软 件 工 程 技 术 丛 书

软件开发过程系列

RUP导论

（原书第3版）

The Rational Unified Process

An Introduction

(Third Edition)

（美） Philippe Kruchten 著

麻志毅 申成磊 杨 智 译

机械工业出版社
China Machine

软 件 工 程 技 术 丛 书

软件开发过程系列

RUP导论

（原书第3版）

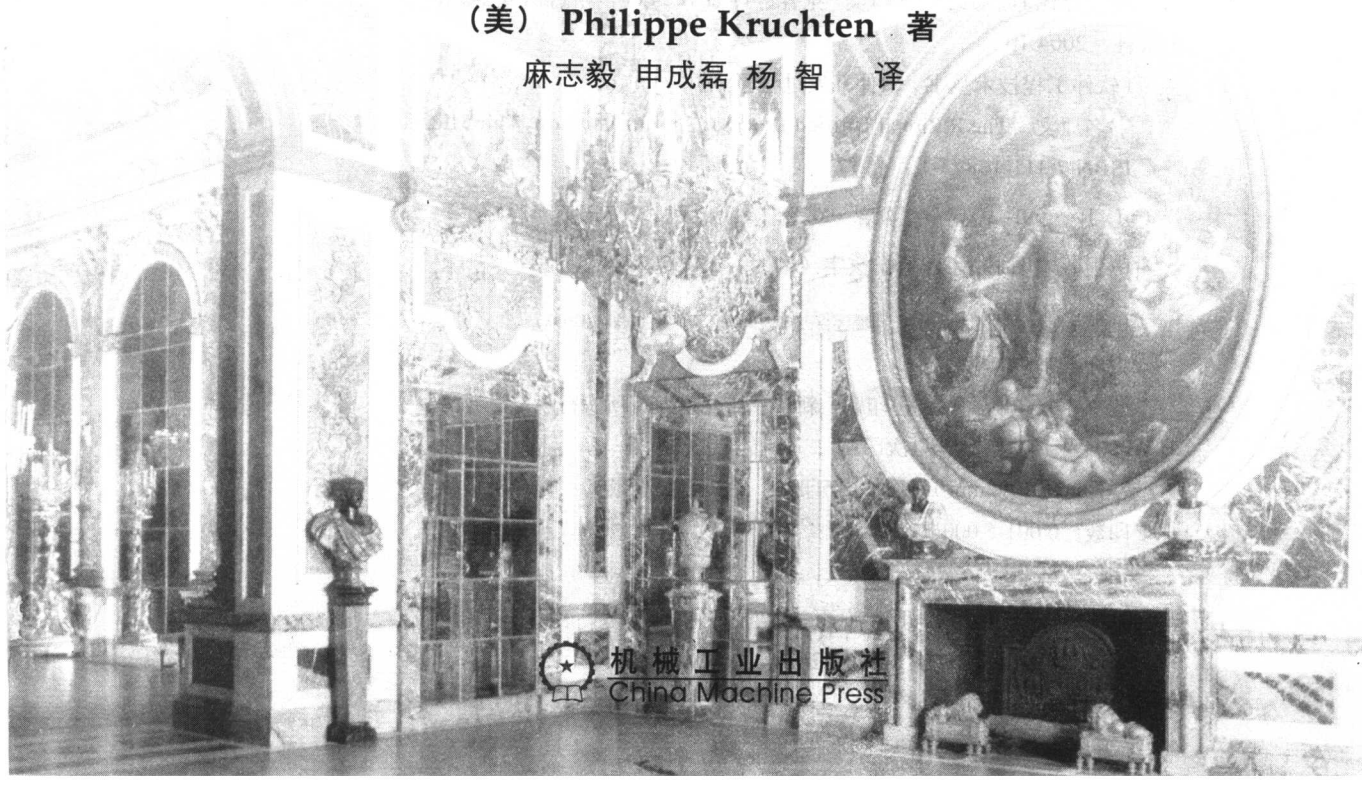
The Rational Unified Process

An Introduction

(Third Edition)

（美） Philippe Kruchten 著

麻志毅 申成磊 杨 智 译



机械工业出版社
China Machine Press

RUP(Rational Unified Process)是Rational公司开发的一种软件工程过程，是开发组织用于分配和管理任务及职责的规范化方法，其目标是帮助开发人员在预定的进度和预算范围内开发出符合最终用户需求的产品。

