管理智库丛书

Model-Based Systems Engineering
Best Practices

基于模型的系统工程
最佳实践

(德)汉斯-彼得·霍夫曼（Hans-Peter Hoffmann）◎著
谷　炼

航空工业出版社

管理智库丛书

基于模型的系统工程最佳实践

Model-Based Systems Engineering Best Practices

（德）汉斯-彼得·霍夫曼（Hans-Peter Hoffmann）著
谷　炼

航空工业出版社
北京

内 容 提 要

本书从方法论的角度，描述了基于模型的系统工程最佳实践。主要从系统工程的视点出发，把系统开发的前期系统工程的工作任务、责任范围，以工作流的方式，解剖得淋漓尽致，为系统的后续开发和系统的确认与验证，提供了无缝衔接。本书以系统工程实践者为对象，通过众多截屏、注释和最佳实践技巧，帮助读者清晰理解工作流的细节。本书的目的是帮助读者在集成系统和软件开发中应用基于模型的系统工程标准建模语言 SysML。

图书在版编目（CIP）数据

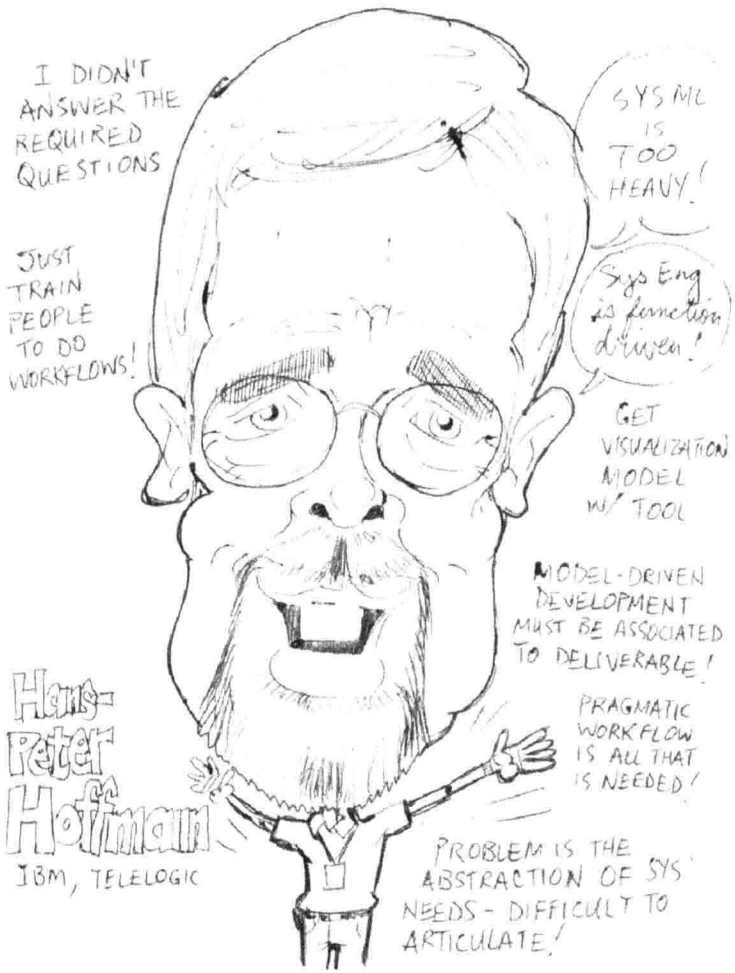
基于模型的系统工程最佳实践 /（德）霍夫曼，谷炼著. --北京 ：航空工业出版社，2014.5
（管理智库丛书）
ISBN 978-7-5165-0222-8

Ⅰ. ①基… Ⅱ. ①霍… ②谷… Ⅲ. ①系统工程—研究 Ⅳ. ①N945

中国版本图书馆 CIP 数据核字（2013）第 154803 号

基于模型的系统工程最佳实践
Jiyu Moxing De Xitong Gongcheng Zuijia Shijian

航空工业出版社出版发行
（北京市朝阳区北苑 2 号院 100012）
发行部电话：010-84934379 010-84936343
北京京华虎彩印刷有限公司印刷 全国各地新华书店经售
2014 年 5 月第 1 版 2014 年 5 月第 1 次印刷
开本：787×1092 1/16 印张：9.75 字数：245 千字
印数：1—700 定价：130.00 元



