

SysML精粹

SysML Distilled *A Brief Guide to the
Systems Modeling Language*

(美) Lenny Delligatti 著 侯伯薇 朱艳兰 译



机械工业出版社
China Machine Press

本书仅提供部分阅读，如需完整版，请联系QQ: 461573687

提供各种书籍pdf下载，如有需要，请联系 QQ: 461573687

PDF制作说明：

本人可以提供各种PDF电子书资料，计算机类，文学，艺术，设计，医学，理学，经济，金融，等等。质量都很清晰，而且每本100%都带书签和目录，方便读者阅读观看，只要您提供给我书的相关信息，一般我都能找到，如果您有需求，请联系我 QQ: 461573687, 或者 QQ: 2404062482。

本人已经帮助了上万人找到了他们需要的PDF，其实网上有很多PDF,大家如果在网上不到的话，可以联系我QQ。因PDF电子书都有版权，请不要随意传播，最近pdf也越来越难做了，希望大家尊重下个人劳动，谢谢！

备用QQ:2404062482

软件工程技术丛书

SysML精粹

SysML Distilled *A Brief Guide to the
Systems Modeling Language*

(美) Lenny Delligatti 著 侯伯薇 朱艳兰 译



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

SysML 精粹 / (美) 德里吉提 (Delligatti, L.) 著; 侯伯薇, 朱艳兰译. —北京: 机械工业出版社, 2014.11

(软件工程技术丛书)

书名原文: SysML Distilled: A Brief Guide to the Systems Modeling Language

ISBN 978-7-111-48458-5

I. S… II. ①德… ②侯… ③朱… III. VRML 语言 IV. TP312

中国版本图书馆 CIP 数据核字 (2014) 第 258286 号

本书版权登记号: 图字: 01-2014-2862

Authorized translation from the English language edition, entitled *SysML Distilled: A Brief Guide to the Systems Modeling Language*, 9780321927866 by Lenny Delligatti, published by Pearson Education, Inc., Copyright © 2014.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

Chinese simplified language edition published by Pearson Education Asia Ltd., and China Machine Press Copyright © 2015.

本书中文简体字版由 Pearson Education(培生教育出版集团)授权机械工业出版社在中华人民共和国境内(不包括中国台湾地区和中国香港、澳门特别行政区)独家出版发行。未经出版者书面许可,不得以任何方式抄袭、复制或节录本书中的任何部分。

本书封底贴有 Pearson Education(培生教育出版集团)激光防伪标签,无标签者不得销售。

SysML 精粹

出版发行: 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: 谢晓芳

责任校对: 董纪丽

印刷: 北京市荣盛彩色印刷有限公司

版次: 2015 年 1 月第 1 版第 1 次印刷

开本: 186mm × 240mm 1/16

印张: 13.25

书号: ISBN 978-7-111-48458-5

定价: 59.00 元

凡购本书,如有缺页、倒页、脱页,由本社发行部调换

客服热线: (010) 88378991 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzjsj@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光 / 邹晓东

本书赞誉

“Lenny 的这本书没有让我失望，它秉承了 Addison-Wesley 技术出版物的优秀品质。Lenny 所完成的工作非常精湛，他抓住了 OMG SysML 的精髓，那是一种实际可用、基于标准的建模语言，可以帮助系统工程师解决日益增长的系统复杂性的问题。这本书具有很重要的观点，首先介绍了基本的 MBSE 概念，以区别用例和场景之间的细微差别，然后阐明了命名空间以及 SysML 包，甚至还提到了不为多数人所知的 SysML 语义，如令牌流等。”

——Jeff Estefan, NASA 喷射推进实验室首席工程师

“像 SysML 这样的建模语言的强大之处在于，它不仅可以促进系统工程师之间的沟通，还可以促进跨规则以及跨开发生命周期的沟通。很多语言都具有改善沟通的潜力，但由于缺少有效的指导，它们都很难达到那个目的。在本书中，Lenny 把恰到好处的技术和一般通用方法结合，从而让 SysML 实现那种沟通。我在过去 30 多年间曾经参与过很多领域的系统和软件工程，并且还在很多组织和大学中讲授过计算机语言、UML 和 SysML，我发现 Lenny 的书籍是无价之宝。他清晰地展现了那些概念，并提供了有用且注重实效的实例，让你快速入门并能够成为有效的建模者。”

——Thomas W. Fagnoli, 洛克希德马丁公司工程技术领导成员

“这本书对 SysML 提供了非常棒的介绍。Lenny 的说明非常简洁且易于理解；例子非常透彻且有趣。”

——Susanne Sherba, 丹佛大学计算机科学系高级讲师

“Lenny 的这本书既可以作为 SysML 的参考书来查阅，也可以作为一本有趣的书来完整阅读，从而帮助我们学习这种语言。它是一本好书，秉承了著名的《UML 精粹》的风格。”

——Tim Weilkiens, oose 公司 CEO

“这本书比单纯的 PPT 具有更多的信息量，却没有 OMG 规格说明那么枯燥。它为从业的系统工程师提供了学习纯粹的 OMG SysML 所需要的动机、概念和标识法，

从而更流畅地使用图形化语言来编写规格说明，并分析他们复杂的实际系统。”

——Lonnie VanZandt, No Magic 公司首席架构师

“Lenny 的这本书是最适用的一本书；它浓缩了讲授和在业界环境中使用 SysML 的精华经验。作者为这种强大且复杂的建模语言提供了非常清晰且可读的视图，通过易于模仿的实际例子说明了它的用法。尽管最初它的目的是介绍 SysML，但我毫无疑问可以把它作为资深实践者的随身参考。”

——Bran Selic, Malina 软件公司总裁

“SysML 是一种让人望而生畏的建模语言，但在这本书里，Lenny 让它变得非常易于理解，而书中的建议会帮助实践者避免各种各样的陷阱，帮助他们领会并应用 SysML 的核心元素以及精神。如果你计划应用 SysML，那么这本书就是你所需要的！”

——Celso Gonzalez, IBM Rational 团队高级开发人员

“本书对于开始钻研基于模型的系统工程的工程师来说是一本非常棒的书。空间系统示例以简单而有效的方式记录了状况，并说明了概念。”

——Matthew C. Hause, OMG UPDM 小组首席顾问工程师

“从 20 世纪 90 年代开始我就深入参与到 OMG 中，但我的专业需求并没有让我过多接触 SysML 领域。所以我觉得自己应该是 Lenny 的书很好的 beta 测试员。让我高兴的是，通过阅读这本书我学到了很多知识，并且知道你会一样。”

——Doug Tolbert, Unisys 的杰出工程师，OMG 委员会成员，架构委员会总监

“本书对基于模型的系统工程的语言组件提供了清晰且易于理解的描述，同时对于在哪里找到工具和方法组件相关的信息提供了建议。本书处处都能够证明，作者对 SysML 以及它在系统开发过程中的应用都有非常深入的理解。我肯定会在自己讲授的 MBSE 课程中使用它作为教科书。”

——J. D. Baker, OCUP, OCSMP, OMG 架构委员会成员

“本书是一本必备书，很多 SysML 建模者都应该在书架上放一本。Lenny 拥有丰富的经验和大量认证，能够帮助你解决日常的建模问题。这本书并不是一个教程，也不是关于 SysML 所有内容的百科全书。如果你使用 SysML 建模，那么它会成为你的

日常伙伴，那意味着你会经常用到它。我相信你的书很快就会有折角和笔记。”

——Robert Cloutier 博士，斯蒂文斯理工学院

“SysML 现在在很多领域都会得到应用，包括深度空间机器人飞船和深入地下的农业设备。这本书简洁地说明了 SysML，展现的方式既可以让新手快速上手，又可以让老手感觉非常方便。”

