UML 简介[⊖]

关键概念
活动图
类图
通信图
依赖
部署图
泛化
交互框架
多重性
对象约束语言（OCL）
顺序图
状态图
构造型
泳道
用例图

统一建模语言（Unified Modeling Language，UML）是“绘制软件蓝图的标准化语言。UML 用来可视化、描述、构造和文档化软件密集系统的人工制品”[Boo05]。换句话说，就像建筑设计师需要为建筑公司设计蓝图一样，软件设计师也需要创建 UML 图帮助软件开发者开发软件。如果了解 UML 的词汇（图示元素和它们的含义），就可以很轻松地理解和描述一个系统，并向他人解释该系统的设计。

Grady Booch、Jim Rumbaugh 和 Ivar Jackson 在 20 世纪 90 年代中期开发出了 UML 语言，引起了广大软件开发人员的强烈反响。UML 融合了大量当时软件产业所使用的具有竞争力的建模表示方法。1997 年，UML 1.0 被提交给对象管理组织（Object Management Group，OMG）——一个致力于维护计算机产业使用规范的非盈利性组织。同年，UML 1.0 修订成 UML1.1 后不久就被对象管理组织采用了。目前的标准是 UML 2.3[⊜]，也是 ISO 标准。因为这个标准很新，所以很多旧的资料，（如 [Gam95]）都没有使用 UML 符号。

UML 2.3 提供了 13 种不同的图以供软件建模使用。在本附录中，将仅讨论类图、部署图、用例图、顺序图、通信图、活动图和状态图。本书使用了这些图。

应该注意，UML 图有很多可选功能。UML 语言提供了这些选项（有时候是隐藏的），使得软件工程师能表达系统的所有重要方面。同时，还可以灵活地隐藏图中那些与建模无关的部分，避免无关的细节将图弄得杂乱。因此，一种特殊功能的省略并不意味着该功能的缺失，它可能意味着该功能被隐藏了。在本附录中，没有介绍 UML 图的所有功能，而是重点介绍标准选项，尤其是本书用到的那些选项。

类图

为了对类建模（包括类的属性、操作和关系以及和其他类的联系[⊛]），UML 提供了类图，类图提供了系统的静态或结构视图。它并不显示图中类的对象之间通信的动态特性。

类图的主要元素是方框，方框是一些用来描述类和接口的图符。每个方框都被水平地划分为多个部分。顶层部分包含类的名字，中间部分列出了类的属性。属性可以是一些数值，

⊖ 本附录由 Dale Skrien 提供并改编自他的著作《面向对象设计和 Java 设计模式的介绍》(McGraw-Hill，2008）。所用内容已经授权。

⊜ http://www.omg.org/spec/UML/2.3/Infrastructure/PDF/ 和 http://www.omg.org/spec/UML/2.3/Superstructure/PDF/ 这两个互补的文档构成了 UML2 建模语言的完整规格说明。

⊛ 如果你不熟悉面向对象的概念，在附录 2 中给出了简要介绍。

这些数值是由类从它的实例变量中计算出来的，或是从组成它的其他对象中得到的。比如，对象可以时刻知道当前时间，无论何时进行查询，都可以反馈给你。在这种情况下，适合列出当前时间以作为该类对象的属性。然而，对象不可能将该时间存储在它的实例变量中。如果这样的话，它需要一直不断地更新该字段。相反，对象更可能在你请求时间时计算出当前时间（例如，通过查询其他类的对象）。类图的第三部分包含类的操作或行为。操作是指类的对象所能做的事情，通常实现为类的方法。

图 A1-1 介绍了一个简单例子，用 Thoroughbred 类对优良种马建模。它有三个属性——mother、father 和 birthyear，还有三个操作——getCurrentAge()、getFather() 和 getMother()。图中也有一些隐藏的属性和操作没有显示。

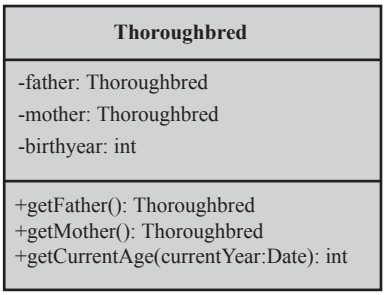

图 A1-1　Thoroughbred 类的类图

每个属性都有名字、类型和可见性级别。类型和可见性都是可选的。类型放在名字后面，并用冒号进行分隔。可见性由前面的－、#、~ 或＋指定，分别代表私有、受保护、包或公有可见性。在图 A1-1 中，所有属性都是私有的，由前面的减号（－）指出。你也可以通过下划线表示属性是否为静态属性或类属性。可以用可见性级别、带名字和类型的参数以及返回类型来表示每个操作。

在类图中，名字设成斜体表示抽象的类或方法。例如，图 A1-2 中的 Horse 类就是抽象类。通过在名字的上方添加短语“ interface”（称为构造型）来指定接口，如图 A1-2 中的 OwnedObject 接口。也可以用空心圆来表示接口。

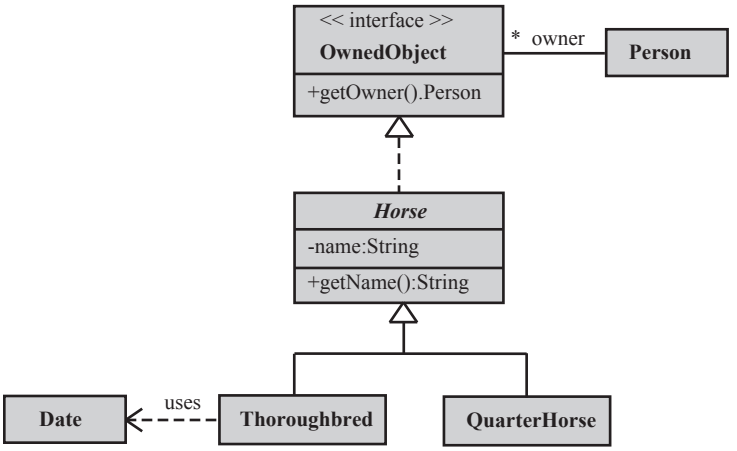

图 A1-2　有关马的类图

值得一提的是，表示类的图符可以有其他可选的部分。例如，处于类方框底层的第四部分可以列出类的责任。在产生属性和操作之前，CRC 卡片上列出的责任如能添加在 UML 图中类方框的第四部分，则这一部分在 CRC 卡片（第 9 章）向类图转换时特别有用。本附录中的所有图均未显示第四部分。

类图也可以表示类之间的关系。类和它的子类之间用实线加上空心的三角箭头连接。箭头的方向是从子类指向父类。在 UML 中，这样的关系称为泛化。比如，在图 A1-2 中，类

Thoroughbred 和类 QuarterHorse 显示为 Horse 抽象类的子类。带虚线的箭头表示接口的实现。在 UML 中，这样的关系称为实现。例如，图 A1-2 中，Horse 类实现了 OwnedObject 接口。

两个类之间的关联是指它们之间存在着结构上的关系。关联用实线表示，有很多可选部分。可以在它的每一端都加上标签，指定关联中每个类的角色。例如，在图 A1-2 中，OwnedObject 和 Person 之间存在关联，Person 在此关联中扮演所有者（owner）的角色。关联线一端或两端的箭头表示可导航性。关联线的每一端也会有多重数值显示。可导航性和多重性将在这一节的后面部分详细讲述。关联也可以通过循环连接它本身，这样的关联表明同一类所创建的不同对象之间可以相互连接。

一端带箭头的关联表明这是一个单向导航。箭头表明从第一个类可以很容易访问关联方向所指的第二个类，但是从第二个类却不能很容易地访问第一个类。关于这种现象的另外一种解释是，第一个类可以察觉到第二个类，但是第二个类的对象将不能直接察觉到第一个类。没有箭头的关联通常表明它是一个双向的关联，这就是图 A1-2 中所指的。也有可能是仅仅意味着可导航性并不重要，所以省略了。

应当注意到，一个类的某个属性和类（其属性所属的类型是类）的关联是一样的。就是说，想表达类中的叫作“ name”的 String 类型的特性，需要如同属性一样地显示它，如图 A1-2 中的 Horse 类。或者，也可以构建一个从 Horse 类到 String 类的单向关联，在此关联中，String 类的角色是“ name”。对于原始数据类型，用属性的办法会更好些，而当特性类在设计中起重要作用时，用关联的方法通常比较好一些，在那种情况下，有那种类型的类方框更有价值。

依赖关系表示类之间的另一种连接，由一条虚线（可选的段末箭头和可选的标签）表示。一个类依赖于另一个类，则改变另一个类也需要改变这个类。从一个类到另一个类的关联就自动表明了一种依赖性。如果类之间已存在关联则不需要虚线。然而，对于短暂的关系（即一个类不需要同另一个类维持长时间的连接，但确实偶尔会用到另一个类），我们应该从第一个类画一条虚线到第二个类。例如，在图 A1-2 中，当 Thoroughbred 类的 getCurrentAge() 方法被调用时，它需要使用 Date 类，所以依赖性被标识为“uses”。