绘画：瑞克·怀特（J. Rick White）

写在出版之际（代序）

“可以了，我们可以出中文版的了。”这是我们合作完稿之时，我通报给彼得的一句话。彼得说：“当我们一起决定将此书改写成中文之时，在我心里，这本书就已经在中国出版了！”

2013 年 9 月，彼得在中国各行业就系统工程进行巡回演讲时，道出了希望将他的《基于模型的系统工程最佳实践》一书书稿以中文的方式在中国出版的心愿。他终于如愿以偿。

彼得·霍夫曼博士（Dr. peter Hoffmann）是 IBM Rational 系统工程首席架构师，也是 IBM Rational 系统工程最佳实践 Harmony SE 的作者和方法论创始人，是全球著名的系统工程专家。彼得有着 30 多年系统工程领域的指导和实践经验。《基于模型的系统工程最佳实践》集方法论和真实项目的指导经验为一体，是他的心血，是他多年来经验的沉淀。本书除了用通俗易懂的建模方法描述什么是基于模型的系统工程最佳实践之外，还以真实的项目为背景，详细举例说明了如何在具体的系统工程项目实践中加以应用。

关于基于模型的系统工程标准建模语言 SysML，已经出版过许多书籍，也发表过很多文章。但都很少提及问题的关键：如何帮助读者清楚明白地把其应用到一个集成的系统和软件开发中去。

撰写本书的目的就是要填补这一空白。

彼得曾多次强调，他有一种帮助和支持中国客户在系统工程领域赶超世界水平的使命感。希望该书的出版会对中国的系统工程发展具有指导意义，也希望读者在阅读此书时，能体会到彼得的这种使命感。

本书适用的对象为：航天航空、机动车研发和生产、铁路工程、造船工程、电子电动工程、核电以及能源工程等行业和领域的系统总设计师、系统工程师、系统需求分析与设计人员、系统测试工程师、项目管理人员和质量控制管理人员。

需要强调的是，我们不能忘记在此书的撰写过程中，得到了IBM专家、“Wizard大师”Andy Lapping（IBM Rational Rhapsody系统工程套件的作者），IBM专家Pavel Vodov（IBM Rational Rhapsody和DOORS集成场景的设计者），IBM Rational中国团队，中国航空工业集团公司、中国商用飞机有限责任公司、中国船舶工业集团公司等的相关领导和团队的大力支持和帮助，在此一并致以衷心的谢意。

谷炼

2013年6月18日

目　录

第1章 绪 论

1.1 范围

关于基于模型的系统工程标准化建模语言 SysML，关键的问题是如何帮助读者清楚明白地把其应用到一个集成的系统和软件开发中去。到目前为止，很少有人认真思考并着手解决这个问题。撰写本书的目的就是要填补这一空白。

IBM Rational 集成系统 / 嵌入式开发流程 Harmony™独立于工具之外，它应用 SysML，为系统工程师提供了细致的、按部就班的流程指南。它从系统工程的视点出发，把系统开发前期的工作任务、责任范围，以工作流的方式，解剖得淋漓尽致，为系统的后续开发和系统的确认与验证，提供了无缝衔接。

本书中所展示的各种模型，是使用 IBM Rational Rhapsody® 建模工具完成的。本书中所展示的需求管理工具画面，是使用 IBM Rational DOORS ® 展现出来的。

本书是以系统工程实践者为对象进行编写，通过众多截屏、注释和最佳实践技巧，帮助读者清晰理解工作流的细节。本书在叙述上力求突出重点、短小精悍、清晰易懂。

本书以方法论的角度描述基于模型的系统工程最佳实践，但它替代不了 IBM Rational Rhapsody（简称 Rhapsody）培训教材。本书在一些章节上，对读者的背景作了一些假设，如熟悉 UML / SysML 语言，并懂得使用建模工具 Rhapsody 等。

1.2 内容概述

本书由 5 个章节构成。

第 1 章，描述本书的结构和范围。

第 2 章，介绍 Harmony 系统工程的基本概念。着重强调系统开发最关键的系统工程阶段的工作方法和责任，并对工作流以及相关的工作产品在不同的系统工程阶段所起的作用进行详细阐述。此外，还针对在系统工程建模中所必需的 SysML 工件和要素，按照服务请求驱动建模方法进行了概要介绍。