本书依照RUP 2003进行编写，全面而简洁地介绍了RUP的概念、结构、内容和动机，帮助读者学会如何开发出高质量的软件。本书作者是RUP的首席架构师，本书融入作者几十年的开发经验，极具实用性。本书适合所有参与软件开发的人员阅读。

本书版权登记号：图字：01-2004-0297

图书在版编目（CIP）数据

RUP导论 (原书第3版) / (美) 克鲁奇特 (Kruchten, P.) 著; 麻志毅等译. -北京：机械工业出版社，2004.10

（软件工程技术丛书　软件开发过程系列）

书名原文：The Rational Unified Process: An Introduction, Third Edition

ISBN 7-111-14823-1

Ⅰ.R…　Ⅱ.①克…　②麻…　Ⅲ.软件开发　Ⅳ.TP311.52

中国版本图书馆CIP数据核字（2004）第065004号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：朱　劼

北京中兴印刷有限公司印刷 · 新华书店北京发行所发行

2004年10月第1版第1次印刷

787mm × 1092mm 1/16 · 14.75印张　（彩插0.5印张）

印数：0 001-5 000册

定价：29.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

译者序

目前，越来越多的领域中的用户都希望把更多、更难的问题交给计算机去解决。这使得计算机软件的规模和复杂性与日俱增，软件技术也不断地受到新的挑战。开发一个具有一定规模和复杂性的软件系统和编写一个简单的程序的方法大相径庭。大型的、复杂的软件系统开发是一项工程，必须按照工程学的方法组织软件的生产与管理，这是人们从软件危机中获得的最重要的教益。开发软件系统时，建模和编程都很重要，但也要遵循合适的开发过程。统一软件开发过程就是目前流行的开发过程之一。

针对《统一软件开发过程》一书中所描述的更通用的过程，RUP为其一个特定的详尽实例。RUP综合了许多现代软件开发的最佳实践，并使用一种可裁剪的方式来表达，以适用于各种各样的项目和组织。它以一种详尽、实用的方式将这些最佳实践在线地提供给项目组。其目的是帮助软件开发人员在预定的进度和预算范围内，开发出符合最终用户需要的高质量软件。

本书讲述了RUP中的主要概念、结构、内容以及动机，各种相关的技术细节可从IBM的在线知识库中获得。

本书按照RUP 2003进行了更新。与第2版相比，本书中的几个术语发生了变动，引入了新元素，对部分过程进行了一些重新设计，并更深入地将RUP开发为一个过程框架。在本书中结合了许多读者的反馈意见。

本书适合所有参加软件开发的人员，如项目经理、质量工程师、过程工程师、研究方法的专家、系统工程师和程序员等使用。打算采用RUP的单位可根据自己的实际需要，对该过程进行裁剪。

RUP本身处于不断的进化之中。机械工业出版社已于2002年出版了《The Rational Unified Process: An Introduction》第2版的中文译本《Rational统一过程引论》，本书是第3版的中文译本。在翻译本书的过程中，有很多地方借鉴了第2版的中文译法，在此向第2版的译者表示感谢。

由于译者自身的水平有限，译文难免存在错误和疏漏，希望读者给予批评与指正。

译 者

2004年5月于北京

前 言

RUP (Rational Unified Process)最初是由Rational软件公司（现在已合并到IBM公司）开发并向市场推广的一个软件工程过程。它是开发组织用以分配和管理任务及职责的一个规范化方法。RUP的目标是在预定的进度和预算范围内开发出符合最终用户需要的高质量软件。

RUP综合了许多现代软件开发的最佳实践，并使用一种可裁剪的方式来体现这些实践，以适用于各种项目和组织。它以一种详尽、实用的方式将这些最佳实践在线地提供给项目团队。

本书介绍了RUP的概念、结构、内容以及动机。

本书的目标

利用本书，你将学习到如下内容:

- 何为RUP。
- RUP的术语及如何理解其结构。
- 在RUP中综合的最佳实践的优点。
- 在一个项目中，RUP如何向你提供完成特定职责所需要的指导。

本书没有涵盖完整的RUP，而只涉及其中一部分内容，用以介绍RUP。在完整的RUP中，你可以找到进行开发工作所需要的详细指导。完整的RUP产品——在线知识库，可从IBM获得。