关联一端的多重性是指类关联于其他类的对象的数量。多重性由非负整数或整数范围描述。通过“0..1”描述的多重性是指在关联的一端存在 0 或 1 个对象。例如，世界上的每个人要么有社会保险号，要么没有，因此，多重性 0..1 就可以在类图中的 Person 类和 SocialSecurityNumber 类之间的关联中使用。由“1..*”描述的多重性是指一个或更多，由“0..*”或只是“ *”描述的多重性是指 0 个或更多。在图 A1-2 中，与 Person 类关联的 OwnedObject 端的多重性使用了“*”，因为 Person 可以拥有 0 个或更多的对象。

如果关联的一端有比 1 大的多重性，那么该端涉及的对象将可能被存储于收集中，如集合或有序列表。在 UML 图中，也可以包括收集类本身，但是，由于关联的多重性，这样的类通常被省略并假定存在那里。

聚合是一种特殊的关联，通过图符一端的空心钻石表示。它表明一种“整体 / 部分”关系，箭头所指的类是关联的菱形端的一“部分”。组合表明聚合对部分的强所有权。在组合中，部分随着所有者而生存或消亡，因为它们在独立于所有者的软件系统中没有作用。参见图 A1-3 中关于聚合和组合的例子。

College 有一个包含 Building 对象的聚合，这表示建筑构成了学院。学院也有一个包含

课程的集合。如果学院倒闭了，建筑仍将存在（假定该学院不是物理上消失）并可用于其他事情，但是 Course 对象在学院之外则毫无用处，它是由学院提供的。如果学院作为一个事务实体不复存在了，那么 Course 对象也就没用了，将不复存在。

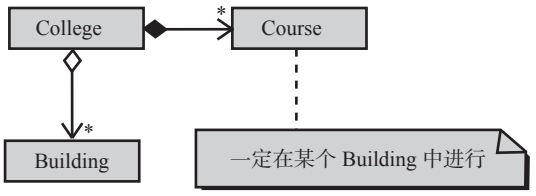

图 A1-3　College、Course 和 Buiding 之间的关系

类图中另一个共同元素是注释，由具有狗耳角的方框表示，通过虚线连至其他图符。它可包含任意内容（文字和图形），类似于编程语言中的注释。它可能包括有关类的作用的解释信息或是该类的所有对象应遵守的约束。如果内容是约束，内容外面将括着大括号。注意图 A1-3 中 Course 类所受到的约束。

部署图

UML 部署图关注软件系统的结构，用于显示软件系统在硬件平台和运行环境中的物理分布。例如，设想你正在开发一个基于 Web 的图形显示包，包的用户将使用他们的 Web 浏览器访问你的网站，得到渲染信息。你的网站应根据用户的描述渲染图像并将其返回给用户。因为图像渲染需要大量的计算，所以你决定将渲染从 Web 服务器中分离出来，放在一个单独的平台上。这样，系统涉及三个硬件设备：Web 客户端（运行浏览器的用户计算机），Web 服务器所在的计算机和渲染引擎所在的计算机。

图 A1-4 显示了此包的部署图。在此图中，硬件部件放在标有“ device”的方框中，硬件部件之间的通信路径用带有可选标签的线表示。在图 A1-4 中，路径是用连接设备的通信协议和网络类型标记的。

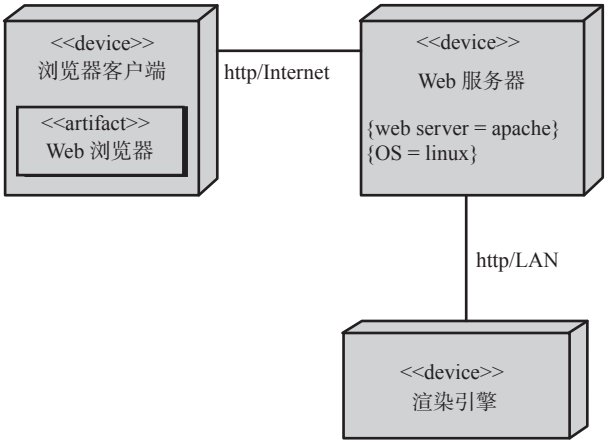

图 A1-4　部署图

部署图中的每个节点也用设备的细节注释。例如，在图 A1-4 中，浏览器客户端被描述为由 Web 浏览器软件构成的人造制品。人造制品是指包含在设备上所运行软件的文件。你也可以描述标记值，就像图 A1-4 中的 Web 服务器节点一样。这些值定义了服务器所采用的 Web 服务器厂家和服务器所使用的操作系统。

部署图也可以显示运行环境节点，将运行环境节点绘制为包含标签“ execution

environment”的方框。这些节点表示可以运行其他软件，如操作系统。

用例图

用例（第 7、8 章）和 UML 用例图可帮助你从用户的角度决定软件的功能和特点。为了让读者了解用例和用例图是如何工作的，以下为在线数字音乐商店管理软件创建一些用例和用例图。软件可能要做的一些事情包括：

- 下载 MP3 音乐文件，并将其存储于应用的存储库中。
- 捕捉流媒体音乐，并将其存储于应用的存储库中。
- 管理应用存储库（例如，删除歌曲或将其添加到播放列表中）。
- 将存储库中的歌曲清单刻录到 CD 上。
- 将存储库中的歌曲清单加载到 iPod 或 MP3 播放器。
- 将一首歌曲从 MP3 格式转换为 AAC 格式，反之亦然。

这并不是一个全面的列表，但它已足够使你理解用例和用例图的作用。

用例通过定义实现明确目标所需的步骤描述了用户和系统之间的交互（例如，将歌曲清单刻录到 CD 上）。一系列步骤的变动描述了各种不同的场景（例如，如果歌曲清单中的所有歌曲不适合放在一张 CD 上该怎么办）。

UML 用例图是所有用例和用例之间关系的视图。它提供了系统功能的整体图示。数码音乐应用的用例图如图 A1-5 所示。

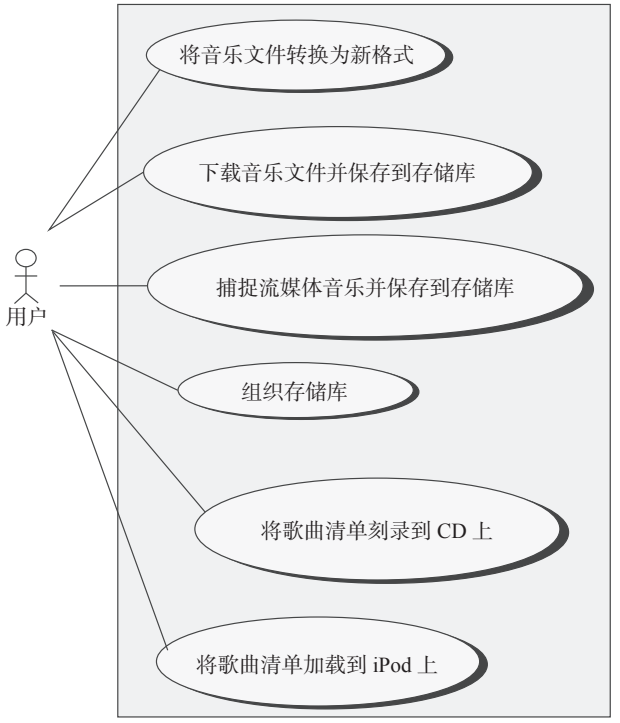

图 A1-5　音乐系统的用例图

在这个图中，木棍小人表示和其他类型的用户（或其他交互元素）相关联的参与者（第 7 章）。复杂的系统通常不止一个参与者。例如，自动售货机应用系统可有三个参与者，表

示客户、维修人员和装货人员。

在用例图中，用例用椭圆形显示。角色通过线连接到所执行的用例上。注意，图中并不包括用例的细节，用例的详细信息需要单独存储。另外，还需要注意，可将用例放在矩形框中，而不能将角色放在矩形框中。矩形框表示的是系统的边界，而角色在系统之外。

系统的一些用例会互相关联，例如，将歌曲清单刻录到 CD 和将歌曲清单存储到 iPod 或智能手机具有相似的步骤。这两种情况下，用户都会首先建立一个空表，然后再将存储库中的歌曲添加进来。为避免用例中的重复，需建立新用例来表示重复活动，然后让其他用例包含这个新用例，作为其他用例的一个步骤。在用例图中，如图 A1-6 所示，这种包含关系用标有 include 的虚线箭头来连接用例和被包含的用例。