第 3 章，描述当 Rhaospdy 应用在一个基于模型的系统工程项目时，其项目结构是如何构成的。

第 4 章，使用 Rhapsody 工具，详细描述 Harmony SE（Harmony 系统工程）工作流的案例。案例所描述的工作流起始于涉众需求导入到 Rhapsody 中，结束于可执行的架构模型的定义完成。该工作流是面向应用的、易于理解的简单案例，聚焦于帮助读者如何更好地使用基于模型的系统工程方法，如何在实践中应用 Rhapsody SE-Toolkit（Rhapsody 系统工具箱）。

第 5 章，描述如何解决交付到后续的子系统开发的问题。

本书最后还提供了一些附录供大家参考和使用：

- 在基于模型的系统工程中使用 SysML 各种图建模的一些指南；
- 从活动图和相关序列图捕捉的信息如何导出状态图的指南；
- Rhapsody 行动语言快速参考指南；
- Rhapsody SE-Toolkit 功能应用概述。

第 2 章　Harmony SE 基础

2.1　Rational 集成系统／嵌入式实时开发流程：Harmony

图 2－1 用传统的系统工程 V 模型展示了 Rational 集成系统／嵌入式实时开发流程 Harmony。V 模型的左翼描述的是自顶向下的设计流，V 模型的右翼展示了自底向上，从单元测试到最终系统验收测试的集成各阶段。使用状态图的符号，便可显示出工作流中的一个变更请求从高层次向下的影响。无论变更请求在哪里出现，流程都将重新从需求分析阶段开始。

Harmony 流程包含两个紧密联系在一起的子流程：

- Harmony SE；
- Harmony 嵌入式实时开发

Harmony SE 的系统工程工作流是增量迭代式的周期活动流。它由需求分析、系统功能分析、系统设计综合 3 个阶段构成。增量迭代是基于用例来完成的。

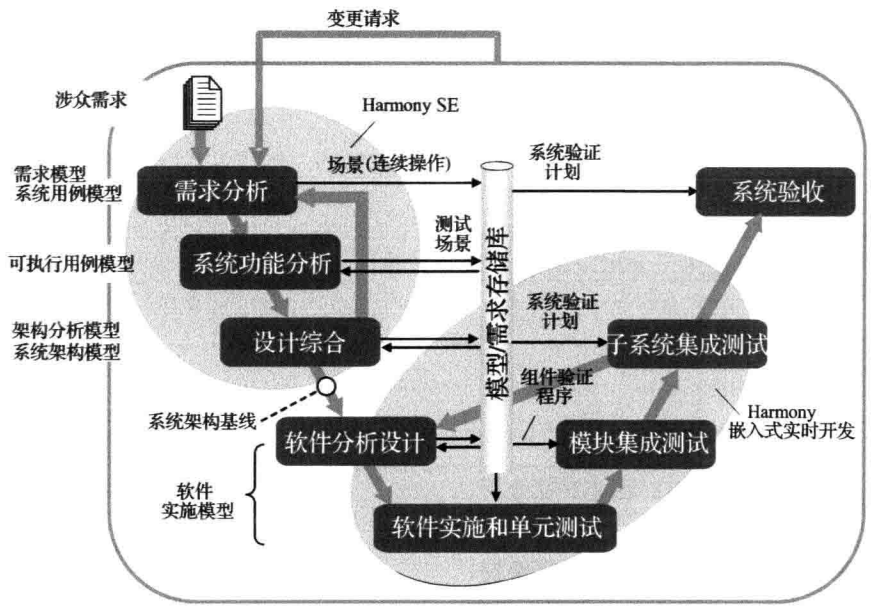


图 2－1　Rational 集成系统／嵌入式实时开发流程 Harmony

软件工程工作流的特点是软件分析、设计、不同层次的集成和测试的迭代式增量生命周期。

值得强调的是，对于系统工程及其实施的分析迭代，是通过不断地实施、测试，并不断提供出可演示的结果进行迭代的。

在自顶向下的设计中，需求的建立和与测试相关的测试场景的重用是非常重要的。这些测试场景同时也支持着自底向上的集成过程，以及当系统变更时的回归测试周期。

Harmony 是 IBM Rational 在系统工程行业的最佳实践。它支持模型驱动开发的流程。在模型驱动开发中，模型是开发流程的核心成果，与分析和设计共存。每一个开发阶段由特定类型的模型支持。

