第 1 章
Chapter 1

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 Dacument，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。也可能会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。