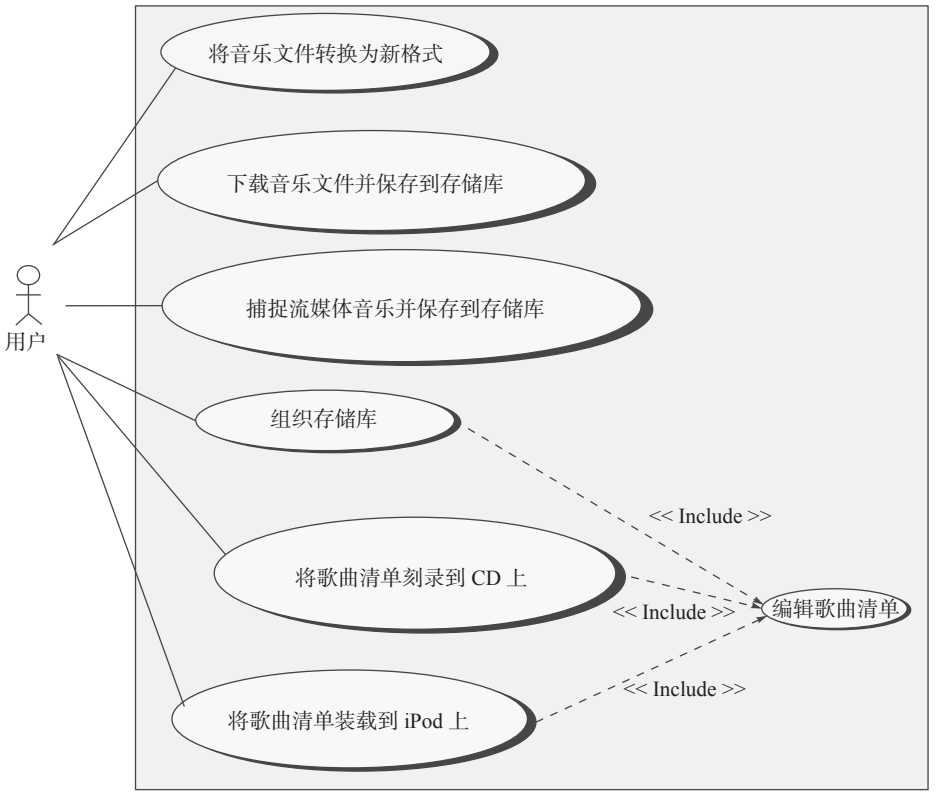

图 A1-6　具有包含用例的用例图

由于用例图显示了所有的用例，因此，用例图有助于确保覆盖系统的所有功能。在我们的数码音乐系统中，确实需要更多的用例，如播放库中歌曲的用例。但是请记住，用例在软件开发过程中最有价值的地方是对每个用例的文字说明，而不是总体用例图 [Fow04]。通过用例的说明，才能对所开发系统的目标形成清晰的理解。

顺序图

类图和部署图显示系统构件的静态结构，而顺序图显示任务执行过程中对象之间的动态通信。它显示了完成任务的对象之间消息发出的先后顺序。顺序图可以显示某个用例或软件系统的某个场景中存在的交互。

在图 A1-7 中，可以看到一个画图程序的顺序图。该图显示了在一幅图中点击一个图形时高亮度显示所涉及的步骤。图顶端行中的每个方框通常对应一个对象，虽然方框也有可能模拟其他东西，比如类。如果方框表示对象（如我们所有例子中的情况），那么在方框内，我们可以有选择地规定冒号前的对象名。也可在冒号之后写上类名，如图 A1-7 中第三个方框所示。在每个方框下面都有一条被称为对象生命线的虚线。顺序图中的垂直轴代表着时间，随着时间逐渐增加，线条逐渐向下移动。

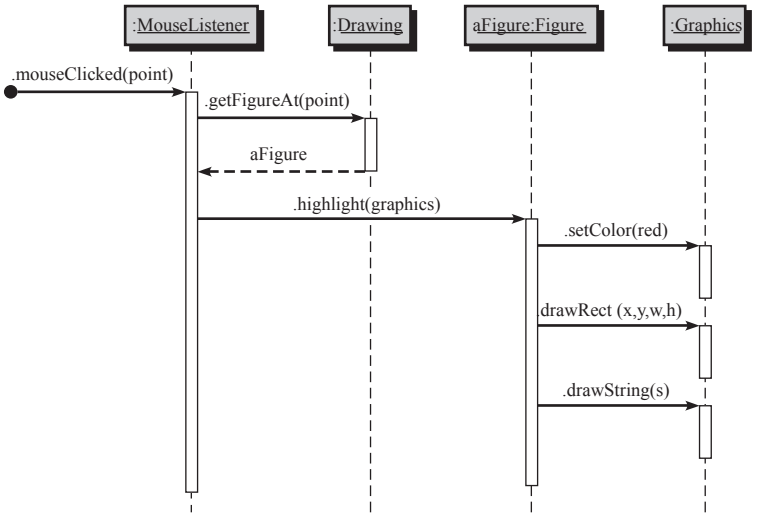

图 A1-7　一个顺序图的例子

顺序图使用从*调用者*到*被调用者*的水平箭头来表示方法调用，箭头上标有方法名称，可选部分包括参数、参数的类型和返回类型。如图 A1-7 所示，MouseListener 调用 Drawing 的 getFigureAt() 方法。当对象在执行方法时（即在堆栈里有活动帧时），在对象的生命线下面可选择性地显示一条空白的条，称为*活动条*。在图 A1-7 中，对于所有方法的调用都绘制了活动条。图也可以通过虚线箭头和可选标记选择性地显示方法调用的返回结果。在图 A1-7 中，getFigureAt() 方法的返回结果由返回对象的名字标记。普遍的做法如图 A1-7 所示，当一个无返回结果的方法被调用时，将不会有返回箭头，因为这样将会使图变得杂乱，而且不会起任何重要作用。带有箭头的黑圆圈表明一条来源未知或无关的发现消息。

你现在应该能够理解图 A1-7 显示的任务了。未知源调用 MouseListener 中的 mouseClicked() 方法，将点击的点作为参数传入。MouseListener 然后调用 Drawing 的 getFigureAt() 方法，并得到返回的一个 Figure 对象。接着 MouseListener 调用 Figure 的 highlight() 方法，传递一个 Graphics 对象作为参数。作为响应，Figure 调用 Graphics 对象的三个方法用红色画出图形。

图 A1-7 比较简单，不包括附加条件和循环。若需逻辑控制结构，则最好为每种情况都绘制一张单独的顺序图。即当消息根据条件可有两条不同的路径时，则需要绘制两张分开的顺序图，为每种可能性绘制一张。

若想要在一张顺序图上包括循环、条件和其他控制结构，则可以使用*交互框*。交互框是矩形，包围图的一部分，并用其所表示的控制结构的类型做标记。图 A1-8 描述了这种情况，高亮显示给定矩形中所有图形所涉及的过程。将消息 rectDragged 发送给 MouseListener，

MouseListener 通知绘图部分高亮显示矩形中的所有图形，具体做法是调用 Drawing 对象的 highlightFiguresIn() 方法，并传递矩形作为参数。此方法要对 Drawing 对象中的所有 Figure 对象进行循环，如果 Figure 和矩形相交，则需要 Figure 高亮显示自身。在方括号内的短语称为守卫，守卫是布尔条件，如果交互框中的动作要继续，则守卫必须为真。

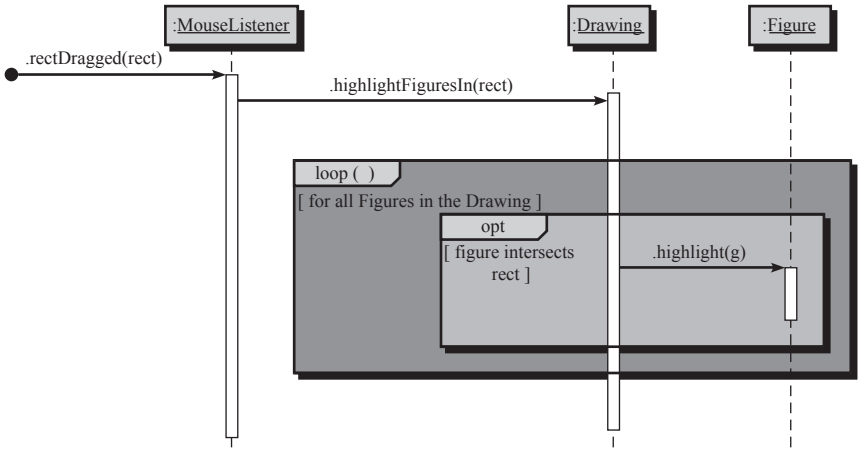

图 A1-8　带两个交互框的顺序图

顺序图还有很多其他特点。例如：

1. 可清楚地分辨出同步和异步消息。同步消息用实体箭头表示，而异步消息用棒形箭头（stick arrowhead）显示。
2. 可用箭头表示对象向它本身发送消息。所用的箭头需从该对象发出，然后折向下，再指回对象本身。
3. 通过绘制指向对象方框的带有适当标记（例如带有 create 标签）的箭头来表示对象的创建。这种情况下，方框将出现在比行动开始时已存在对象对应的方框低一些的位置。
4. 也可通过对象生命线末端的大写 X 显示对象的销毁。其他对象可以销毁一个对象，在这种情况下，一个箭头从其他对象指向 X。X 通常表示该对象已不再有用，因此可以准备进行垃圾回收。