支持需求分析阶段的模型是：

- 需求模型；
- 系统用例模型。

需求模型使得需求类型可视化。系统用例模型把需求以用例方式组织在一起，但这些模型是不可执行的模型。

系统功能分析阶段关注在把功能需求解释成与系统功能（操作）一致性的描述。每一个用例被解析成可执行的模型，并通过模型执行来验证相关的系统需求。

有两种可执行的模型支持设计综合阶段：

- 架构分析模型；
- 系统架构模型。

架构分析模型的目的与权衡分析模型相关。如：通过参数分析，针对所确定的操作实施方式，详细阐述系统架构概念。

系统架构模型，是对前阶段的架构分析阶段所形成的系统架构，进行系统操作分配。系统架构模型的正确性和完整性是由模型执行来验证的。一旦模型被验证，架构设计则可以进入性能和安全性需求分析。该分析包括失效模式和效果分析（FMEA）、任务危害性分析。

打上基线后的系统架构模型，用于定义后续硬件 / 软件开发的交付物。

模型驱动软件开发由软件实施模型支持。该模型是手动和自动生成代码的基础。

模型驱动开发流程中的必要元素是模型 / 需求存储库。它包含开发中系统的知识配置信息。如：

- 需求文档；
- 需求跟踪性信息；
- 设计文档；
- 测试定义。

2.2　基于模型的系统工程流程

Harmony 系统工程的主要目标是：

- 识别并推导所需的系统功能；
- 识别相关的系统模式和状态；
- 把系统功能和模式 / 状态分配到子系统结构中。

对于建模，这些目标意味着自顶向下的高层次抽象。主要强调的是需要的功能和基于状态的行为的确定和分配，而不是其功能行为的细节。

图 2-2 描绘了 Harmony 系统工程的概述，对系统工程的每一个阶段，显示了基本的输入和输出。

接下来的各个段落会详细描述工作流和基于模型的系统工程流程所产生的工件，并勾画出与需求管理和可追溯性（RT）相关的概念。有关更多面向应用的工作流的描述，详见第 4 章。

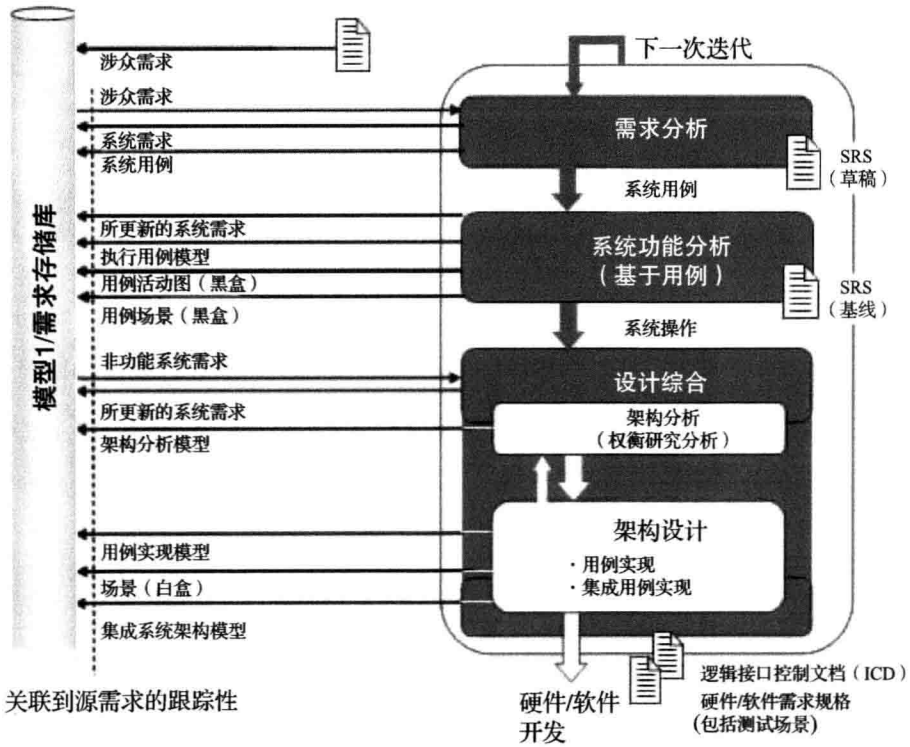


图 2-2　基于模型的系统工程

2.2.1 需求分析

需求分析阶段的目的是分析流程的输入：涉众需求。需求分析阶段之后，涉众需求翻译成系统需求。系统需求定义系统必须做什么（功能需求）以及如何执行好（服务需求的质量）。

需求分析工作流的基本步骤如图 2－3 所。它起始于对涉众需求的分析和完善。这个阶段的输出是涉众需求规格说明书。基本上，涉众需求聚焦于所需要的能力上。下一个步骤是将这些需求转变成系统所需要的功能需求（用“应该”作为对需求的陈述），并建立系统需求规格说明书文档初稿。至于跟踪性，应该将确定的系统需求关联到相关的涉众需求上。