——Russell Peak，乔治亚科技空间系统设计实验室，MBSE 分部总监

“本书在系统建模领域是一本写得非常好、内容丰富并且简洁的书籍。简单易懂的说明能够带领新手为相当复杂的系统建立模型，而对 SysML 建模语言最常用部分丰富、有深度的阐述，使其可以适用于大多数系统的描述。它还可以作为手边的参考书。感谢 Lenny 先生给了系统建模领域这样好的一件礼物。”

——Bobbin Teegarden，OntoAge 公司 CTO、
首席架构师，No Magic 公司董事会成员

译者序

之前在对日软件项目开发的时候，项目组里面有各种各样的角色——PG、SE、PL、PM等。其中PG是编写代码的程序员（Programmer），而SE是系统工程师（System Engineer）。在那个时候，对SE这个职位的理解还不是很清晰，似乎他们更多的是在从事沟通和测试的工作。

当看到这本书的时候，我才发现系统工程师并不是那么简单。真正意义上的系统工程师要比当初那个职位的意义丰富很多。最重要的就是，系统工程师要创建并维护系统的模型。而软件项目中的系统局限在软件系统的范围之内，其实在很多领域中都会有系统，因此在更多的领域中都会有系统工程师这个重要的角色存在。

在敏捷还没有像现在这么流行的时候，我们的软件开发过程还是遵循瀑布式的流程，需求分析，概要设计，详细设计，编码，测试，一个一个步骤按部就班地进行，而且每个阶段都会有完备的文档作为产出物，下一个阶段会以上一个阶段的产出物作为输入。然而，相信经历过这类流程的朋友都会有过很惨痛甚至至于不堪回首的体会，那就是在出现需求变更的时候，我们不得不把所有的文档都修改一遍，当然，也包括我们作为最终产出物的代码。所以，那个时候，需求变更真的是所有项目成员的噩梦。

究其原因，就是因为当时对系统的所有记录都是基于文档的，所以每次变更都会导致所有文档的变更，我们不得不深入到每个部分去做修改，而即便如此，也难免会有遗漏和错误的情况发生。

本书介绍了和基于文档的系统工程完全不同的方式——基于模型的系统工程，这对系统工程师来说是一大福音，特别是对于需求变更非常频繁的软件行业的从业者来说更是如此。

书中首先就说明了基于模型的系统工程方法，并与基于文档的方法做了对比，这样就从根本上解决了出现变更时所需要解决的问题。当需求发生变更的时候，我们不再需要把所有文档都翻出来修改一遍，而是只需要修改一下模型，当需要文档的时候，都是从模型直接生成，这样就不需要到处修改了。

除此之外，书中还详细介绍了各种图，让读者可以在读过之后就了解如何创建SysML表述的模型，所有内容都不依赖于特殊的建模工具，所以读者不必担心对自

已选择的工具不适用。此外，图书的网站上还提供了适用于各种工具的示例，结合阅读这本书，读者可以更好地了解 SysML。

SysML 中的各种图和 UML 很类似，而本书的编写风格和经典的《UML 精粹》又非常类似，所以，如果你读过关于 UML 的那本著作，那么阅读这本书就会有驾轻就熟的感觉，由此也看出作者是多么体贴和用心。

总之，对系统工程师们来说，不管是属于软件系统领域还是属于其他领域，本书都非常有价值，我们既可以把它当成教程从头至尾仔细阅读，也可以将其作为手册放在手边随时拿过来作为参考。

愿系统工程师朋友们享受阅读本书以及使用 SysML 的快乐！

侯伯薇

序 一

系统工程这个学科很难讲授。在职业生涯早期，就有人向我强调，我们无法在教室中学习系统工程，而只能通过实践学习。尽管这种说法并没有得到明确验证，但在系统工程实践中，确实有些概念很微妙，而且晦涩难解。

在模型中表达这些概念需要一种合适而健壮的语言，于是我们这群人于 2002 年早期开始专门做这方面的工作，SysML 由此而来。在设计这种语言的时候，我们试图让它足够简洁和直接，特地把这种语言的目的设定为用于系统工程实践。我确信我们创建的语言既灵活又实用。我很高兴它已经成为沟通系统相关想法的领域标准。

然而，就像系统工程实践一样，我们发现 SysML 也很难有效地讲授。系统工程涉及的范围很广泛，尽管 SysML 是一种相对简洁的语言，但同学们还是经常会因为其复杂性而手足无措。学习 SysML 和基于模型的系统工程的资源现在还相对有限，但已经在逐步丰富。某些大学以及教育机构，已经在定期讲授正式的 MBSE 和 SysML 课程，而且现在已经有了至少一本综合性的教材。

但想要偶尔抽时间学习 SysML 基础知识的工程师和经理可能不会去上课。高级系统工程师也会忙于期限紧张的项目，而没有时间去上课。对于上述两种情况，这本书都价值非凡。

这本书和 Martin Fowler 的畅销书《UML 精粹》结构类似，它以清晰、简洁的术语介绍了 SysML 图的基础知识。它以轻量级的非正式方式编写，但传递了每种概念的主旨以及图形化的表示方法。关于这本书我最喜欢的一点就是，它会让我手不释卷，而不会陷入到“元说法”和“UML 主义”中。书中有很多有趣且切实可行的建议。

这并不是一本应用 SysML 或者部署 MBSE 的教材和指导书籍，也不会详细解释每个系统工程概念的方法论原理。尽管它没有在各个章节中使用一系列一致的例子，但也不会仅仅带着读者浏览任何特定的 MBSE 过程。它不是一本工作手册，没有包含问题，也没有让读者解决的示例练习。作为 SysML 的使用者或者高级的 MBSE 实践者，你可能最终需要其他资源，但这本书是个非常不错的开端。

这本书是一本有干货、可以自主学习、轻量级的 SysML 参考指南。世界对其充满期待。

Rick Steiner

《A Practical Guide to SysML》合著者

序 二

采用新技术要花时间

我很幸运能够在 20 世纪 70、80 年代技术及实业领域最好的大学之一读书。正如史蒂夫·乔布斯所说，未来是发明出来的，而不是发现出来的。在那所大学中，黑客们通宵达旦，帮助世界创建了雷达、闪光摄影术以及因特网。那些技术有助于改变世界；更重要的是，人们创建了很多公司和其他组织来使用那些技术，经济也随之繁荣起来。计算爆炸从 20 世纪 60 年代开始，到 1980 年在麻省理工结束。

我在学术领域最早的贡献是在麻省理工的 11 年间做出的，随着我在学术上的兴趣转移和改变，贡献也有了变化，它从人工智能领域开始（最早关注于手写识别技术），在研究生阶段转移到计算系统架构，最终是二者的融合。我在所关注领域做出的贡献，当我还在 MIT 的时候就已经被五家创业公司采用（尽管可能那是在 MIT 期间做出的一个大贡献），但不仅限于那些。人工智能先驱——像 Symbolics 和金山计算机——对我关于应用技术的理解都非常重要；我自己的创业公司，A.I. Architects（其中有我所见过的最好的系统工程师），也非常依赖于一种冲突，那就是对人工智能的需求和早期个人电脑有限的计算能力之间的冲突。

我在这段时间产生的最重要的认识可能是，使技术从实验室进入实际生产的时间花销，要比任何学术领域想象的还要多。20 世纪 80 年代的专家系统，现在是诊断系统以及世界范围内其他系统的主要补充（尽管一般来说要使用“基于规则的系统”的标记），很明显基于 20 世纪 60 年代 PLANNER 和 CONNIVER 之类的系统。20 年看起来像是弹指一挥间；让一种技术经过工程需求的检验，足够稳定，并在工业范围成为一种方法，然后再面向市场生产和整合，都需要花费大量时间。

OMG 敬畏的对象

不管怎样，当对象管理组织（Object Management Group, OMG）在 1989 年创建的时候，人们承诺对象技术和分布式对象将会改变计算领域的面貌。随着因特网慢慢