最后三个特点都显示在图 A1-9 所示的顺序图中。

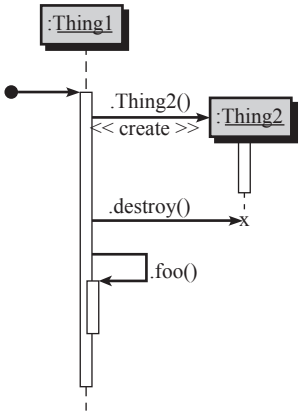

图 A1-9　顺序图中的创建、销毁和循环

通信图

UML 通信图（在 UML 1.x 中称为“协作图”）提供了通信时间顺序的另一种表达形式，但是强调对象和类之间的关系，而不是时间顺序。图 A1-10 所示的通信图与图 A1-7 中的顺序图显示的是相同的动作。

在通信图中相互作用的对象由矩形表示。对象之间的关联由矩形之间连接的线表示。对于图中启动消息传递序列的对象，通常有传入箭头指向该对象。箭头由数字和消息名称标

记。如果传入消息被标记为数字 1 并且它使接收对象调用其他对象的其他消息，则这些消息将沿关联线从发送者到接收者的箭头表示，并按照它们被调用的顺序，标以数字 1.1、1.2 等。如果这些消息又调用了其他消息，那么另外的十进制小数点和数字将被添加到标记这些消息的数字中，说明消息传递的进一步嵌套。

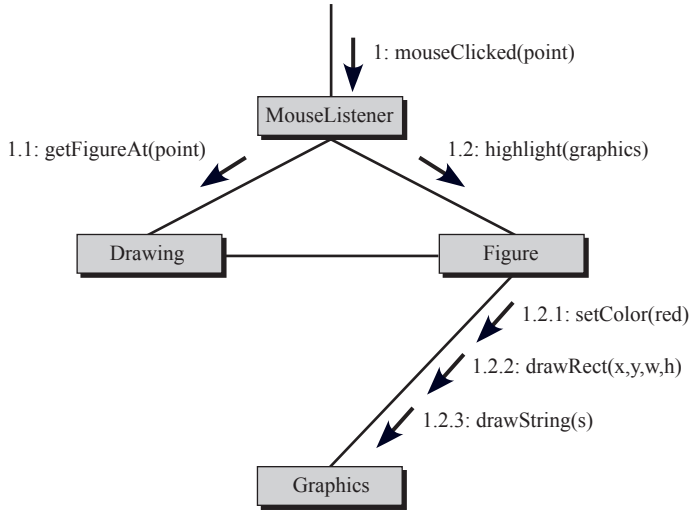

图 A1-10　UML 通信图

在图 A1-10 中，可以看到 mouseClicked 消息调用 getFigureAt() 方法，然后又调用 highlight() 方法。highlight() 消息调用其他三个消息：setColor()、drawRect() 和 drawstring()。每个标签上的数码编号就像每条消息的时序性质那样显示嵌套。

有很多可选特点被添加进箭头标记中。例如，可以在数字前面添加一个字母。传入箭头可以标记为 A1 ：mouseClicked(point)，指明可执行线程 A。如果其他消息在其他线程中执行，那么它们的标记前面将会是一个不同的字母。例如，如果 mouseClicked() 在线程 A 中可执行，但是它创建了一个新线程 B，并且调用该线程中的 highlight()，那么从 MouseListener 到 Figure 的箭头将被标记为 1.B2：highlight(graphics)。

如果对显示对象之间的关系及对象间传递的消息感兴趣，那么通信图可能是比顺序图更好的选择。如果对命令传递的时间顺序感兴趣，那么顺序图也许更好。

活动图

UML *活动*图通过系统所执行动作之间的控制流来描述系统或部分系统的动态行为。它类似于流程图，但活动图可以显示并行流。

活动图的主要构件是*动作*节点，由圆角矩形表示，对应于软件系统执行的任务。从一个动作节点到另一个*动作*节点的箭头表示控制流。也就是说，在两个动作节点之间的箭头意味着第一个动作完成后，第二个动作才开始。实心黑点表示活动开始的初始节点。被黑圆圈包围的黑点表示活动结束的最终节点。

*分叉*表示活动分为两个或更多并行活动，被绘制成水平黑条，并带有一个流入箭头和两个或更多流出箭头。每个流出箭头都代表一个控制流，此控制流可与其他流出箭头所对应的控制流并行执行。这些并行活动可在一台计算机上使用不同的线程执行，甚至使用不同的计

算机执行。

图 A1-11 显示了一个烤蛋糕的样本活动图。第一步是寻找食谱。一旦找到了食谱，就可以称量和搅拌干原料和湿原料，并且可以预热烤箱。干原料的搅拌可与湿原料的搅拌以及烤箱的预热并行进行。

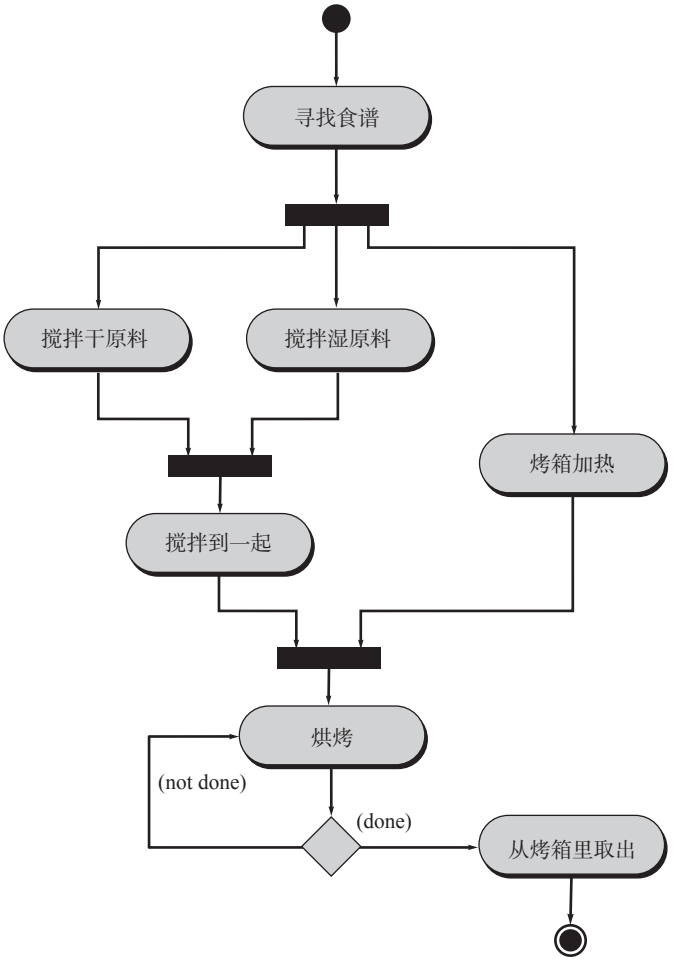

图 A1-11　烤蛋糕的 UML 活动图

合并是同步并行控制流的一种方式。它由水平黑条表示，黑条带有两个或更多的流入箭头和一个流出箭头。直至所有流入箭头所表示的控制流均已完成，流出箭头表示的控制流才能执行。在图 A1-11 中，在将干湿原料搅拌在一起的动作之前有一个合并。该合并表明在两种原料搅拌在一起之前，必须先各自搅拌。图中第二个合并表明，在能够烘制蛋糕之前，所有的原料必须搅拌到一起，并且烤箱必须处于合适的温度。

判定节点与依赖条件的控制流分支相对应。这样的节点用菱形表示，带有一个流入箭头和两个或更多的流出箭头。每个流出箭头都有守卫标记（括号里的条件）。控制流沿守卫为真的流出箭头方向进行。有必要确保条件已经覆盖了所有分支，这样使得每次到达决策节点时，其中都有一个条件为真。图 A1-11 显示了判定节点沿着烘制蛋糕过程的进行状况。如果蛋糕烤好了，那么将会从烤箱中拿出来。否则，还需再烤一段时间。

图 A1-11 中的活动图并未说明的是谁或什么做了每次的动作。通常没有必要准确地区分动作的主体。但是，如果想表明动作在参与者中是如何划分的，则可以采用泳道活动图，如图 A1-12 所示。正如名字所暗示的那样，将图划成条或“道”就形成了泳道，每道都对应一个参与者。一个泳道内的所有动作都由该泳道对应的参与者完成。在图 A1-12 中，Jennie 负责搅拌干原料，然后将干原料与湿原料搅拌到一起，Helen 负责加热烤箱和取出蛋糕，Mary 负责其余的所有事情。