本书大量引用统一建模语言（Unified Modeling Language，UML）的内容，但它并不是一本介绍UML的书。如果读者希望了解UML，请参考以下两本书:《UML用户指南》和《UML参考手册》[⊖]。

本书还讲授了建模和面向对象技术，但并没有讲授设计方法，也没有讲解如何建模。有关嵌入RUP中的各种技术的详尽步骤和指南，只能在过程产品中找到。

本书有几章讨论了项目管理问题，描述迭代开发计划和风险管理等主题。然而本书决不是一本完整的有关项目管理和软件经济学的手册。如果要得到这方面的更多信息，请参考《软件项目管理: 一个统一的框架》这本书[⊜]。《统一软件开发过程》[⊝]这本书描述了更加通用的软件开发过程，而本书只介绍一个特殊而详细的实例。

⊖ 《UML用户指南》（The Unified Modeling Language User Guide）和《UML参考手册》（The Unified Modeling Language Reference Manual）英文原版由Addison-Wesley公司于1999年出版，中文版由机械工业出版社出版。——编辑注

⊜ 《软件项目管理: 一个统一的框架》（Software Project Management: A Unified Framework）英文原版由Addison-Wesley公司于1998年出版，中文版由机械工业出版社出版。——编辑注

⊝ 《统一软件开发过程》（The Unified Software Development Process）英文原版由Addison-Wesley公司于1998年出版，中文版由机械工业出版社出版。——编辑注

谁应该阅读这本书

本书是为参与软件开发的各类人员撰写的，这些人员包括：项目经理、开发人员、质量工程师、过程工程师、方法专家、系统工程师以及分析员。

本书尤其适合那些已经采纳或即将采纳RUP的开发组织中的人员阅读。通常，一个组织可能会裁剪RUP来适应自己的需求，但是本书中描述的核心过程在RUP的所有实例中都应该保持统一的特性。

本书对于学生学习由IBM Rational软件公司以及其业界和学术界的伙伴提供的专业教程也是一本很好的补充教材，它提供了这些教程中涉及的一些主题的相关内容。

本书假定读者已对软件开发具有一定的理解，但不要求读者具备编程语言、面向对象方法或统一建模语言（UML）方面的特定知识。

如何使用本书

在全部或部分采用RUP的组织中工作的软件专业人员应该顺序地阅读本书。本书的结构按RUP自身特性的顺序组织。

项目经理可以只读第1章、第2章、第4章和第7章。这些章介绍风险驱动的迭代开发过程的含义。

过程工程师或方法学专家可能需要裁剪RUP，并在他们的开发组织中建立这种过程。他们应该认真学习第3章、第14章和第17章，这些章描述了过程结构以及全面实施RUP的方法。

本书的组织和特征

本书分为两部分。

第一部分介绍过程、内容、历史、结构和软件开发生命周期，还描述了RUP区别于其他软件开发过程的关键特征。

- 第1章　最佳的软件开发实践
- 第2章　RUP
- 第3章　静态结构：过程描述
- 第4章　动态结构：迭代开发
- 第5章　以架构为中心的过程
- 第6章　用况驱动的过程

第二部分给出了过程的各种构件或者规程的概况。每一章对应一个规程。

- 第7章　项目管理规程
- 第8章　业务建模规程
- 第9章　需求规程
- 第10章　分析和设计规程
- 第11章　实现规程

- 第12章　测试规程
- 第13章　配置和变更管理规程
- 第14章　环境规程
- 第15章　部署规程
- 第16章　典型的迭代计划
- 第17章　实施RUP

大多数介绍规程的章由六个部分组成:

- 规程的目的。
- 定义和关键概念。
- 角色和制品。
- 一个典型的工作流: 活动的概况。
- 工具支持。
- 小结。

书后用附录总结了从第7章~第15章中介绍的过程的所有角色和制品（过程的工作产品）。最后提供了缩写词和术语表以及一个带简短注释的参考文献。

获得更多的信息

关于RUP的资料，如数据表和可下载的演示版本，可以通过因特网从IBM Rational软件公司的网站www.rational.com/rup/获得。

假如你已经在使用RUP，可以从Rational开发者园地（Rational Developers Network，RDN）获得更多的资料，包括额外的产品、更新以及合作者链接。在RUP的在线版本中可以找到到RDN的超级链接。