变成 3 维网，人们开始看清，一致、标准化的中间件对于让集成——不仅仅是世界范围内的文字页面，还包括应用程序的互操作性——更可能成为现实。“混合”全世界计算能力和数据源的能力（正如我们在 20 年后所说的），以及使用标准化 API 和在线协议，要比面向对象的方法更简单。

尽管 OMG 在最初做得不错，没有大肆宣传，避免了 20 世纪 80 年代由于过分宣传 AI 市场而导致的“人工智能的冬天”，但它对于采用技术需要花费的时间似乎认识不够。在“混合”真正成为现实之前可能需要 15 到 20 年，而面向对象语言（最初是 C++，它本身就是从 Simular 发展了 20 年的产物；现在是 Java、C# 和 Ruby-on-Rails）会渗透到计算领域中。像所有技术一样，OMG 敬畏的对象会成为世界各地软件开发所使用的日常工具，但那会花费很多年。

建模造就内行

同时，OMG 发现了另一个机会。在 1996 年有人提出，面向对象分析和设计市场（那时就是这么叫的）已经到了穷途末路，因为没有基于继承技术，基于群集的方法（甚至更糟的标记法）在市场上已经泛滥成灾。即使是喜爱那些方法的技术专家也因为有太多选择而感到手足无措（一旦做出选择，很难保证可移植性和互操作性）。20 世纪 90 年代中期分析和设计市场的巩固，创建了以厂商为中心的市场力量，它们创建了标准，并被慢慢增长的用户社区所接受。在 1997 年它们创建了统一建模语言（Unified Modeling Language, UML）标准，尽管只有共享的标识法，而没有共享的方法，但足以在接下来的 15 年内让市场进入到超过百分百的 CAGR。由于应用程序的生命周期不仅仅是分析和设计，还包括开发、测试、实现和维护，之前针对“分析和设计”的活动称为建模。

即便是使用像 UML 这样的早期标准，在几年内建模的重要性也显现出来了。早期的科学分析显示，使用建模方法（相对于低级的编程语言开发）提升了 35% 的生产力；可能更重要的是，在软件开发生命周期中维护和支持占据了 80% ~ 90% 的工作量，而一些关键的分析显示，那 35% 的生产力提升恰好在维护和整合方面。对这些标准的认可——在写这本书的时候，根据 Gartner & Forester 的市场分析，包括了 71% 以上的软件开发团队——使得在 OMG 出现了与建模相关标准的爆发性增长。

在 OMG UML 标准（及其相关的强大父标准，元对象工具（Meta Object Facility，

MOF)) 出现后的 15 年, 出现了很多标准化的领域特定建模语言。各种功能的语言, 包括定义在芯片上的系统、面向服务架构 (SoaML)、用于业务建模和分析 (BPMN)、记录企业架构 (UPDM)、定义基于规则的系统 (SBVR), 甚至是记录系统开发背后的动机 (BMM), 所有这些都加入了 OMG。更重要的是, OMG 的大多数工作都转移到“垂直市场”, 解决医疗信息技术、财务服务、生命科学、自动化以及其他消费设备依赖性的分析等问题, 所有这些都基于系统的视图, 而系统又基于高层次的模型。

为系统的散播提供服务

OMG 中最重要的一个成员就是系统建模语言 SysML。SysML 被定义为一种特殊的 UML, 作为能够完成多项任务的语言, 它可以整合很多大型系统工程的视图: 不仅仅是软件和硬件, 还包括需求、数学参数、工具管理、维护设计, 甚至包括对人力和其他资源, 以及设计中的系统行为的管理。我在 2001 年提出的愿景叫做模型驱动架构 (model driven architecture), 可能会通过这样的方法实现。这种方法可以整合各种工程, 而不仅仅是“软件架构”, 还包括复杂系统的总体架构, 如载人飞行器和化工厂等。就像 IDEF 系列规范在 20 世纪 80 年代早期承诺的一样, SysML 可以集成在众多需要它的领域中, 以构建设计良好、符合目标且可维护的大型系统。

从最早提出模型驱动架构到现在已经有 12 年, 从统一建模语言的发布到现在也快要 20 年了, 当前一书在手, 就可以整合各位专家关于如何思考并使用 SysML 来交付真正的系统方面的观点。现在我们有了这本书; 它把重要的内容以及最重要的概念都提取出来, 展现给大家。

复杂系统开发本来就是一项团队活动。单独一个人无法管理甚至无法搜集大型系统的需求; 规模本身就这样的系统非常复杂了。由于设计真正关注的是简洁性, 还有一个或多个维度, 我们需要标识法和过程, 不仅要沟通简化的愿景, 还需要让设计师、开发者和工程师深入到系统设计中, 并以不规则的方式探寻底层的设计部分、期望和需求以及整合方法。你要知道, 像 SysML 这样的标识法——大型且复杂, 当然包含各种的不同工具——可以支持大型系统开发。想要有效地使用那些工具, 需要努力地学习。我的岳父熟知如何使用螺丝刀来完成房子周围的手工任务 (包括敲钉子); 我更喜欢使用一些合适的工具, 并了解如何以集成的方式来使用那些工具。此外, SysML 建模语言不仅仅用于实现大型复杂系统, 还用来和系统的用户、维护人

员，以及可能会对系统进行调试和集成扩展、修正、变更的人来沟通设计。

这本书介绍了那些工具集；更好的是，它说明了如何一起使用多种工具，以获取需求、构件设计、分析设计并和其他团队中的其他人（你们可能在将来组成一个团队）沟通这个过程。那是工程师所做的工作，而 SysML 是完成那项工作的最好方式。

Richard Mark Soley 博士

OMG 总裁及首席执行官

前言

为什么要写本书呢？很简单，因为忙碌的你需要了解 SysML。你已经有一些系统建模工作要做。你不需要知道语言的所有细节。你只是想要有本书能够让你专注于 SysML 在日常工作中最常用、最有用的部分。本书就是你想要的那本书。

你可以选择把本书作为必备参考书，当遇到难题并且快到最后期限的时候随手翻阅。或者你可以选择每次深入阅读一章，为即将到来的工作向你的工具箱添加新的建模技能。或者你可以选择逐页阅读，从而为 OMG 认证系统建模专家（OCSMP, OMG Certified Systems Modeling Professional）的前两级认证——OCSMP 模型用户和 OCSMP 模型构建者：基础——做准备。本书的设计可以满足你所有这些方面的要求。

本书读者对象

SysML 是一种图形建模语言，可以使用它来可视化并沟通各种规模的社会技术（sociotechnical）系统的设计——由硬件、软件、数据、人和过程组成的系统。系统工程师会负责对社会技术系统进行规范、分析、设计、验证和检验。因此，系统工程师以及系统工程的学生是本书的目标读者。

但这样说十分笼统。很多作者和老师都说过：一切都是系统。让我来补充一下：一切工程师都是系统工程师。不管你的领域和工作职责是什么，你都可能会执行我提到的某些或者全部系统工程任务。这本书的前提是，相对于通过非标准化的沟通模式，加上一系列文档和图，通过集成的 SysML 模型的标准化媒介，可以更有效地执行这些活动。你是一名系统工程师，你想要更有效地完成工作。因此你也是这本书的目标读者。

在阅读本书之前需要了解什么呢？你应该至少对系统叙述、分析、设计、验证和检验有概念性的了解。提前知道其中包括这些活动，会帮助你内化 SysML 能够帮助你更好完成这些活动的方式。国际系统工程委员会（International Council On Systems Engineering, INCOSE）的《系统工程手册》（Systems Engineering Handbook）是本权威的参考书。

在阅读本书之前，你并不需要在建模语言方面有任何经验。你可能已经知道，SysML 基于统一建模语言（UML）。事实上，你可能已经读过 Martin Fowler 的《UML 精粹》。我把本书设计为系统工程师的必备书，他们需要在软件系统（创建 UML 的原因）之外更广泛的领域进行系统建模。因此，你不需要了解 UML 对作为阅读本书的前提。本书的结构和内容使得阅读它本身就足以学习 SysML 了。