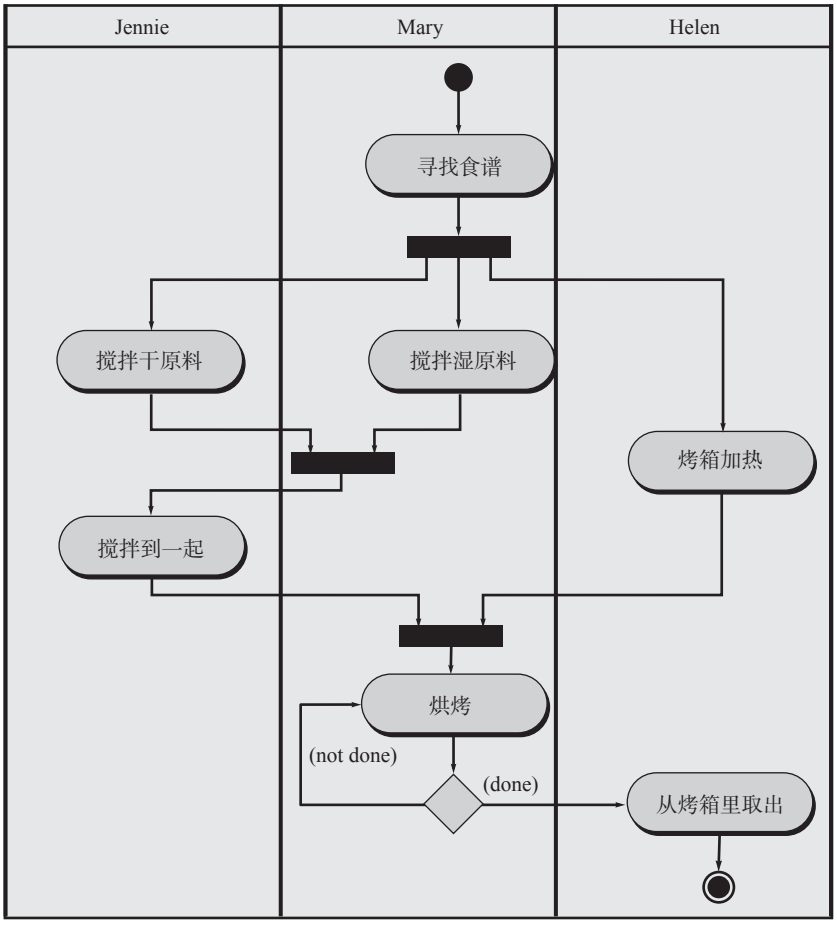

图 A1-12　增加了泳道的蛋糕烘制活动图

状态图

对象在特殊点的即时动作取决于当时的状态，也就是当时变量的值。作为一个小例子，想想带有布尔实例变量的对象。实施操作时，如果变量为真，则对象可以做某件事，如果变量为假，则对象可以做其他事情。

UML 状态图模拟了对象的状态，图中执行的动作取决于这些对象的状态和状态之间的转换。

作为例子，考虑 Java 编译器的一部分状态图。输入编译器的是文本文件，可以认为文本是长字符串。编译器每次读一个字符，并且根据读入的字符决定程序结构。读字符过程中

忽略了“空白”字符（例如，空格、制表、换行和返回字符）以及注释中的字符。

假定编译器将越过空白和注释字符的任务授权给 WhiteSpaceAndCommentEliminator 类。也就是说，该对象的任务是读取输入字符，直到所有空白和注释字符都被读出，这时它会将控制权转交给编译器去读取和处理非空白和非注释的字符串。思考一下 WhiteSpaceAndCommentEliminator 对象是怎样读取字符的，并判断下一字符是否为空白或注释。对象可通过测试下一字符的“”、“\t”、“\n”和“\r”来检查空白字符。但对象如何判断下一字符为注释的一部分呢？举个例子，当它第一次看到“/”时，并不知道该字符是表示除法操作符还是 /= 操作符的一部分，或者是行注释或块注释的开始。为了做出决定，WhiteSpaceAndCommentEliminator 需要记住这样的事实：即它看到了一个“/”，然后移动到下一字符。如果“/”后的字符为另一个“/”或者“*”，那么 WhiteSpaceAndCommentEliminator 就会知道它正在读取注释，能直接到达注释末端而不用处理或保存任何字符。如果第一个“/”后的字符是除了“/”和“*”的任意字符，那么 WhiteSpaceAndCommentEliminator 就会知道“/”表示除法操作或 /= 操作的一部分，它就会停止向前检查字符。

总之，WhiteSpaceAndCommentEliminator 在读取字符的同时，还需要追踪几个事件，包括当前字符是否为空白字符，前面读取的字符是否为“/”，正在读取的字符是否为注释，是否到达了注释的末端等。这些分别对应 WhiteSpaceAndCommentEliminator 对象的不同状态。在每个状态中，WhiteSpaceAndCommentEliminator 对读入的下一字符采取不同的行为。

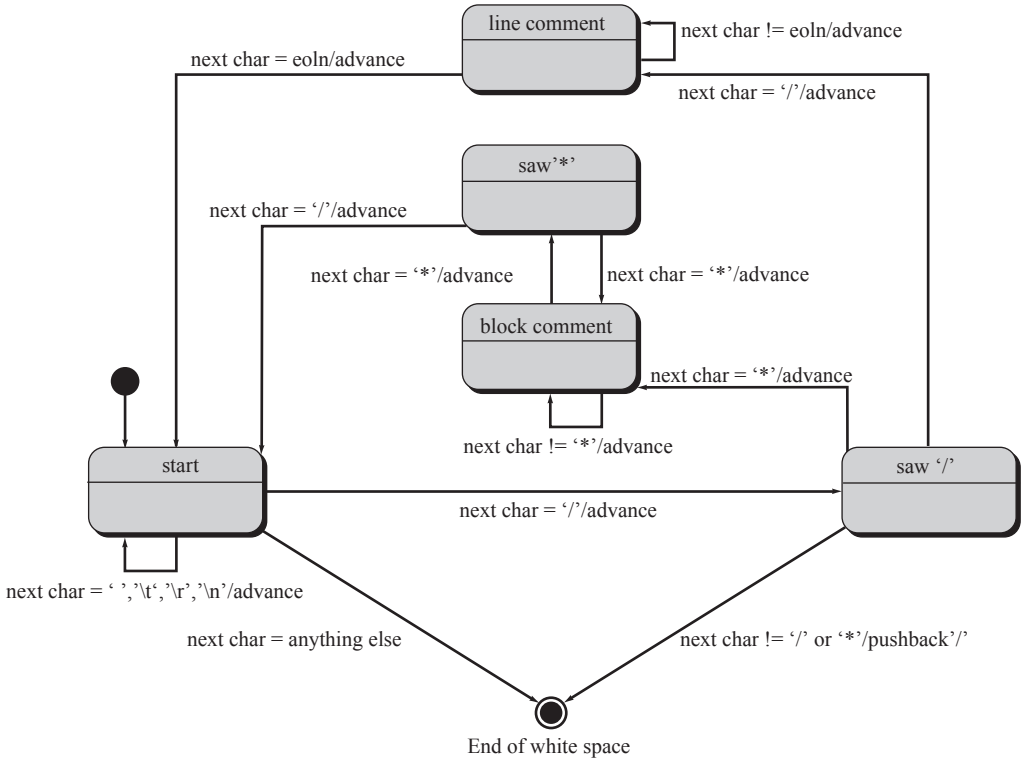

图 A1-13　Java 中越过空白和注释的状态图

为了帮助观察对象的所有状态以及状态的改变，可以利用图 A1-13 所示的 UML 状

态图。状态图通过圆角矩形表示状态，每个矩形的上半部分有它的名字。状态图中还有一个称为“初始伪态”的黑色圆圈，它并不真正表示一个状态，只是指向初始状态。在图 A1-13 中，start 状态即为初始状态。从一个状态指向另一个状态的箭头表明对象状态之间的转换或改变。每个转换由一个触发事件、一个斜杠（/）和一项活动标记。转换标记的所有部分在状态图里是可选的。如果对象正处于某一状态，它的一个转换的触发事件发生了，则执行转换活动，对象将呈现由转换所确定的新状态。例如，在图 A1-13 中，如果 WhiteSpaceAndCommentEliminator 对象处于 start 状态，并且下一字符为“/”，那么 WhiteSpaceAndCommentEliminator 将越过该字符并且转换为 saw‘/’状态。如果“/”之后的字符是另一个“/”，那么对象将前进到 line comment 状态并将一直停留在该状态，直至读到行末尾的字符。如果“/”后的下一字符是“*”，那么对象将前进到 block comment 状态，并一直停留在该状态直到看见另一个“*”，后面跟有一个“/”，这表明块注释的结束。研究此图，确保你已经理解了。注意，当前进到空白字符或注释时，WhiteSpaceAndCommentEliminator 返回到 start 状态并从头开始。这个动作是必需的，因为在 Java 源代码中，任意字符之前都可能有几段连续的注释和空白字符。

对象可以转换到最终状态，通过用白色圆圈里的黑色圆圈来表示最终状态，这表明没有任何转换了。在图 A1-13 中，下一字符不是空白或注释的一部分时，WhiteSpaceAndCommentEliminator 对象结束。注意，所有的转换中除了两个指向最终状态的转换之外都有包括前进到下一字符的活动。这两个指向最终状态的转换不可以前进到下一字符，因为下一字符是编译器感兴趣的单词或符号的一部分。注意，如果对象处于 saw‘/’状态，但下一字符不是“/”或“*”，那么“/”是一个除法操作或 /= 操作的一部分，因此我们不想前进。实际上，我们更希望后退一个字符，使得“/”成为下一字符，这样“/”可以被编译器使用。在图 A1-13 中，后退活动被标记为 pushback‘/’。

状态图有助于发现遗漏或意外情况。也就是说，使用状态图易于保证对于所有可能的状态，已经考虑了所有可能的触发事件。例如，在图 A1-13 中，很容易验证每一状态均包括所有可能字符的转换。