需求分析中的下一个主要步骤是定义系统用例。用例是从一个特定的操作方面（操作线程）描述系统。它详细描述角色（用户）的行为和角色与用例之间的信息流。角色可以是人，也可以是系统，或者是开发中的系统（SuD）外部的一个硬件。用例是不会揭示和暗示系统的内部结构的（黑盒视点）。

用例可分层结构化，但不是按功能去分解用例。用例不是功能，但它们使用功能。关于一个系统需要由多少个用例来描述，没有一成不变的规律。经验显示：对一个大系统，在顶级层次通常需要 10～15 个系统用例定义；在最底层级，描述一个用例，至少需要 5 个，但最多不超过 25 个用例场景。在该阶段，重点是放在用例的“正常情况”（假定无出错或异常系统行为）的识别上。异常或例外场景的描述，将放在下一个阶段（即系统功能分析阶段），通过模型执行的方式进行识别。如果在一个用例中有多于 5 个出错或异常的场景出现，它们应被组织到一个单独的例外用例中，并以包含和排除的依赖关系，关联到“正常情况”的用例上。

为保证所有的功能性需求和相关的性能性需求被这些用例覆盖，需要分别对应建立起跟踪性来。

一旦系统级的用例被定义，并保证了功能性需求和性能性需求能完整地覆盖到系统用例上，就需要根据系统架构定义，对重要性进行分类排序。一组被选用例定义系统工程工作流迭代的增量。每次增量结束时，分类排序需要更新。

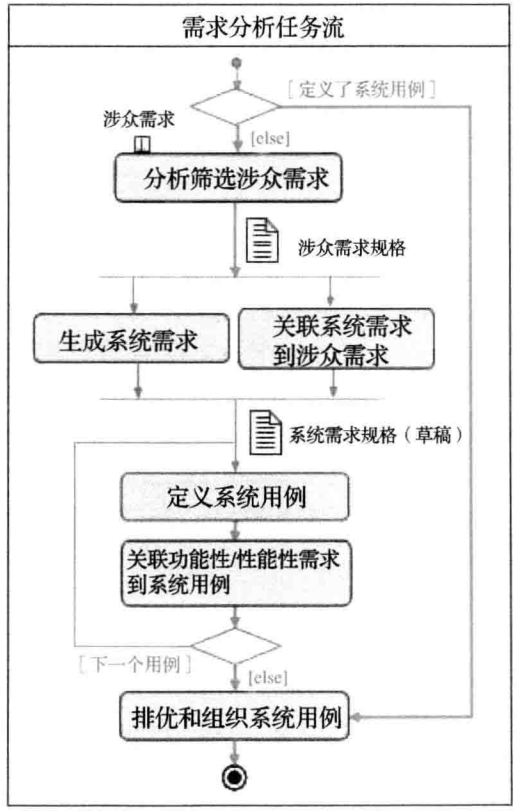


图 2－3 需求分析阶段工作流

2.2.2　系统功能分析

系统功能分析阶段的主要重点，是把系统功能性需求转化为一个连贯的系统功能描述（操作）。分析是基于用例进行的，即：把每一个在前面需求分析阶段确认的用例翻译成一个可执行模型。该模型和相关需求由模型的执行来验证。

图 2－4 详细介绍了在系统功能分析阶段的建模工作流及相关工件。首先，用例模型的上下文是在内部模块图中定义的。该图中的元素是 SysML 模块的实例，它代表用例和与其相关角色。在这一阶段它们是空的而且是非关联的。

建模工作流的下一个步骤是定义用例模块的行为，由 3 个 SysML 图来展现：

- 活动图；
- 序列图；
- 状态图。

每一个图在用例行为的详细描述中扮演着一个特定的角色。

活动图：活动图称为黑盒用例活动图，描述用例整体功能流（故事板）。它以活动（在 Harmony SE 中等同于操作）的方式来组织功能需求，并显示这些活动是如何相互关联的。

序列图：序列图被称为黑盒用例序列图，描述用例通过的一个特定路径并定义操作和角色之间的互动（信息或消息）。

状态图：状态图把活动图（功能流）和序列图（角色之间的互动）的信息会聚在了一起。在状态图中，可以展现融合系统状态的背景，并增添了不同优先级的外界影响对系统行为的描述。