本书结构

本书包含 12 章以及 2 个附录。第 1 章介绍了基于模型的系统工程（Model Based Systems Engineering, MBSE）的概念，并提供了学习 SysML 的情境和业务案例。第 2 章讨论了为什么创建 SysML，并介绍了可以创建的 9 种 SysML 图。第 2 章还提到了能够应用在所有 9 种图上的通用概念。

第 3 ~ 11 章详细说明了每种 SysML 图的细节，介绍了可以在其中显示的元素和关系。尽管能够在这些图上显示的元素和关系种类偶尔有重复，但我还是每次在一章中专注于一个图，以有效地把相关的观点分组，帮助你在需要的时候更容易定位特定主题。第 3 ~ 11 章分别介绍以下 SysML 图。

❑ 模块定义图

❑ 内部模块图

❑ 用例图

❑ 活动图

❑ 序列图

❑ 状态机图

❑ 参数图

❑ 包图

❑ 需求图

最后一章也就是第 12 章，介绍了分配的概念——它是可以横跨所有 9 种 SysML 图关联元素的一种关系。

书中的示例图显示了 DellSat-77 卫星系统的各个方面，这完全是我为了撰写本书假设的一个系统（在此我确保没有泄露任何航空公司的任何专利信息）。我选择用一个卫星系统来演示可以如何使用 SysML 来建立复杂、现实的社会技术系统的模型——而不采用在建模工作坊中常见的典型模型（ATM 和巡航控制系统）。本书通篇

选择使用唯一的系统作为运行的示例，也是为了展示 9 种类型的 SysML 图如何对底层的系统模型展现相互补充且一致的视图。

你可以进入我的网站 www.lennydelligatti.com，在 Articles and Publications 页面下载 DellSat-77 卫星系统的 SysML 模型。我分别以 XML 格式和各种建模工具的本地格式提供数据文件。这些资源让自学者、老师和学生能够选择自己的建模工具，实际操作本书中出现的系统模型。

附录 A 是书中图形表示法的简单总结，标明了对其进行详细讨论的章节。附录 B 介绍了 SysML v1.3 引入的元素种类，那也是在撰写本书时 SysML 的最新版本。

SysML v1.2 是当前 OCSMP 认证考试所采用的版本。SysML v1.2 和 v1.3 之间最大的区别在于端口——一种可以出现在模块定义图（Block Definition Diagram, BDD）和内部模块图（Internal Block Diagram, IBD）中的元素。第 3 章介绍了 BDD，第 4 章介绍了 IBD。在这些章节中，出于以下三种原因我专注于 SysML v1.2 的定义。

- 在 SysML v1.3 发布之前开始的建模项目中（这些项目很多现在仍然在进行中），它们是系统模型中主要的端口类型。
- 某些建模工具还没有跟上 SysML 的改变，还没有实现 SysML v1.3 中对端口的定义。
- OCSMP 认证考试在 SysML v1.3 发布之后还没有修订，仍然采用 SysML v1.2 中对端口的定义。

但是，不需要担心。附录 B 完整介绍了 SysML v1.3 对端口的定义。如果你的建模团队想要创建新的系统模型，那么我建议使用新的端口来替换旧的（假设你的 SysML 建模工具支持）。

章节的顺序大至基于图的一般使用频率。它并不反映每种图的相对价值，当然，它也无法反映。毕竟价值是很主观的东西，你的团队会基于所采用的建模方法以及为客户产出的交付物来决定。

章节的顺序也没有反映——也不应该建议——任何特定的建模方法。毕竟这不是一本方法学方面的书籍，而是一本语言方面的书。第 1 章讨论了建模方法和建模语言之间的区别。我列举了几种众所周知的建模方法，并给出了一些全面讨论它们的参考书。

本书的目标是，以简洁、专注目标的方式向你展示 SysML 最常用、最有用的特

性——不管采用什么建模方法都有用的特性。关键的一点是，SysML 只是一种语言，它与方法无关。我把本书也设计为与方法无关。我希望你能够知道，SysML 是用于沟通的具有附加值的媒介，不管你的团队为了完成工作、满足利益相关者的需求采用的是什么过程、流程和工具。

我希望你觉得这本书在学习 SysML 的过程中有用。它是一种丰富、利于表达的语言——具有足够的广度和深度，可以让你可视化并沟通系统设计的所有方面。要了解的知识还有许多，但你并不需要知道所有内容就可以创建能够清晰沟通的有效系统模型。阅读本书并获得你所需要的内容。你会发现如何快速把那些知识应用到工作中，并向你的客户交付价值。

致谢

很多聪明和敬业的人都为本书的出版付出了很多。我首先要特别感谢 Jim Thompson，他是我的朋友、同事、精神导师以及每周寿司学习班的搭档。在我撰写初稿的时候，他花了几个月来帮我审校，并且提供了重要、精辟的反馈。本书很大程度受益于他敏锐的技术头脑以及优秀的沟通技巧。

我要特别感谢 Addison-Wesley 出版社的 Chris Guzikowski。他带领我完成这个项目，让我这个菜鸟作者度过了最初的学习阶段。我特别要感谢他提出的宝贵建议：要不断精简，从而让我最终能够完成这本书。

Chris Zahn 是 Addison-Wesley 出版社的策划编辑，Betsy Hardinger 是一位优秀的文字编辑，他们为我提供了所需的特别支持，让这本书最终出版成册。他们教我好的想法转换成好书以及好的稿件的艺术。这本书的因为他们的贡献而质量优越。

Elizabeth Ryan 是 Addison-Wesley 出版社的项目编辑，她和产品团队协作，设计了这本书的版式，并把各个部分集中起来以便印刷。他们把复杂的过程变得简单，并创建了最终的产品。非常感谢他们辛苦的工作。

我还要深深感激由工程师和系统建模者组成的外部团队，他们是这本书的技术审校者：Celso Gonzalez、Robert Cloutier、Susanne Sherba、John Pantone、Michael Engle 和 Michael Chonoles。他们的专业能力和洞察力让我把粗糙的初稿，修改成更加专注的最终版本——它可以更好地为系统工程师社区服务。非常感谢他们。

我还要特别感谢下面这些人：Jeff Estefan、Susanne Sherba、Lonnie VanZandt、Bran Selic、J. D. Baker、Tim Weikiens、Tom Fagnoli、Robert Cloutier、Matthew

Hause、Russell Peak、Doug Tobert、Celso Gonzalez 和 Bobbin Teegarden，他们审校了修订的原稿，并提供了支持。他们中有建模语言开发者、建模认证开发者、系统架构师、教师和领域内的专家，他们对系统建模社区做出了特别突出的贡献。我非常感激他们的参与。

特别感谢 Rick Steiner 为本书撰写了序言。Rick 是 SysML 的最初创建者之一，并且一直服务于 SysML 修订任务团队（Revision Task Force, RTF）——这个团队会定期根据系统建模社区的反馈对 SysML 规格做出改进。由于他的经验和贡献，我们的专业变得越来越完善。我要对他所做的一切表示感谢。

我还要特别感谢 Richard Soley 为本书撰写了序言。Richard 从 1989 年就开始领导对象管理组织（Object Management Group, OMG）。OMG 在工程领域留下的功绩不可尽数。作为 OMG 的一员，Richard 和作为 OMG 工作组志愿者的工程师专家已经改变了我们做工程的方式。基于模型的工程图创建并深入我们的工作之中，已经为我们这些喜欢和实践工程学的人，以及我们的客户提供了更多财富。我要感谢在社区持续改变这个领域的过程中，Richard 所提供的愿景和领导。

在十多年中，很多既聪明又有经验的工程师对 SysML 的发展都做出了很大贡献。我无法逐一说出他们的名字，只能在此对他们的努力一并表示感谢。正是因为他们的辛勤工作，我们才有了 SysML 这种丰富的媒介，可以彼此沟通我们的系统设计。我努力让这本书成为 SysML 的代表。