学术机构可以与IBM取得联系，以获得将RUP作为课程的资料。

第2版

《The Rational Unified Process: An Introduction》的第2版按照“RUP 2000”进行更新。

第3版

本书按照“RUP 2003”进行更新。在本版本中，几个术语发生了变动，对部分过程进行了一些重新设计（特别是测试和环境规程），并更深入地将RUP开发为一个过程框架。本版中还引入了新的元素: 过程构件和过程插件的概念，一个由RUP Modeler、RUP Organizer、RUP Builder和MyRUP构成的新工具集，以及一个独立的过程工程过程。在本版本中结合了许多读者的反馈意见。

致谢

RUP凝聚了来自Rational软件公司以及其他各方面软件专业人员的共同智慧。本书第2章介绍了这一过程的历史。但是要把各个方面人员的共同智慧综合成一本书，即使是像这样一本概论式的书，也需要一批人付出极大的热忱和努力。在这里，我把他们一一介绍给读者。

过程开发组(Process Development Group)的早期成员创建了RUP，并对本书做出了贡献。首先是Stefan Bylund，然后是Kurt Bittner，最后是Bruce Macisaac对有关分析和设计的章做出了贡献。Maria Ericsson与Leslee Probasco开发了业务工程和需求管理的有关内容。测试部分开始于Bruce Katz，然后由Paul Szymkowiak完全重新设计。John Smith扩展了RUP 2000的项目管理方面的内容。Jas Madhur对配置管理、变更管理和部署贡献了很多见解。Håkan Dyrhage对过程的组织和结构及其实施和配置贡献了许多想法。Margaret Chan和Susan Buie负责产品集成以及本书绝大多数原图的汇编工作。Sigurd Hopen在Björn Gustafsson的帮助下开发了PEP。

Per Kroll是开发组的经理，Mike Barnard是产品经理。Alfredo Bencomo、Chinh Vo、Philip Denno、Lars Jenzer、Glenys Macisaac以及Fionna Chong开发了产品的基础结构。Debbie Gray是这个横跨9个时区的团队的尽职尽责的行政助理。

RUP以及本书得益于过去8年来许多人的评阅和思想。他们包括：Stefan Ahlqvist、Dave Bernstein、Grady Booch、Murray Cantor、Geoff Clemm、Catherine Connor、Mike Devlin、Christian Ehrenborg、Ian Gavin、Christina Gisselberg、Sam Guckenheimer、Björn Gustafsson、Matt Herdon、Ivar Jacobson、Paer Jansson、Ron Krubek、Dean Leffingwell、Andrew Lyons、Bruce Malasky、Roger Oberg、Gary Pollice、Terri Quatrani、Walker Royce、Jim Rumbaugh、Ian Spence、John Smith以及Brian White。

我们非常感谢Grady Booch为本书撰写了第1章。

特别感谢Rational领域的人员、我们的Chat_RUP列表中的成员以及RUP论坛的成员所提供的反馈意见和做出的贡献。

最后要深深感谢本书的编辑Mary O'Brien、本书前任编辑J.Carter Shanklin以及Brenda Mulligan、Amy Fleischer和她的团队，还要感谢Addison-Wesley的所有员工，正是由于他们的努力才使得本书能够尽快问世。

Philippe Kruchten
温哥华 B.C.，加拿大

目 录

第二部分 过程规程

附 录

第一部分

过　程

第1章
最佳的软件开发实践

——由Grady Booch撰写

本章探讨了最佳的软件开发实践，并对RUP做了大体的介绍。

1.1 软件的价值

软件是运营现代化的业务、政府管理以及加强社会联系的动力，软件也以前所未有的方式帮助我们创建、访问信息，并使信息可视化。从全球范围来看，软件业的飞速发展推动了世界经济更快地向前发展。从个人的角度来看，软件密集型产品可以帮助治愈疾病，可以让聋哑人听到声音，让受伤者恢复活动，让不健全的人获得更多的机会。总之，软件已经成为我们这个现代世界中不可缺少的一部分[⊖]。

1
~
3

对于软件专业人士来说，一个好消息就是世界经济越来越依赖于软件。各种在技术上可行并且满足社会需求的软件密集型系统在规模、复杂度、分布和重要性上都在不断地进行着扩张。