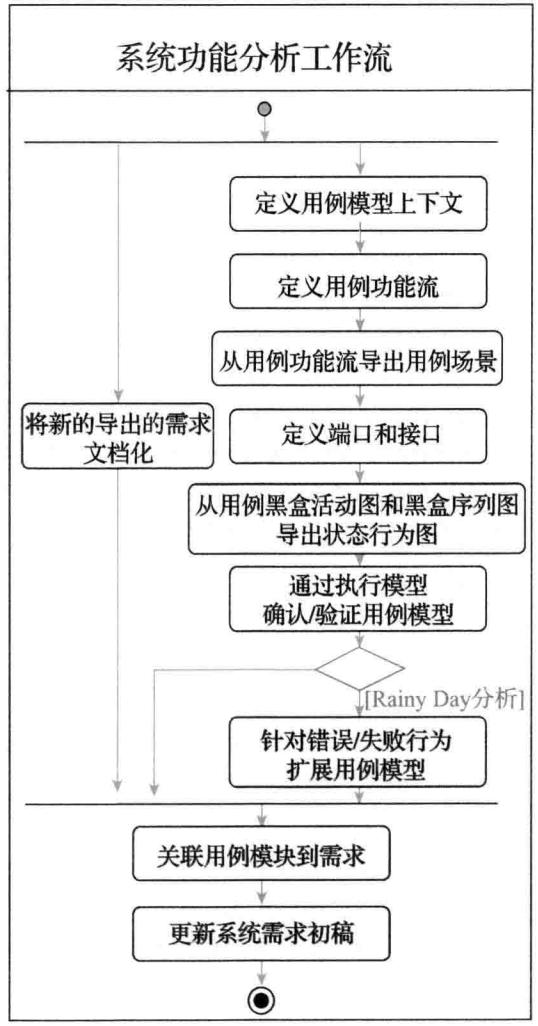


图 2－4　功能分析阶段工作流

不强行规定生成这些图的顺序，因为生成图的顺序也可以依赖于可获得的信息和建模者的偏好。图 2－5 中给出了 3 种可选方法。

可选方法 1 始于对用例场景的定义。客户的做法通常是用序列的方式描述所需的系统（如操作的概念）。一旦一系列的基本场景被捕捉出来，所识别出的功能流就被合并到活动图的通用描述中。端口和接口是由序列图创建的（参考章节 2.4），它们在用例模型的内部模块图中定义角色和用例模块的关联关系。该方法的最后步骤是在状态图中定义用例模块基于状态的行为 。

可选方法 2 始于对用例功能流的定义。如果系统工程师需要详细阐述需求，这是一种通用的方法。通常情况下，客户喜欢从“整体”的视点来表达他们的需求。一旦整体功能流定义了，使用活动图就能导出用例场景（参考图 2－6）。用例模块的端口和接口由序列图生成。最后，用状态图捕捉其基于状态的行为。

可选方法 3 始于对用例基于状态行为的定义。如果设计中的系统强烈基于状态，推荐使用该方法。这种情况下，用例黑盒活动图的创建甚至可以省略，用例场景可由状态图的路径导，从序列图生成端口和相应的接口。

值得注意的是，不管选择哪种方法，在系统功能分析流程中，最重要的图是用例模块状态图，它包含了黑盒序列图和用例黑盒活动图的信息，并可通过模型执行来确认。用例黑盒活动图和相关的黑盒序列图还可以被重用到后续的设计流程中。

在基于用例系统功能分析过程中，无论何时，新需求被识别或用导出的需求细化高层次需求，都需要文档化。最后，在系统功能分析阶段的终期，这些附加的需求需要由利益相关者审批并导出到需求跟踪工具。

用例模型的分析，是通过使用黑盒用例场景作为模型执行的各自基础进行的。应该指出，按照前面所描述的该流程的主要目的，主要关注点是对产生序列的验证而不是相关功能的确认。