我还要感谢我的妻子 Natalie 以及孩子 Noelle 和 Aidan。两年间，当我每晚和周末在计算机上花费大量时间的时候，他们表现出非常大的耐心和理解。他们是我生活的动力，我非常感谢他们的爱和支持。

Lenny Delligatti

得克萨斯州 休斯敦

2013 年 10 月

作者简介

Lenny Delligatti 在卡耐基梅隆大学获得电子和计算机工程学的学士学位，在丹佛大学获得了计算机科学系统专业的硕士学位。他拥有 OMG 的认证系统建模专家 (OCSMP) 高级模型构建者证书。这是在 SysML 和基于模型的系统工程 (MBSE) 方法学方面的最高级别认证。此外，他还持有 OMG 认证 UML 高级专家 (OMG Certified UML Professional, OCUP) 证书，那是 UML 领域的最高级别认证。

Lenny 是一位高级系统工程师，和 Lockheed Martin 一起创建了 SysML 模型，并在约翰逊航天中心的 NASA 任务控制中心：21 世纪 (MCC-21) 项目中担任 MBSE 主管。他之前曾经在 NASA 的飞行器模拟项目 (Aircraft Simulation Program, ASP) 中担任嵌入式软件工程师，为 NASA 构建了 VxWorks 内核，并为 Gulfstream II 空间火箭模拟器编写了飞行软件。他还在卡耐基梅隆大学空间机器人中心的 Nomad 项目中担任软件工程师，为 Nomad 自动漫游者设计和开发传感器管理子系统。

Lenny 是对象管理组织 (OMG) SysML 修订任务团队 (RTF) 以及 OCUP2 认证开发团队的成员。他还是国际系统工程委员会 (INCOSE) 得克萨斯州海湾分部 (TGCC) 的教育和外联总监，为休斯敦地区的系统工程社区提供专业发展支持。

除了他的工程经验之外，**Lenny** 还曾担任过美国海军的水面作战指挥官，完成过对任务支持的部署，出过两次任务，分别位于日本的佐世保和弗吉尼亚的诺福克。在海军服过兵役之后，他在 Old Dominion 大学接受了教育学方面的正式培训，并获得了弗吉尼亚州的数学教师资格证书。在转回工程领域并搬到得克萨斯的休斯敦之前，他在 Fairfax 公立学校系统中担任过数学教师和系主任。

Lenny 对工程学充满热情，乐于帮助工程师开发出更有效的方式来完成工程。他曾为数百位系统和软件工程师创建并提供了上百小时的课程，内容主题包括 UML、SysML、MBSE，帮助很多人获得了 OMG 认证，并教会他们在项目中使用 MBSE。他还曾在 INCOSE 会议以及约翰逊航天中心的美国航天航空公司 (AIAA) 技术大会上发表 SysML 和 MBSE 方面的演讲。

推荐阅读



人件 (原书第3版)

作者: (美) Tom DeMarco 等 ISBN: 978-7-111-47436-4 定价: 69.00元

公认对软件行业影响最大、最具价值的著作之一，历时15年全面更新
与《人月神话》共同被誉为软件图书领域最为璀璨的“双子星”，近30年全球畅销不衰

在软件管理领域，很少有著作能够与本书媲美。全书从管理人力资源、创建健康的办公环境、雇用并留用正确的人、高效团队形成、改造企业文化和快乐工作等多个角度阐释了如何思考和管理软件开发的重大问题——人（而不是技术），以得到高效的项目和团队。

设计原本——计算机科学巨匠Frederick P. Brooks的反思 (经典珍藏)

作者: (美) Frederick P. Brooks, Jr. ISBN: 978-7-111-41626-5 定价: 79.00元

图灵奖得主、《人月神话》作者Brooks封笔之作，揭秘软件设计神话！
程序员、项目经理和架构师必读的一本书！

《设计原本》开启了软件工程全新的“后理性时代”，完成了从破到立的圆满循环，具有划时代的重大里程碑意义，是每位从事软件行业的程序员、项目经理和架构师都应该反复研读的经典著作。全书以设计理念为核心，从对设计模型的探讨入手，讨论了有关设计的若干重大问题：设计过程的建立、设计协作的规划、设计范本的固化、设计演化的管控，以及设计师的发现和培养。

目

本书赞誉	1.8.3
译者序	1.8.6
序一	1.8.8
序二	1.8.8
前言	1.8.8
作者简介	1.8.9
第1章 基于模型的系统工程概览	1
1.1 什么是 MBSE	1
1.2 MBSE 的三大支柱	4
1.2.1 建模语言	4
1.2.2 建模方法	4
1.2.3 建模工具	6
1.3 MBSE 迷思	7
小结	8
第2章 系统建模语言概览	9
2.1 SysML 是什么, 不是什么	9
2.2 SysML 确实基于 UML——但 你可以从 SysML 开始	10
2.3 SysML 图概览	11
2.4 通用的图概念	13
小结	17
第3章 模块定义图	18
3.1 目的	18
3.2 何时创建 BDD	18

录

3.3 BDD 外框	19
3.4 模块	19
3.4.1 结构特性	21
3.4.2 行为特性	30
3.5 关联: 属性的另一种表示法	33
3.5.1 引用关联	34
3.5.2 组合关联	36
3.6 泛化	38
3.7 依赖	40
3.8 执行者	41
3.9 值类型	42
3.10 约束模块	44
3.11 注释	46
小结	47
第4章 内部模块图	48
4.1 目的	48
4.2 何时创建 IBD	48
4.3 对模块的回顾	49
4.4 IBD 外框	50
4.5 BDD 和 IBD: 相互补充的 模块视图	51
4.6 组成部分属性	51
4.7 引用属性	51
4.8 连接器	52
4.9 项目流	54

4.10 内嵌组成部分和引用	55	6.7 边	76
4.10.1 点标识法	56	6.7.1 对象流	76
4.10.2 连接内嵌属性	57	6.7.2 控制流	78
小结	57	6.8 再次阐述动作	79
第5章 用例图	58	6.8.1 动作何时开始	79
5.1 目的	58	6.8.2 调用行为动作	80
5.2 何时创建用例图	58	6.8.3 发送信号动作	81
5.3 什么是用例	58	6.8.4 接受事件动作	83
5.3.1 用例说明书	59	6.8.5 等待时间动作	85
5.3.2 用例与场景	61	6.9 控制节点	86
5.4 用例图外框	61	6.9.1 初始节点	87
5.5 用例	62	6.9.2 流最终节点和活动最终 节点	87
5.6 系统边界	63	6.9.3 决定节点	88
5.7 执行者	63	6.9.4 合并节点	89
5.8 将执行者与用例关联	63	6.9.5 分支节点	89
5.9 基础用例	64	6.9.6 集合节点	91
5.10 内含用例	64	6.10 活动分区: 把行为分配给结构	92
5.11 扩展用例	66	小结	93
小结	67	第7章 序列图	94
第6章 活动图	68	7.1 目的	94
6.1 目的	68	7.2 何时创建序列图	95
6.2 何时创建活动图	69	7.3 序列图外框	95
6.3 活动图外框	69	7.4 生命线	96
6.4 关于令牌流的一个词	70	7.5 消息	99
6.5 基本动作	71	7.6 析构事件	105
6.6 对象节点	73	7.7 执行说明	106
6.6.1 栓	73	7.8 约束	108
6.6.2 活动参数	74	7.8.1 时间约束	108
6.6.3 流与非流	75	7.8.2 期间约束	109

7.8.3 状态常量	109
7.9 组合片段	110
7.9.1 opt 操作符	111
7.9.2 alt 操作符	112
7.9.3 loop 操作符	113
7.9.4 par 操作符	115
7.10 交互使用	116
小结	117