UML 状态图还包括许多图 A1-13 不具有的其他特性。例如，当对象处于某一状态时，它通常不做任何事情，只是等待触发事件的发生。然而，也存在一种特殊状态，称为*活动状态*，在这种状态下，对象执行某些活动，称为 do-activity。要表明一种状态是状态图中的活动状态，将短语“do/”添加进状态圆角矩形的下半部分，短语“do/”的后面跟有该状态下将要完成的活动。do-activity 可以在任何状态转换发生之前结束，在此之后活动状态类似于正常的等待状态。如果活动状态之外的转换在 do-activity 结束之前发生，则 do-activity 中断。

由于在转换发生时触发事件是可选的，因此就可能出现转换的标记中没有触发事件的情况。这种情况下，对于正常的等待状态，对象会立即从该状态转换到新状态。对于活动状态，这样的转换在 do-activity 完成后立即进行。

图 A1-14 通过商业电话的状态描述了这种情况。当呼叫者处于等待状态时，呼叫进入播放音乐等待状态（舒缓的音乐将会播放 10 秒钟）。10 秒之后，该状态的 do-activity 完成，状态行为如同正常的不活动状态。如果呼叫者在播放音乐等待状态下按下“#”键，则来电转换至取消状态，接着立即转换至拨号音状态。如果“#”键在 10 秒钟音乐播放完之前就被按下，则 do-activity 中断，音乐也会立即停止。

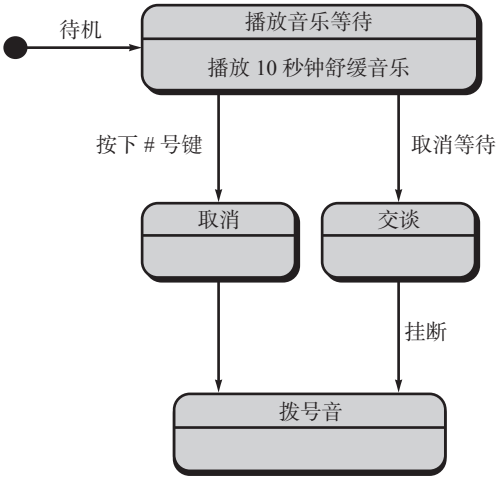

图 A1-14　具有活动状态和无触发转换的状态图

对象约束语言概述

UML 的多种图为设计模型提供了一套丰富的表达方式。然而，图形表达通常是不够的，需要明确和正式的表示信息的机制来约束设计模型的元素。当然，可能会使用自然语言（如英语）来描述约束，但是，这种方法必定导致不一致性和歧义性。因此，一种更为形式化的语言——借鉴集合论和形式化规格说明语言但具有编程语言的少量数学语法——似乎是恰当的。

对象约束语言（Object Constrain Language，OCL）是对 UML 的补充，方法是允许使用形式语法和语义构造有关不同设计模型元素（例如，类和对象、事件、消息、接口）的无歧义语句。最简单的 OCL 语句由以下 4 部分构造：（1）语境定义语句有效的限制情况；（2）特性表示语境的某些特点（例如，如果语境是类，特性可能是属性）；（3）操纵或取得特性的操作（例如，面向算术，面向集合）；（4）关键词（例如，if、then、else、and、or、not、implies）是用来具体说明条件表达式的。

举个简单的 OCL 表达式的例子，考虑第 13 章讨论的影印中心系统。守卫条件位于 jobCostAccepted 事件上，该事件在 PrintJob 对象的状态图中可以引起 computingJobCost 和 formingJob 状态之间的转换（图 13-9）。在该图（图 13-9）中，守卫条件以自然语言表达，表明只有在顾客同意了工作成本时才进行授权。在 OCL 中，表达式可采取如下形式：

```
customer
    self.authorizationAuthority = 'yes'
```

其中，Customer 类（实际上是类的一个具体实例）的布尔属性 authorizationAuthority 对于需要满足的守卫条件必须设为“yes”。

当创建了设计模型时，通常有这样一些实例，在这些实例中，设计说明的一些动作完成之前，前置条件和后置条件必须得到满足。OCL 提供了强大的工具，以形式化的方式说明前置条件和后置条件。举个例子，考虑对影印中心系统的扩展（第 13 章讨论的例子），在该系统中，顾客为印刷工作提供了成本上限，同时，当指定了其他印刷工作特性时，客户提供“下拉式”交付日期。如果成本和交付估计超过了这些界限，该工作将不会提交，并且客户

一定要得到通知。在 OCL 中，可以用下面的方式来描述前置条件和后置条件：

```
context PrintJob::validate(upperCostBound : Integer, custDeliveryReq :
    Integer)
 pre: upperCostBound > 0
    and custDeliveryReq > 07
    and self.jobAuthorization = 'no'
 post: if self.totalJobCost <= upperCostBound
        and self.deliveryDate <= custDeliveryReq
    then
        self.jobAuthorization = 'yes'
    endif
```

OCL 语句定义了一个不变式（inv）——某些行为之前（pre）和之后（post）必须存在的条件。一开始，要建立前置条件，客户必须说明有限制的成本和交付日期，并且授权也必须设置为“no”。在成本和交付确定之后，就可以应用后置条件了。也应该注意到，表达式

```
self.jobAuthorization = 'yes'
```

不是将值设置为“yes”而是声明 jobAuthorization 必须在操作结束时被设置成“yes”。对 OCL 的完整描述已经超出了本附录的范围。完整的 OCL 描述可以在 www.omg.org/technology/ documents/formal/ocl.htm 获得。⊖

扩展阅读与信息资源

现在有大量的书在讨论 UML。讨论在软件开发中使用 UML 的书包括：Hay（《UML and Data Modeling: A Reconciliation》, Technics Publication, 2011 和《Enterprise Model Patterns: Describing the World》, Technics Publications, 2011），Gomaa（《Software Modeling and Design: UML, Use Cases, Patterns, and Software Architecture》, Cambridge University Press, 2011 和《Requirements Engineering: From System Goals to UML Models to Software Specifications》, Wiley, 2009），以及 Podesa（《UML for the IT Business Analyst》, Course Technology, 2009）。提供有用信息的其他书包括：Miles 和 Hamilton（《Learning UML 2.0》, O'Reilly Media, Inc., 2006），Booch、Rumbaugh 和 Jacobson（《Unified Modeling language User Guide》, 2nd ed., Addison-Wesley, 2005），Ambler（《The Elements of UML 2.0 Style》, Cambridge University Press, 2005），以及 Pilone 和 Pitman（《UML 2.0 in a Nutshell》, O'Reilly Media, Inc., 2005）。

网上有大量在软件工程建模中使用 UML 的信息资源。最新的参考文献可以在 SEPA 网站 www.mhhe.com/pressman 中的“analysis”和“design”下找到。

⊖ OCL 最新版本的介绍可以在 http://www.omg.org/spec/OCL/2.3.1/ 找到。

附录 2

Software Engineering: A Practitioner's Approach, Eighth Edition, Chinese Abridgement

面向对象概念

关键概念

属性
类
　边界
　类的特性
　控制器
　类的定义
　实体
封装
继承
消息
方法
操作
多态性
服务
子类
超类

什么是面向对象（Object-Oriented，OO）观点？为什么一个方法被认为是面向对象的？什么是对象？在 20 世纪 80 年代和 90 年代，面向对象概念赢得了广泛的传播，在那时，关于这些问题的答案有很多不同观点，但是今天关于面向对象概念的一个统一观点已经形成。本附录将提供这个重要课题的简要描述，并介绍基本概念和术语。

为了理解面向对象的观点，首先考虑一个现实世界对象的例子——你正在坐着的东西——一把椅子。Chair 类是更大的类 PieceOfFurniture 的子类，椅子这一个体是 Chair 类的成员（通常称为实例）。可以将一组一般属性关联到 PieceOfFurniture 类的每个对象上。例如，在很多可能的属性中，所有家具都有成本、尺寸、重量、位置和颜色。不管我们在讨论桌子还是椅子、沙发还是衣橱，这些属性都适用。因为 Chair 是 PieceOfFurniture 的一个成员，所以 Chair 继承了 PieceOfFurniture 类的所有属性。

我们通过描述类的属性来定义类，但是会缺少一些东西。PieceOfFurniture 类的每个对象都可以通过很多方法进行操作。它可以被买卖，可以被物理地改变（例如，你可以锯掉它的一条腿或将它涂成紫色），也可以将它从一个地方移到另一个地方。每个操作（其他术语为服务或方法）都会修改该对象的一个或多个属性。例如，如果位置属性为一个组合数据项，其定义如下：

location = building + floor + room

那么名为 move() 的操作将会修改形成属性 location 的一个或多个数据项（building、floor 或 room）。为此，move 必须具有这些数据项的“知识”。move() 操作可以用在椅子或桌子上，只要它俩都是 PieceOfFurniture 类的实例，PieceOfFurniture 的有效操作——buy()、sell()、weigh()——被描述为类定义的一部分，并被类的所有实例继承。