这种扩张不利的一面是我们所熟悉的软件开发方法受到了限制，将传统的系统升级为更现代化的技术产品也会带来一系列的技术和组织问题。综合而言，随着业务的快速发展和展开，对生产力和产品质量的要求也越来越高。但同时，却没有足够的合格的开发人员以满足这种快速发展的需求。

网络使构造和维护软件产品变得越来越困难，而以循环的、可预测的方式构造高质量的软件产品仍很困难。

1.2 软件开发问题的症状和根本原因

不同的软件开发项目可能由于不同的原因而导致失败。遗憾的是，有太多的项目最终都失败了。我们可以从这些项目中找出一些共同的症状[⊖⊜]：

- 对于最终用户的需求理解得不够精确。
- 不能处理需求变更。
- 模块之间不兼容。
- 软件不易维护和扩展。
- 对项目的严重缺陷发现较晚。

⊖ Grady Booch, "Leaving Kansas," *IEEE Software* 15(1), 1998年1~2月，第32~35页。

⊜ Capers Jones, *Patterns of Software Systems Failure and Success*, London: International Thompson Computer Press, 1996。

⊜ Edward Yourdon, *Death March: Managing "Mission Impossible" Projects*, Upper Saddle River, NJ: Prentice-Hall, 1997。

- 软件质量低劣。
- 软件性能无法令人接受。
- 团队中人员按各自的开发方式工作，这使得对谁在何时、何处以及为什么做出什么更改进行重构难以进行。
- 一个不可靠的构造和发布过程。

4

但是，治愈这些症状不等于治愈疾病。例如，对项目的严重缺陷发现较晚只是一个更大问题的一个症状，而这个大问题可能是对于项目状况的评估过于主观，并且没有发现项目需求、设计和实现中的不一致。

尽管不同的项目失败的原因是不同的，但是基本上大多数项目的失败是由于以下几个根本原因的组合造成的:

- 特别的需求管理。
- 模糊和不精确的交流。
- 脆弱的架构。
- 过度复杂。
- 未检测出需求、设计和实现中的不一致。
- 测试不足。
- 对项目状况的评估过于主观。
- 未解决存在的风险。
- 无法控制变化的传播。
- 自动化程度不足。

1.3 最佳的软件实践

如果解决了这些根本问题，那么不仅消除了那些症状，而且可以更好地以一种循环的、可预测的方式来开发和维护高质量的软件产品。

这就是最佳的软件实践，即在商业中综合运用能够解决软件开发中的根本问题的策略[⊖]。之所以称之为“最佳实践”，不仅因为可以对它的价值进行精确地量化，而且因为业界成功的组织普遍应用它。本章中我们将描述6项重要的最佳实践:

5

1）软件的迭代开发。
2）管理需求。
3）应用基于构件的架构。
4）为软件建立可视化的模型。
5）对软件质量进行持续的验证。

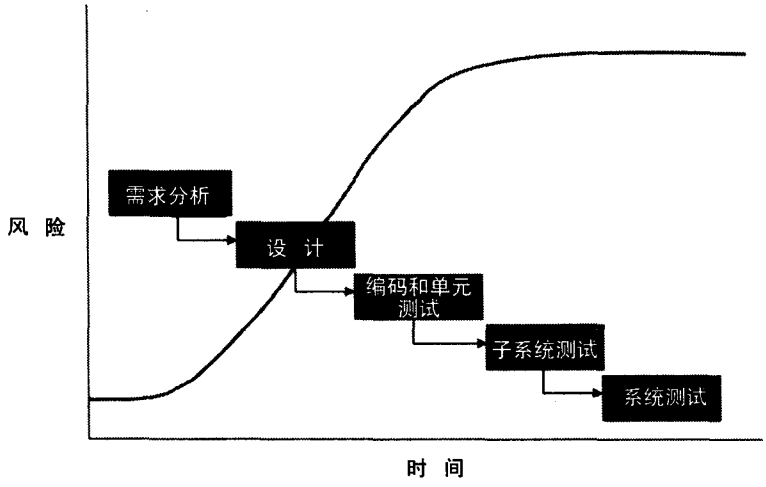
⊖ 参见“the Software Program Manager's Network”上的最佳实践工作（http://www.spmn.com）。