第 8 章 状态机图 118

8.1 目的	118
8.2 何时创建状态机图	118
8.3 状态机图外框	119
8.4 状态	120
8.4.1 简单状态	120
8.4.2 复合状态	122
8.4.3 最终状态	123
8.5 转换	123
8.5.1 外部转换与内部转换	125
8.5.2 事件类型	126
8.6 伪状态	131
8.7 区域	132
小结	133

第 9 章 参数图 134

9.1 目的	134
9.2 何时创建参数图	135
9.3 对模块的再次阐述	135
9.4 参数图外框	138
9.5 约束属性	139
9.6 约束参数	140

9.7 值属性	140
9.8 绑定连接器	141
小结	142

第 10 章 包图 143

10.1 目的	143
10.2 何时创建包图	143
10.3 包图外框	144
10.4 命名空间包含标记法	145
10.5 包之间的依赖关系	146
10.6 引入包	146
10.7 特定包	147
10.7.1 模型	147
10.7.2 模型库	147
10.7.3 特征	148
10.7.4 视图	149
10.8 灰色地带：你看的是包图 还是模块定义图	150
小结	150

第 11 章 需求图 151

11.1 目的	151
11.2 何时创建需求图	151
11.3 需求图外框	152
11.4 需求	153
11.5 需求关系	154
11.5.1 包含关系	154
11.5.2 跟踪关系	155
11.5.3 继承需求关系	155
11.5.4 改善关系	156
11.5.5 满足关系	156

11.5.6 验证关系	156
11.6 需求关系标识法	157
11.6.1 直接标识法	157
11.6.2 分隔框标识法	157
11.6.3 插图标识法	158
11.6.4 矩阵	159
11.6.5 表格	159
11.7 基本原理	160
小结	161

第 12 章 分配：横跨关系

12.1 目的	162
12.2 并没有分配图	162
12.3 分配关系的用处	163
12.3.1 行为分配	163

12.3.2 结构分配	164
12.3.3 需求分配	165
12.4 分配关系标识法	165
12.4.1 直接标识法	165
12.4.2 分隔框标识法	165
12.4.3 插图标识法	166
12.4.4 矩阵	166
12.4.5 表格	167
12.4.6 分配活动分区	167
12.5 基本原理	169
小结	169

附录 A SysML 标识法随身参考

附录 B SysML 各版本的变化

基于模型的系统工程概览

MBSE 是一种应用建模方法的正式方式，用于支持系统需求、设计、分析、检验和验证活动，这些活动从概念设计阶段开始，贯穿整个开发过程及后续的生命周期阶段。

——INCOSE, 《Systems Engineering Vision 2020》

你读这本书是为了学习系统建模语言 (Systems Modeling Language, SysML)。你可能是要在自己的系统工程团队中创建 SysML 模型，或者是为了获得 OMG(Object Management Group, 对象管理组织) 认证系统建模专家的称号，也可能二者兼而有之。然而，SysML 只是基于模型的系统工程 (Model Based Systems Engineering, MBSE) 这个更大话题的一个方面。MBSE 是一种实践；它是你所做的某件事情。而 SysML 是一种图形建模语言，让你可以实践 MBSE。MBSE 的实践会为学习 SysML 提供具体情境以及业务案例。

这一章首先会回答最基本的问题：什么是 MBSE？然后会讨论 MBSE 的三大支柱——让我们可以实践基于模型的系统工程的三个点。最后会揭示来自于 MBSE 的迷思，从而使你在交付 MBSE 承诺的投资回报时，更好地应对客户的期望。

1.1 什么是 MBSE

理解 MBSE 方法的最好方式是，首先理解另一种形式：建模实践者在工程中称为基于文档的方法，非实践者则称为“我们一直以来处理此事的方式”。不管他们是应用基于文档的方法，还是 MBSE，系统功能工程师都会执行《INCOSE 系统工程手册》(INCOSE 指的是国际系统工程委员会) 所描述的生命周期活动。然而，两种方法之间最关键的区别在于生命周期活动的主要产出物的特质。

使用基于文档的方法，系统工程师会手动生成以下一种或多种产出物：操作概念（Concept of Operations, ConOps）文档、需求说明书、需求跟踪和验证矩阵（Requirement Traceability and Verification Matrice, RTVM）、接口定义文档（Interface Definition Document, IDD）、 N^2 表（也叫做 N 平方表——结构化接口的矩阵）、架构说明文档（Architecture Description Document, ADD）、系统设计说明书、测试案例说明书、特性工程分析（例如：对可靠性、可用性、可排程性、吞吐量以及响应时间的分析）。基于文档的系统工程会以多个文本文档、电子表格、图表和演示文档（以及配置——用来在各个地方管理他们）的形式来创建这些产出物。

问题在于：对于系统工程来说，基于文档的方法非常昂贵。更准确的说法是，它本没有必要那么昂贵；你会花费整个生命周期的很大一部分成本来维护那些彼此分离的产出物。如果你不付出那些代价，那么产出物就会变得不一致，并最终被废弃。

考虑一下下面这个会在日常生活中出现的场景。一位系统架构师决定对设计做第四次迭代，重构系统层次关系中的一个单独模块，把它分割成两个模块，从而实现对关注点更好地分离。他决定重命名最初的模块，更好地表明、更窄的新关注点。为了完全且一致地实现这项变更，他需要定位所有包含那个模块的文本文档、表格、矩阵、图表以及演示文档，从各种各样的文件服务器、内部网站以及配置管理库打开每个文件，然后手动把相同的变更输入到所有那些产出物中。

这种方法会花费大量时间，而且很容易出错。架构师可能会把新模块的名称敲错。更重要的是，他需要提前知道所有需要更改的产出物，这可能会有很多。他很可能会遗漏很多，那样就会导致它们与其他部分不一致。以那些文档作为生命周期阶段输入的开发团队就会因此产生问题。这对于项目经理也是个问题，他必须负责修改日程安排，增加生命周期的成本，以修正传播到生产环境中的缺陷。

这种场景在采用基于文档的传统方法来执行系统工程的组织中非常常见。不一致性是问题所在。而 MBSE——当正确实践的时候——就是解决方案。

使用 MBSE 方法，系统工程师会执行同样的生命周期活动，并创建同样的交付物。但是交付物并不是生命周期活动的直接输出，它们也不是主要的产出物。使用 MBSE 方法，那些活动的主要产出物是一份集成、清晰并且一致的系统模型，它是使用专门的系统建模工具创建的。所有其他产出物都是次要的——使用同样的建模工具从系统模型自动生成。

系统模型是设计的中心；设计中做出的每个决定都被捕获为一个模型元素（或者元素之间的关系），它只位于系统模型的单一位置。使用 MBSE 方法，所有图表和自动生成的文字产出物都只是底层系统模型的视图；它们并不是模型本身。由此而来的最大区别就是，MBSE 所获得的投资回报率要远远高于传统方法。

让我们回到日常情境中，这次使用 MBSE 方法。系统架构师决定重命名最初的（已经重构完毕）的模块，更好地表明更窄的新关注点。为了完整而一致地实现这项变更，他在系统模型层次关系中找到那个模块（通常可以在建模工具中使用关键字搜索），并为那个模块键入新的名称。就这么简单。

建模工具会自动（且立即）把变更传递给所有出现过这个模块的图表，而不管那个集合有多大。毕竟那些图表只是底层模型的视图。如果模型发生了变更，那么图表就会随之而变。建模工具还会在架构师下次从模型导出文字产出物的时候，把变更插入其中。这样，模型的各种视图之间就不会产生任何不一致的情况。

MBSE 之所以能够承诺不断提升质量和可提供性，就是因为一个简单的原因：有了成功的预防，修复缺陷的代价可以降到最低。这种方法的核心是一种新的工程产出物，我们把它叫做系统模型。