Chair 类（通常和所有对象）封装数据（定义椅子的属性值）、操作（改变椅子属性的动作）、其他对象、常量（设值）和其他相关信息。封装意味着该信息的所有部分被打包在一个名字下，并且可以作为一份规格说明或程序构件加以复用。

现在我们已经介绍了一些基本概念，面向对象的更正式的定义会更有意义。Coad 和 Yourdon[Coa91] 是这样定义面向对象的：

面向对象 = 对象 + 类 + 继承 + 通信

其中的三个概念都已经介绍过了，通信概念将在本附录的后面部分讨论。

类和对象

类是面向对象的概念，它封装了描述现实世界实体的内容和行为所需要的数据抽象和过程抽象。描述类的数据抽象被能以某种方式操作数据的过程抽象“墙”所围绕 [Tay90]（如

图 A2-1 所示）。在设计良好的类中，到达属性（和其上操作）的唯一途径是通过组成墙的方法，如图中所示，因此，该类封装了数据（在墙里面）和操作数据的处理（组成墙的方法）。这样做可以获得信息隐藏（第 11 章），并减少由变更引起的副作用。

因为方法倾向于操作一组受限的属性，所以它们的内聚性得到了改进，并且因为通信仅仅通过组成“墙”的方法才能进行，所以类趋向于减弱和系统[⊖]其他元素的连接。

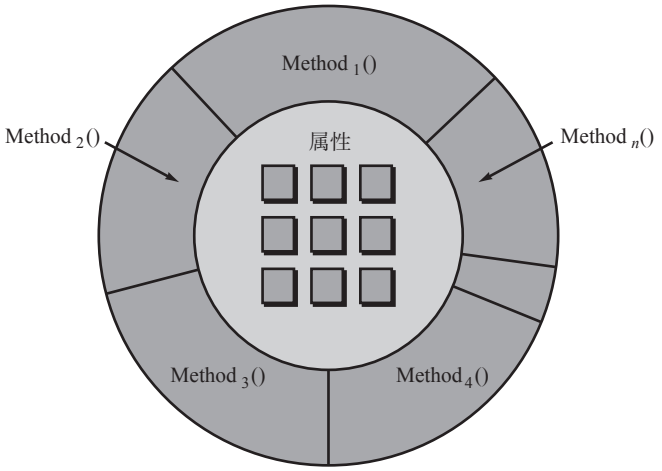

图 A2-1　类的图解表示

换种说法，类是一般化的描述（例如，模板或蓝图），它描述了相似对象的集合。通过定义，对象是特定类的实例并继承类的属性和控制属性的操作。*超类*（通常称为*基类*）是对与其关联的一组类的泛化。*子类*是超类的特殊化。例如，超类 MotorVehicle 是类 Truck、SUV、Automobile 和 Van 的泛化。子类 Automobile 继承 MotorVehicle 的所有属性，并且还有特定于汽车的其他属性。

这些定义意味着存在一种类的层次结构，在这种结构中，超类的属性和方法由子类继承，每个子类又可以增加附加的“私有”属性和方法。例如，操作 sitOn() 和 turn() 对于 Chair 子类是私有的。

属性

属性关联于类并以某种方式描述类。属性可以具有由枚举域定义的数值。在大多数情况下，域仅仅是一组特定的值。例如，假定 Automobile 类有一个 color 属性。color 的值域为 {white，black，silver，gray，blue，red，yellow，green}。在更复杂的情况下，域可能是一个类。继续这个例子，Automobile 类还有一个 powerTrain 属性，该属性本身就是一个类。PowerTrain 类将包括描述车辆的特定发动机和变速器的属性。

可以通过将缺省值（特征）赋给属性来扩大特征（值域）。例如，color 属性的缺省值为 white。通过分配 { 值，概率 } 对，将某个概率与特定的特征联系起来也是有用的。考虑汽车的 color 属性。在某些应用系统中（例如，制造计划），就有必要为每种颜色分配一个概率

⊖ 然而，值得注意的是，在面向对象系统中，耦合会成为一个严重的问题。当系统中不同部分的类用作属性的数据类型和方法的参数时，耦合就产生了。即使只能通过过程调用进入对象，也并不意味着耦合必然很低，只不过比直接进入对象的程度低而已。

（例如，白色和黑色作为汽车颜色的概率非常高）。

操作、方法和服务

对象封装着数据（表示为属性的集合）和处理数据的算法。将这些算法称为操作、方法或服务[⊖]，并可以视为处理构件。

每个由对象封装的操作都提供了对象行为的一种表示。例如，Automobile 对象的 GetColor() 操作可以取出存储在 color 属性中的颜色。该操作的含义是：Automobile 类可接收一个刺激（我们称这个刺激为消息），该刺激请求类的特定实例的颜色。只要对象收到该刺激，它就会初始化一些行为。这样的行为可以简单到检索汽车的颜色，也可以复杂到初始化连接大量不同对象的刺激链。在后一种情况下，思考一个例子，由 Object1 对象收到的初始化刺激产生两个不同的刺激分别送往 Object2 和 Object3。由第二个和第三个对象封装的操作作用于刺激，返回必要的信息到第一个对象。Object1 然后使用返回信息以满足初始化刺激所要求的行为。

面向对象分析和设计概念

需求建模（也称分析建模）主要集中在由问题陈述中直接抽取的类上。这些实体类通常表示的是存储在数据库中并在应用程序持续的时间内一直存在的事物（除非特意将其删除）。

设计改进和扩展了实体类的集合。边界类和控制器类在设计期间得到了开发和改进。边界类创建用户可以看到的接口（例如，交互屏幕和输出报告），该接口可在使用软件时与用户进行交互。设计边界类是为了管理实体对象显示给用户的方式。

控制器类用来管理：（1）实体对象的建立或更新；（2）当边界对象从实体对象获取信息时边界对象的实例化；（3）对象集合之间的复杂通信；（4）对象之间或用户与应用系统之间通信数据的有效性。

以下段落讨论的概念在分析和设计工作中十分有用。

继承。继承是传统系统和面向对象系统之间的主要区别之一。子类 Y 继承其超类 X 的所有属性和操作，这意味着原来为 X 设计和实现的所有数据结构和算法都可以立即为 Y 所用——不需要做进一步的工作，重用可直接完成。

超类的属性或操作的任何改变都可以立即被所有子类继承。因此，类的层次结构成为一种机制，（在高层次上的）改变可以立即在系统中传播。

值得注意的是，在类继承的各个层次上，对于那些继承自更高层次的类，还可以增加新的属性和操作。实际上，每当创建一个新类时，你都可以有很多选择：

- 可以从头开始设计和构建类，也就是说，可以不使用继承。
- 可以对类的层次结构进行查找，以确定在较高层次的类中是否包含了大多数需要的属性和操作。新类可以根据需要继承较高层的类，然后进行增加。
- 可以对类的层次结构进行调整，使得所需要的属性和操作可以被新类继承。
- 已有类的特征可以被重写，对于新类，可以实现不同版本的属性或操作。

像所有基本的设计概念一样，继承可为设计带来显著好处。但是，若使用不恰当[⊜]，就

⊖ 在本段的上下文中使用了术语操作，但术语方法和服务同样流行。

⊜ 例如，设计子类继承多个超类的属性和操作（有时称为“多重继承”）会让大多数设计者费解。

会使设计呈现出不必要的复杂性，并导致难以维护和易于出错的软件。

消息。类之间必须彼此交互来完成设计目标。消息刺激接收对象产生某种行为，当操作执行时，完成该行为。

对象之间的相互作用由图 A2-2 做了概要描述。SenderObject 内的操作产生一条 message (<parameters>) 形式的消息，其中，参数（parameters）指定 ReceiverObject 作为消息刺激的对象、将要接收这条消息的 ReceiverObject 中的操作以及为了使操作获得成功需要提供信息的数据项。所定义的类之间的协作是分析模型的一部分，在消息设计中提供了十分有用的指导。

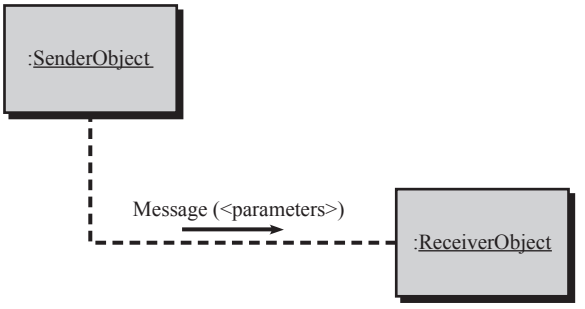

图 A2-2　对象之间的消息传递

Cox[Cox86] 以如下方式描述了类之间的交互：

通过给对象（类）发送消息告诉对象做什么来要求其执行某个操作。接收者（对象）通过选择对应消息名字的操作去执行，然后将控制权交还给调用者，从而响应消息。消息提供了单个对象和整体的面向对象系统行为的洞察力。

