计划开支是否可以接受？
7. 如果无法通过这些里程碑，则项目可能被取消或仔细地重新考虑。

### 构建阶段

https://www.ibm.com/developerworks/cn/rational/r-rupbp/image007.gif

在构建阶段,所有剩余的构件和应用程序功能被开发并集成为产品,所有的功能被详尽的测试。

构建阶段，从某种意义上说，是重点在管理资源和控制运作以优化成本、日程、质量的生产过程。就这一点而言，管理的理念经历了初始阶段和细化阶段的智力资产开发到构建阶段和交付阶段可发布产品的过渡。

许多项目规模大的足够产生许多平行的增量构建过程，这些平行的活动可以极大地加速版本发布的有效性；同时也增加了资源管理和工作流同步的复杂性。健壮的体系结构和易于理解的计划是高度关联的。换言之，体系结构上关键的质量是构建的容易程度。这也是在细化阶段平衡的体系结构和计划被强调的原因。

本阶段的主要目标如下：

1. 通过优化资源和避免不必要的返工达到开发成本的最小化
2. 根据实际需要达到适当的质量目标
3. 据实际需要形成各个版本（Alpha,Beta,and other test release）
4. 对所有必须的功能完成分析、设计、开发和测试工作
5. 采用循环渐进的方式开发出一个可以提交给最终用户的完整产品
6. 确定软件站点用户都为产品的最终部署做好了相关准备
7. 达成一定程度上的并行开发机制
8. 构建阶段的产出是可以交付给最终用户的产品。它最小包括：
9. 特定平台上的集成产品
10. 用户手册
11. 当前版本的描述

#### 里程碑：初始运作能力

https://www.ibm.com/developerworks/cn/rational/r-rupbp/image008.gif

创建阶段结束是第三个重要的项目里程碑（初始功能里程碑）。此刻，决定是否软件、环境、用户可以运作而不会将项目暴露在高度风险下。该版本也常被称为"beta"版。

构建阶段主要的审核标准包括回答以下的问题：

1. 产品是否足够稳定和成熟得发布给用户?
2. 是否所有的风险承担人准备好向用户移交?
3. 实际费用与计划费用的比较是否仍可被接受?
4. 如果无法通过这些里程碑,则移交不得不被延迟。

### 交付阶段

https://www.ibm.com/developerworks/cn/rational/r-rupbp/image009.gif

交付阶段的目的是将软件产品交付给用户群体。

只要产品发布给最终用户，问题常常就会出现：要求开发新版本，纠正问题或完成被延迟的问题。

当基线成熟得足够发布到最终用户时，就进入了交付阶段。其典型要求一些可用的系统子集被开发到可接收的质量级别及用户文档可供使用，从而交付给用户的所有部分均可以有正面的效果。

这包括：

1. 对照用户期望值，验证新系统的"beta测试"
2. 与被替代的已有系统并轨
3. 功能性数据库的转换
4. 向市场、部署、销售团队移交产品

构建阶段关注于向用户提交产品的活动。典型的，该阶段包括若干重复过程，包括 beba 版本、通用版本、bug 修补版和增强版。相当大的工作量消耗在开发面向用户的文档，培训用户。在初始产品使用时，支持用户并处理用户的反馈。开发生命周期的该点，用户反馈主要限定在产品性能调整、配置、安装和使用问题。

1. 本阶段的目标是确保软件产品可以提交给最终用户。本阶段根据实际需要可以分为几个循环。本阶段的具体目标如下：
2. 进行 Beta 测试以期达到最终用户的需要
3. 进行 Beta 测试和旧系统的并轨
4. 转换功能数据库
5. 对最终用户和产品支持人员的培训
6. 提交给市场和产品销售部门
7. 和具体部署相关的工程活动
8. 协调 Bug 修订/改进性能和可用性(Usability)等工作
9. 基于完整的 Vision 和产品验收标准对最终部署做出评估
10. 达到用户要求的满意度
11. 达成各风险承担人对产品部署基线已经完成的共识
12. 达成各风险承担人对产品部署符合 Vision 中标准的共识
13. 该阶段依照产品的类型,可能从非常简单到极端复杂的范围内变化。例如，现有的桌面产品的新版本可能非常简单，而替代国际机场的流量系统会非常复杂。

#### 里程碑：产品发布

https://www.ibm.com/developerworks/cn/rational/r-rupbp/image010.gif

在交付阶段的终点是第四个重要的项目里程碑，产品发布里程碑。此时，决定是否目标已达到或开始另一个周期。*在许多情况下，里程碑会与下一个周期的初始阶段相重叠。*

1. 发布阶段的审核标准主要包括以下两个问题：
2. 用户是否满意？
3. 实际费用与计划费用的比较是否仍可被接受？

### 迭代过程

Rational Unified Process 的每个阶段可以进一步被分解为迭代过程。迭代过程是导致可执行产品版本（内部和外部）的完整开发循环，是最终产品的一个子集，从一个迭代过程到另一个迭代过程递增式增长形成最终的系统。

**迭代方法的益处**

与传统的瀑布式方法相比，迭代过程具有以下的优点：

1. 减小了风险
2. 更容易对变更进行控制
3. 高度的重用性
4. 项目小组可以在开发中学习
5. 较佳的总体质量