我刚刚描述的方法实际上是一种混合形式，它填补了 MBSE 和基于文档的方法之间的鸿沟。在客户要求提供文字产出物作为交付物以进行评审和批准的时候，基于模型的工程组织就需要采用这种混合的方法。然而，没有这种约束的组织可以以纯粹的方式来实践 MBSE。

获得最高级别 MBSE 成熟度等级的组织会完全放弃创建文字产出物的想法。系统模型本身就是可以评审和批准的产出物。当设计从一个阶段到另一个阶段，它也是可以传递、改善和进化的产出物。

生产软件系统的组织甚至可以使用（足够稳健的）建模工具，把系统模型转换成软件模型，并最终转换成符合生产环境质量要求的源代码。这种级别的 MBSE 成熟度模糊了设计和开发之间的界限，让我们可以更快地创建原型以及进行系统模拟。然而，在任何时候，模型都是最重要的产出物，当客户需求发生改变或者做出新的设计决定时候，都会进行修改。所有其他产出物，包括源代码，都是通过模型的产出物自动生成的，持续与模型以及其他产出物保持一致。

这就是 MBSE。

1.2 MBSE 的三大支柱

你要如何实践 MBSE 呢？你需要了解哪些知识呢？

简而言之，你需要通晓三种东西：一种建模语言、一种建模方法以及一种建模工具。我把这三者称为 MBSE 的三大支柱。作为创建集成系统模型的设计团队成员，你会使用专门的建模工具来执行建模方法中规定的一系列设计任务，向以标准化建模语言表达的集成系统模型中添加元素（以及元素之间的关系）。

建模语言知识本身让你可以在纸上或者白板上草绘系统设计想法，从而快速有效地和其他团队成员沟通。学习建模语言是你应该获得的第一项能力，也是这本书的关注点所在。然而，实践 MBSE 需要你掌握以上三种技能，以获得这种方法提供的投资回报率。

让我们依次详细讲述这三个支柱。

1.2.1 建模语言

当你创建模型的时候，就是在说一种语言。它不是你孩童时在家和学校里面学到的自然语言。也不是我现在用来和你沟通的自然语言。它是一种建模语言：一种半正式的语言，定义你放到模型中的元素的种类，以及元素之间的关系，并且在图形建模语言的情况下，还要定义你可以使用的一系列标识法，从而在图表中显示元素和关系。

MBSE 实践者通常会使用系统建模语言（SysML）来创建系统结构、行为、需求和约束的模型。SysML 是本书的关注点所在，但它并非是唯一的建模语言。其他设计领域的工程师和分析师（例如，系统之系统、软件、硬件、性能、业务过程等）都有可用的建模语言，更适合他们所设计的系统类型。像 SysML 一样，那些语言中很多都是图形建模语言（例如：UML、UPDM、BPMN、MARTE、SoaML、IDEFx 等）；其他是文本的建模语言（例如：Verilog、Modelica 等）。

这里的关键是，每种建模语言都是用于沟通的标准化媒介；在特定语言中定义的规则会赋予模型的元素和关系清晰的意义。能够构建和阅读形式良好的模型，是 MBSE 方法的核心。

1.2.2 建模方法

学习建模语言只是 MBSE 路上的第一步。建模语言会定义语法：决定特定模型

的形式是否良好的一系列规则。那些规则不会指定如何和什么时候使用语言来创建模型；也不会指定任何特定的建模方法。

相反，建模方法类似于路线图；它是建模团队创建系统模型要执行的一系列设计任务的文档。更准确的说法是，它是确保团队中所有人都以一致的方式构建模型，并朝着同一个目标努力的文档。没有这样的指导，团队中每个成员构建到系统模型中的内容就会在广度、深度和准确度方面有很大区别。

和所有项目类似，MBSE 项目也需要一个计划，而每个计划首先要声明目的。你的团队首先要回答以下问题：你为什么要建模？更准确的说法是，建模工作期望得到什么结果？你创建的模型是否只作为所有设计决定的权威中心记录？你是否需要从模型自动生成文本产出物，用于评审和批准？你会使用模型来管理需求可跟踪性，并执行下游影响分析吗？你会使用模型来执行另一种配置的优劣势研究吗？系统模型会与专门的公式计算工具和模拟工具整合，以直接执行模型吗？模型本身会是下游设计和开发团队工作——像软件、硬件、可靠性 / 可用性 / 性能分析——的输入项吗？模型会包含在开发后验证系统程序集的整合测试以及接受性测试的案例吗？这些问题的答案会决定团队建模工作的目的。

一旦你的团队已经明确了那些目的，你就可以回答一系列新问题。系统的外部环境有多少需要建模？系统的哪些部分需要建模？哪些行为需要建模？你需要以多么深入的程度解析内部结构和行为？在模型中需要有哪些细节？哪些细节可以忽略（可以留给开发团队在实现的时候细化）？这些问题的答案会决定构建系统模型的范围。

对范围的定义会设置团队工作的目标；它让你的团队可以决定模型何时才算完成。说得直白一点，你的团队需要随着时间的推移让模型不断发展，因为需求会改变，团队会做出新的设计决定。在这种情境下，“完成”意味着模型满足了你在项目计划中概述的目的。

模型的范围还决定了团队将会遵循的建模方法。文献已经记录了多种建模方法。你的团队可以采纳其中一种已经存在的方法，对其进行剪裁以满足你的需要和目的。如果没有一种适合你，那么还可以创建自定义的建模方法。然而，那些内容不在本书的讨论范围之内。

这里的重点在于帮助你精通 SysML 这门建模语言，而不是教你某种特定的建模方法。SysML 与方法无关；你可以使用 SysML 创建系统模型，不管你认为对于自己的需求哪种建模方法最合适。然而，我在此使用了一些篇幅来列举一些众所周知的建

模方法（以及一些参考，其中提供了更深入的内容）来帮助你。

- ❑ 方法：INCOSE 面向对象系统工程方法（Object Oriented Systems Engineering Method, OOSEM）

- ❑ 参考：Friedenthal、anford 等著，《A Practical Guide to SysML, Second Edition: The Systems Modeling Language》（Boston: MK/OMG 出版社，2011）

- ❑ 方法：Weilkiens 系统建模（System Modeling, SYSMOD）方法

- ❑ 参考：Weilkiens、Tim 著，《Systems Engineering with SysML/UML: Modeling, Analysis, Design》（Boston: MK/OMG 出版社，2008）

- ❑ 方法：IBM Telelogic Harmony-SE

- ❑ 参考：Hoffmann、Hans-Peter 著，“(Harmony-SE/SysML Deskbook: Model-Based Systems Engineering with Rhapsody)”，Rev. 1.51, Telelogic/I-Logix 白皮书（Telelogic AB，2006 年 5 月）

这些建模方法广泛分布在系统工程生命周期的多个阶段中。并非这些方法规定的所有步骤都适用于你的项目。你所采用的任何建模方法都需要剪裁，以满足你的项目的特殊需要。不过，这些方法都是很好的开始。

1.2.3 建模工具

熟练掌握建模工具是 MBSE 的第三个支柱。建模工具是一类特殊的工具，设计和实现它们就是为了遵守一种或多种建模语言的规则，让你可以用那些语言创建形式良好的模型。

建模工具和绘图工具——像 Visio、Schematic、SmartDraw、ProcessOn 等——各不相同。你可以使用绘图工具创建图——页面上的形状。在那些图下面并没有任何模型可以保证彼此之间自动保持一致。而使用建模工具，你创建的是模型——一系列元素以及元素之间的关系，可能会有一系列图，作为底层模型的视图。

当你在建模工具中修改图中的元素时，实际上是在底层模型中修改元素本身。然后建模工具会立即更新所有其他显示了相同元素的图。这是很强大的功能——也是只有这类工具才能够提供的功能。

注意，建模语言规格——像 SysML——与厂商无关。特定的建模工具只是一家厂商对于语言规格的实现。几家商业化工具厂商以及非盈利组织已经为各种建模语言创建了建模工具。这些工具在价格、功能以及对建模语言规格的符合程度上都各有不