6）控制软件的变更。

当然还有许多其他的软件开发的最佳实践，这些实践将在后续各章中介绍。

1.4 软件的迭代开发

经典的软件开发过程遵循瀑布式生命周期，如图1-1所示。在这种方法中，开发过程按照需求分析、设计、编码和单元测试、子系统测试和系统测试这一过程线性地运行。



6

图1-1 瀑布式生命周期

这种方法的本质问题是它推迟了解决风险的时间，因此在后面阶段清除早期阶段产生的错误要付出巨大的代价。一个最初的设计可能并没有注意到系统的关键需求，而在后期发现设计中的缺陷可能会导致付出巨额的代价，甚至导致项目的失败。正如Tom Gilb所说的那样："如果你不积极地解决你项目中存在的风险，你就会深受其害。"[⊖] 瀑布方法可能会掩盖项目中存在的真正的风险，而当你太晚发现它时已经于事无补了。

相对于瀑布方法，另一个可选的方法是迭代和增量的过程，如图1-2所示。这种方法建立在Barry Boehm的螺旋模型的基础之上[⊜]。它会在生命周期的早期强制性地确定项目中存在的风险，这时我们可以及时、高效地解决这些风险。这种方法是一个持续的发现、创造和实现的过程，每一个这样的迭代过程都会促使开发小组以一种可预测的和循环的方式来完善项目制品。

迭代的软件开发可以提供一系列的解决软件开发根本问题的方案：

1）可以在生命周期早期发现严重的需求理解错误，这时还可以修正这些错误。

2）这种方法允许并鼓励用户反馈信息，从而抽取出系统的真正需求。

3）这种方法使开发团队将注意力集中到项目中最关键的问题上，并屏蔽掉那些分散他们对项目中真正风险的注意力的问题。

⊖ Tom Gilb, *Principles of Software Engineering Management*. Harlow, UK: Addison-Wesley, 1988, 第73页。

⊜ Barry W.Boehm, "A Spiral Model of Software Development and Enhancement," *IEEE Computer*, 1988年5月，第61~72页。

4）持续的、迭代的测试可以为项目状况给出客观评估。

5）需求、设计和实现中的不一致能够在早期被发现。

6）在整个项目的生命周期中可以更加平均地分配整个团队，尤其是可以平均分配测试团队的工作量。

7）团队可以在过程中总结经验教训，不断地改善开发过程。

8）在整个生命周期中，项目相关人员可以通过具体证据来了解项目情况。

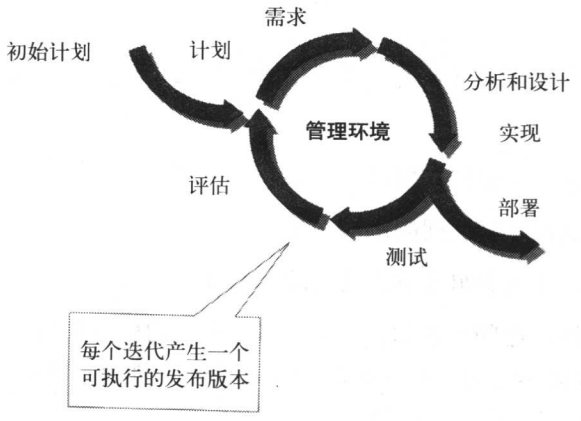


图1-2　一个迭代和增量过程

1.5　管理需求

软件密集型系统的需求管理的挑战在于它们的动态性：你务必要意识到需求在整个软件项目的生命周期中会发生变化。另外，明确一个系统的真正需求（这对于系统的经济和技术目标来说是最重要的）是一个持续的过程。除了一些非常小的系统之外，开发人员在开发之前完全不可能详尽地描述系统的需求。实际上，当一个新的或者进化的系统改变时，用户对系统需求的理解也会改变。

需求（requirement）是一个系统必须具备的条件或性能。动态管理需求包括三项活动：抽取、组织系统所需要的功能和约束并将其文档化；估计需求的变化并评估这些变化所带来的影响；跟踪并记录所做出的权衡和决定。

管理项目需求可以提供一系列的解决软件开发根本问题的方案：

1）在需求管理中构造原则性方法。

2）人员之间的交流建立在已定义的需求之上。

3）区分需求优先级，并进行过滤与跟踪。

4）可以对功能和性能做出客观的评估。

5）不一致性会尽早检测出来。

6）借助适当的工具支持，使用与外部文档的自动链接，可以为系统的需求、属性和轨迹提供库支持。