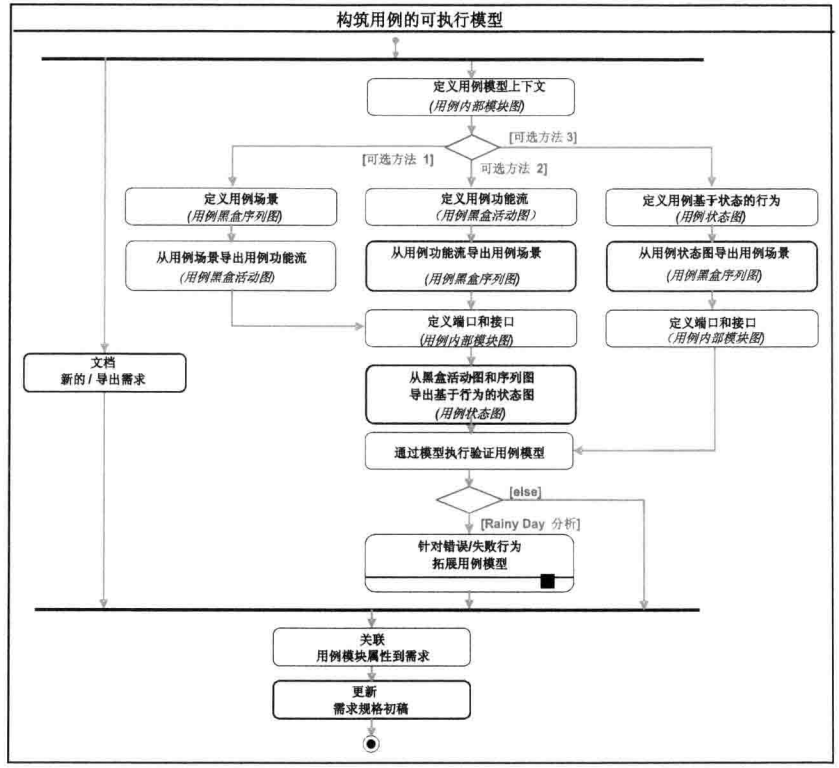


图 2－5　构筑可执行用例模型的可选方法

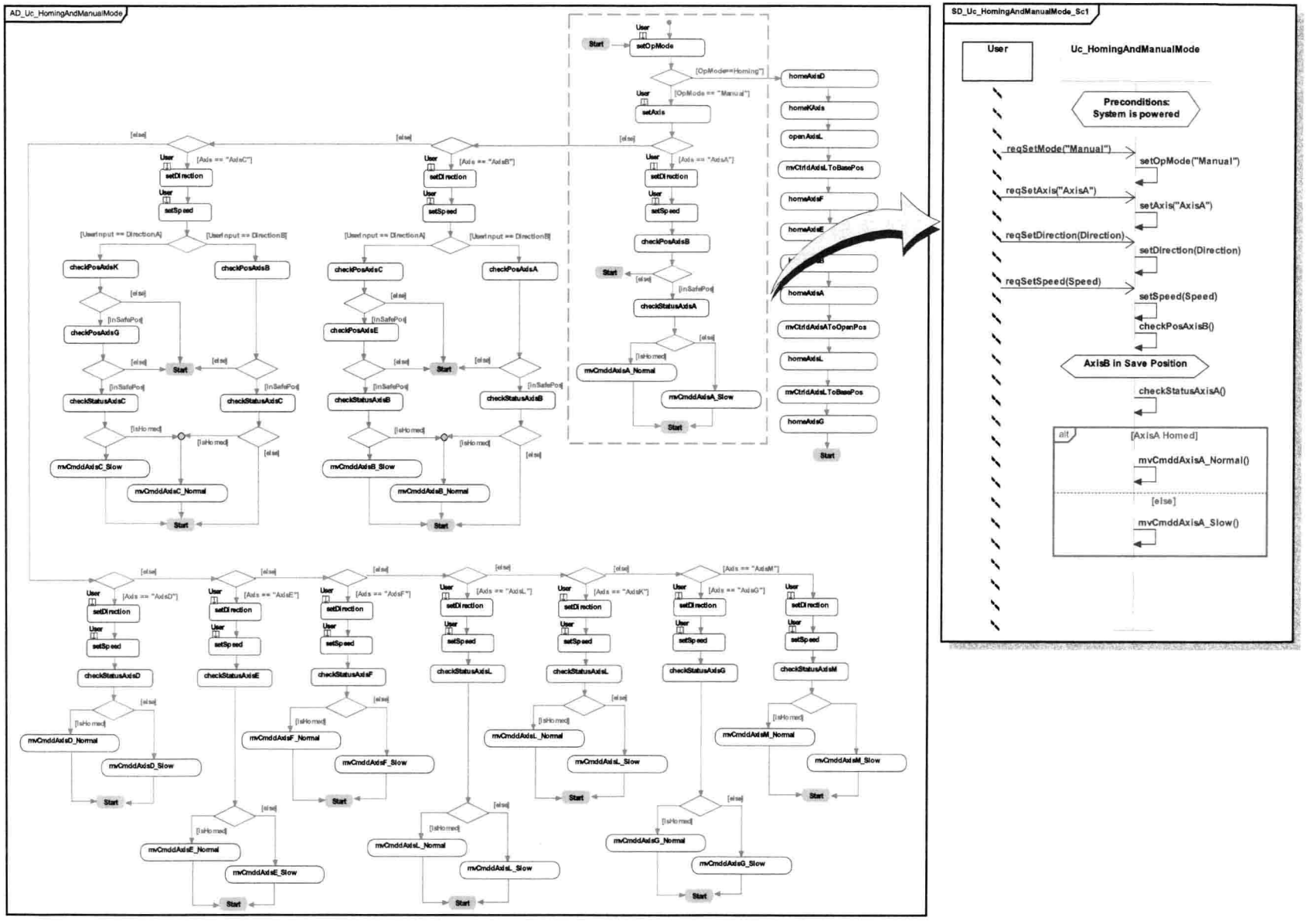


图 2-6　来自一个用例黑盒活动图（工业自动化用例）用例场景的导出（框架示意图）

一旦用例模型和相关的功能性需求被验证，异常情况分析就可以进行了。该分析关注识别系统错误 / 失败行为，这是在初始需求系列中没有覆盖到的。

图 2-7 详细描述了工作流和异常情况分析相关的工作产品。它首先推荐添加个别例外行为到状态图中，因为状态图最佳地展现了整体系统行为。如果错误 / 失败行为包括了新功能，用例黑盒活动图、用例错误行为场景以及用例内部模块图（如果需要），需要进行相应的更新。拓展后的用例模型通过模型执行得以验证。

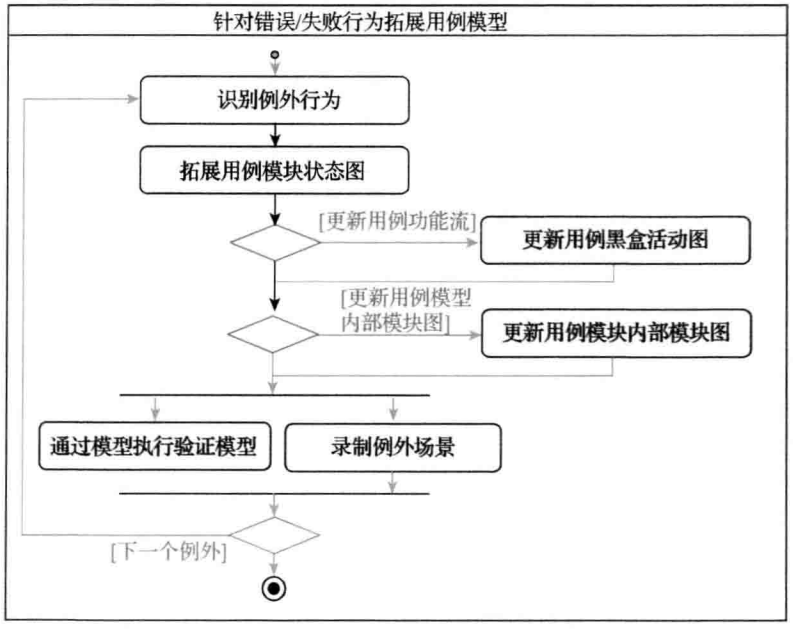


图 2-7　用例模型异常情况分析的工作流

用例建模工作流结束于用例黑盒属性和相关系统需求的跟踪关联的定义。如果新需求或导出需求在建模流程中被识别出来，系统需求规格初稿需要作相应更新。

一旦所有用例的一次迭代被验证，系统功能分析阶段就终止于对系统需求规格基线的生成。另一个在该阶段生成的文档是系统级接口控制文档（ICD）。它定义（黑盒）系统和其角色间的逻辑（功能）接口，也是所有用例模块接口的集合。这个 ICD 是后续系统级（黑盒）测试定义的基础。

有时有人会问：为保证系统完整地被用例描述出来，黑盒的功能系统模型是否应该包括集成的黑盒状态图？原则上说，没有理由不这么做。但注重实效和省时的方法，是将其移到后续的设计综合阶段中去完善。用例应该把足够的系统信息带到架构设计中，然后再开始架构设计。丢失的信息在通过模型执行对系统架构模型的验证时被识别出来。