同。选择最好的工具——基于项目的特殊要求以及价格的限制——应该是在你组织中采纳 MBSE 过程的一部分。

和 SysML 以及其他建模语言非常类似，我也是与厂商无关的。我不会在这本书中评价各种产品的优劣。不过，我为你列举了一些 SysML 建模工具供你研究，因为对于你的组织来说，评定这些工具是必须要做的工作。以下是商业级别（即收费的）的建模工具：

- ❑ Agilian（厂商：Visual Paradigm）
- ❑ Artisan Studio（厂商：Atego）
- ❑ Enterprise Architect（厂商：Sparx Systems）
- ❑ Cameo Systems Modeler（厂商：No Magic）
- ❑ Rhapsody（厂商：IBM Rational）
- ❑ UModel（厂商：Altova）

以下是免费的建模工具，基于 Eclipse 公共许可（Eclipse Public License, EPL）或者一般公共许可（General Public License, GPL）提供：

- ❑ Modelio（创建者：Modeliosoft）
- ❑ Papyrus（创建者：Atos Origin）

当你选择工具的时候，需要考虑多方面因素。我强烈推荐你选择一种兼容 XML 元数据交换（XMI）的工具。XML 标准让兼容的工具可以交换模型数据。当你需要变更工具（并且受到成本限制）的时候，这能确保你不会陷入厂商的闭锁陷阱中。

1.3 MBSE 迷思

MBSE 迷思来自于利益相关者，他们是外部客户以及内部的下游设计和开发团队，对 MBSE 略有所知，但是不会自己来实践。这些都是期望从你那里得到交付物的利益相关者。他们至少在理论上了解，你会从系统模型自动生成那些交付物。

深深困扰他们的迷思是，MBSE 是一种简单按钮：你按下它，好东西就会喷薄而出。具体来说，他们会错误地相信，MBSE 会让所有工程任务更加简单，并在生命周期的每个时间点都会降低成本。

但事实上，MBSE 不会（也不能）减轻很好地创建系统架构和设计系统所需的繁重工作。它不会降低系统说明和设计过程所需要的工程严密性——对于创建任何成功的系统，相同的严密性都非常必要。

建立好模型很困难，做好设计也很困难。我们可能会创建出很差的模型。也可能为设计很差的系统创建出良好的模型。为设计良好的系统创建出好的模型，在广度、深度和保真度方面都达到要求，以满足模型的目的，需要花费时间和精力，并要遵循一定的规则。只是按按钮，你无法从模型中受益，除非你努力地工作，首先把它完成。

当发生变更的时候——当做出新的设计决定，而利益相关者的需求在系统生命周期中始终变化时——MBSE 的投资回报就会显现出来。当然，发生变更是不可避免的。直到那个时候，MBSE 的优势会自动显现出来，调整利益相关者的期望，并驱散他们对 MBSE 的迷思。

小结

MBSE 是执行系统工程的一种方法，它比基于文档的传统方法投资回报率更高。MBSE 的实践需要三大支柱：建模语言、建模方法和建模工具。接下来的章节会讲授 SysML（一种图形建模语言，它已经成为 MBSE 实践者的事实标准），从而帮助你确立第一支柱。

1.3 MBSE 迷思

系统建模语言概览

SysML 是一种应用广泛，表意丰富的图形建模语言，可以使系统设计的重要方面——结构、行为、需求和参数（数学模型）——可视化，便于就设计内容进行沟通。如第 1 章所讨论的，SysML 可以作为 MBSE 的第一支柱使用。

本章提供了对 SysML 的高层次概览：SysML 的总体目的、9 种 SysML 图各自的目的，以及对它们全部适用的普遍概念。这些讨论会为后续章节深入探讨每种图提供重要的背景信息。

2.1 SysML 是什么，不是什么

SysML 是多种图形建模语言中的一种。这里的关键词是语言。SysML 是一种语言——人与人沟通的媒介。它和我们所说的自然语言（例如：汉语、日语、英语）一样，有语法和词汇。SysML 是 MBSE 实践者在创建系统模型时“说”的语言，可以把他们的系统设计观点可视化，并与利益相关者沟通。

之所以给“说”字加引号，是因为 SysML 是一种图形语言。它的词汇包含能够表示特殊意义的图形标识。例如，带有箭头的虚线和带有箭头的实线（后续章节会详细介绍）拥有不同的意义。关键在于 SysML 的目的：在利益相关者之间可视化并沟通系统设计。

SysML 的语法和标识法都定义在标准说明书里面，它由对象管理组织（Object Management Group, Inc. OMG）发布，并归其所有。OMG 是由上百家计算机行业公司、政府机构和学术单位组成的联合体，它们合作以创建一系列企业集成标准，并推动业务技术。你可以在它的网站 www.omg.org 上找到更多关于 OMG 的信息。你在那里还会找到 SysML 的说明文档——关于 SysML 规则信息的重要资源。

SysML 说明书试图以精确、清晰的方式来定义这种建模语言的语法，很大程度上它已经成功达到了这个目的。但说明书中的文字有时可能难以解析和执行。这份 SysML 信息主要来源的目标读者是建模工具厂商以及建模语言的设计者（以及撰写这个主题相关书籍的建模极客们）。简言之，SysML 说明书并不是给初学者使用的。

如果你觉得那并不会难倒你，那么更可怕的还在后头。如下一节所述，SysML 并不是一种独立的语言。而是统一建模语言（Unified Modeling Language, UML）子集的一种形式（扩展）。因此，如果你想知道 SysML 语法和词汇表的完整定义，那么还需要参考 UML 说明文档的一部分内容（在 OMG 的网站上也可以找到）。

如果你第一次接触 SysML，我建议你跳到附录 A，从而对其各种各样的图形标识法有个初步认识。那些标识法将在你创建的 SysML 图中出现。这些标识法也组成了系统建模语言的词汇表。

那就是 SysML：一种建模语言。了解 SysML 不是什么也同样重要：它不是一种建模方法。说得更明白一点，正式的 SysML 说明文档只定义了语言本身（语法和词汇），而没有规定任何特定的建模方法。

注意 请参考第 1 章对建模语言和建模方法之间区别的讨论。

例如，SysML 规格说明书并没有告诉你应该在生命周期的哪个时间点创建用例图。也不会指定你必须使用活动图来详细说明用例。它不会要求你创建一系列内部模块图（Internal Block Diagram, IBD），每个模块专注于说明系统架构的特定方面。所有这些方法学上的决定，都不在 SysML 说明书的讨论范围之内。采纳建模方法并对其进行剪裁完全是你和团队的工作，你们要自己确定哪种方法能达到项目的特定目标。

2.2 SysML 确实基于 UML——但你可以从 SysML 开始

如前所述，SysML 并不是一种独立的语言。它是 UML 的一种形式或者扩展，而 UML 是特别为系统工程领域所创建的。UML 被设计为针对软件工程领域的标准化建模语言。系统工程师发现，使用标准的建模语言来构建系统模型很有价值，但他们并不觉得 UML 足以记录系统工程中所有有意义的概念。

本书仅提供部分阅读，如需完整版，请联系QQ: 461573687

提供各种书籍pdf下载，如有需要，请联系 QQ: 461573687

PDF制作说明：

本人可以提供各种PDF电子书资料，计算机类，文学，艺术，设计，医学，理学，经济，金融，等等。质量都很清晰，而且每本100%都带书签和目录，方便读者阅读观看，只要您提供给我书的相关信息，一般我都能找到，如果您有需求，请联系我 QQ: 461573687, 或者 QQ: 2404062482。

本人已经帮助了上万人找到了他们需要的PDF，其实网上有很多PDF,大家如果在网上不到的话，可以联系我QQ。因PDF电子书都有版权，请不要随意传播，最近pdf也越来越难做了，希望大家尊重下个人劳动，谢谢！

备用QQ:2404062482