多态。多态是一种特性，这种特性可以显著减少扩展已存在的面向对象系统的设计所需要的工作量。为了理解多态，考虑一个传统的应用程序，此程序必须绘制 4 种不同类型的图：线图、饼图、直方图和 Kiviat 图。理想情况下，只要针对某种特殊类型的图收集数据，该图就可以自行绘制。要在传统程序中完成这项工作（并维护模块的内聚性），就有必要为每类图形都开发绘图模块。在设计中必须嵌入类似下面的控制逻辑：

```
case of graphtype:
    if graphtype = linegraph then DrawLineGraph (data);
    if graphtype = piechart then DrawPieChart (data);
    if graphtype = histogram then DrawHisto (data);
    if graphtype = kiviat then DrawKiviat (data);
end case;
```

虽然设计很简单，但是添加新图类型会很棘手。要为每种图创建新的绘图模块，然后还要修改控制逻辑以反映新的绘图类型。

为了解决这个问题，所有图形都成为一般类 Graph 的子类。使用重载的概念 [Tay90]，每个子类都定义 draw 操作。一个对象可以发送一条 draw 消息给由任意子类实例化的任意对象。接收消息的对象将会调用它本身的 draw 操作去创建适当的图，因此，将设计精简为：

```
draw <graphtype>
```

当系统中增加一种新的画图类型时，就会创建一个带有它独有的 draw 操作的子类。但是，在想要绘图的任何对象内不需要任何改变，因为消息 draw〈 graphtype 〉没有改变。总之，多态能够使很多不同的操作拥有相同的名字。反过来也使对象之间彼此分离，使每个对象更加独立。

设计类。需求模型定义了分析类的完整集合。每个分析类都描述了问题域的某种元素，集中在用户或客户可见的问题方面。分析类的抽象等级相对比较高。

随着设计模型的进展，软件团队必须定义一组设计类：（1）通过提供使类得以实现的设计细节对分析类进行细化；（2）创建一组新的设计类实现支持业务解决方案的软件基础结构。建议采用 5 种不同类型的设计类，每种设计类表示设计架构的下述不同层 [Amb01]。

- *用户界面类*定义所有人机交互（HCI）所必需的抽象。
- *业务域类*通常为早先定义的分析类的改进。类确定实现业务域的某些元素所需要的属性和服务（方法）。
- *处理类*执行需要管理业务域类的低层业务抽象。
- *持久类*表示持续时间超过软件执行时间的数据存储（如数据库）。
- *系统类*实现软件管理和控制功能，使系统能在内部计算环境和外部世界中操作和交流。

随着体系结构设计的进展，软件团队应当为每个设计类开发出完整的一组属性和操作。随着每个分析类转化为设计表示，抽象程度会降低。也就是说，分析类使用业务域术语表示对象（及应用于对象上的相关方法）。作为实现的指南，设计类在一定程度上给出了更多的技术细节。

Arlow 和 Neustadt[Arl02] 建议对每个设计类进行评审，以确保它是“组成良好”的。组成良好的设计类的 4 个特征如下。

完整性和充分性。设计类应当完整地封装期望可能存在于类中的所有属性和方法（基于对类名的合理解释）。例如，对于为视频编辑软件定义的 Scene 类，仅当它包括所有与创建视频场景有关的属性和方法时，才是完整的。充分性确保设计类仅包括那些能够完成类的意图的方法，不会多也不会少。

原始性。与设计类有关的方法应集中在完成类的一个特定功能上。一旦该功能由一个方法实现了，该类不应当再提供其他方法去完成同样的事情。例如，视频编辑软件的 VideoClip 类也许由 start-point 和 end-point 属性来表明剪辑的开始点和结束点（注意，载入到系统中的未加工视频也许会比剪辑过的要长一些）。方法 setStartPoint() 和 setEndPoint() 提供了唯一的手段来建立剪辑的开始点和结束点。

高内聚。一个内聚的设计类是专一的。也就是说，它有一组小的、集中的责任，并专一地应用属性和方法来完成这些责任。例如，视频编辑软件的 VideoClip 类也许包括一套方法用来编辑视频剪辑。只要每个方法单独集中于与视频剪辑有关的属性上，就维持了内聚性。

低耦合。在设计模型中，设计类彼此协作是有必要的。然而，协作应保持在一个可接受的最低限度。如果设计模型高度耦合（所有设计类都与所有的其他设计类协作），那么随着时间的推移，系统将很难实现、测试和维护。总之，在子系统中，设计类应该对其他类只有有限的了解。这个限制称为 Demeter 原则 [Lie03]，建议一个方法应该只能给相邻类的方法发送消息。⊖

扩展阅读与信息资源

在过去的 30 年里，有成百上千本书讲述面向对象的编程、分析和设计。Weisfeld（《The Object-Oriented Thought Process》，4th ed.，Addison-Wesley，2013）提供了对一般的面向对象概念和原

⊖ 叙述 Demeter 原则的一种不太正式的说法是“每个单元应当只与它的朋友交谈，不要与陌生人谈话”。

则的有用论述。Wong 和 Nguyen（《Principles of Object-Oriented Programming》，Amazon Digital Services，2011）在很多重要的面向对象概念方面提供了实践指导。McLaughlin 和他的同事（《Head First Object-Oriented Analysis and Design: A Brain Friendly Guide to OOA&D》，O'Reilly Media，2006）提供了面向对象分析和设计方法的可行的和有趣的论述。Keogh 和 Gianni（《OOP Demystified》，McGraw-Hill，2004）提供了该主题的有用指导。

Booch 和他的同事（《Object-Oriented Analysis and Design with Applications》，3rd ed., Addison-Wesley，2007）对面向对象分析和设计进行了更深入的讨论。在很多为不同编程语言编写的面向对象书籍中，有代表性的著作包括 Clark（《Beginning C# Object-Oriented Programming》，Apress，2013）、Kochen（《Programming in Objective-C》，5th ed., developer's Library，2012）、Phillips（《Python 3 Object-Oriented Programming》，Packt Publishing，2011）、Khurana（《Object-Oriented Programming with C++》，Vikas Publishing，2010）以及 Wu（《An Introducation to Object-Oriented Programming with Java》，2nd ed., McGraw-Hill，2009）。

网上有大量的关于面向对象技术的信息。最新的参考文献可以在 SEPA 网站 www.mhhe.com/pressman 的“analysis”和“design”下找到。

参 考 文献

[Abb83] Abbott, R., "Program Design by Informal English Descriptions, "*CACM*, vol. 26, no. 11, November 1983, pp. 892–894.

[Abr09] Abrial, J., "Faultless Systems: Yes We Can!" *IEEE Computer*, vol. 42, no. 9, September 2009, pp. 30–36.

[ACM12] ACM/IEEE-CS Joint Task Force, *Software Engineering Code of Ethics and Professional Practice*, 2012, available at www.acm.org/serving/se/code.htm.

[Ada93] Adams, D., *Mostly Harmless*, Macmillan, 1993.

[AFC88] *Software Risk Abatement*, AFCS/AFLC Pamphlet 800-45, U.S. Air Force, September 30, 1988.

[Agi03] The Agile Alliance Home Page, available at www.agilealliance.org/home.

[Air99] Airlie Council, "Performance Based Management: The Program Manager's Guide Based on the 16-Point Plan and Related Metrics," Draft Report, March 8, 1999.

[Aka04] Akao, Y., *Quality Function Deployment*, Productivity Press, 2004.

[Ale11] Alexander, I. "Gore, Sore, or What?" *IEEE Software*, vol. 28, no. 1, January–February 2011, pp. 8–10.

[Ale77] Alexander, C., *A Pattern Language*, Oxford University Press, 1977.

[Ale79] Alexander, C., *The Timeless Way of Building*, Oxford University Press, 1979.

[All08] Allen, J. H., et al., *Software Security Engineering: A Guide for Project Managers*, Addison-Wesley, 2008.

[Amb01] Ambler, S., *The Object Primer*, 2nd ed., Cambridge University Press, 2001.

[Amb02a] Ambler, S., "What Is Agile Modeling (AM)?" 2002, available at www.agilemodeling.com/index.htm.

[Amb02b] Ambler, S., and R. Jeffries, *Agile Modeling*, Wiley, 2002.

[Amb02c] Ambler, S., "UML Component Diagramming Guidelines," 2002, available at www.modelingstyle.info/.

[Amb04] Ambler, S., "*Examining the Cost of Change Curve*," in *The Object Primer*, 3rd ed., Cambridge University Press, 2004.

[Amb06] Ambler, S., "The Agile Unified Process (AUP), 2006, available at www.ambysoft.com/unifiedprocess/agileUP.html.

[Amb95] Ambler, S., "Using Use-Cases," *Software Development*, July 1995, pp. 53–61.

[Amb98] Ambler, S., *Process Patterns: Building Large-Scale Systems Using Object Technology*, Cambridge University Press/SIGS Books, 1998.

[And05] Andreou, A., et al., "Key Issues for the Design and Development of Mobile Commerce Services and Applications," *International Journal of Mobile Communications*, vol. 3, no. 3, March 2005, pp. 303–323.

And06] Andrews, M., and J. Whittaker, *How to Break Web Software: Functional and Security Testing of Web Applications and Web Services*, Addison-Wesley, 2006.

[ANS87] ANSI/ASQC A3-1987, *Quality Systems Terminology*, 1987.

[Ant06] Anton, D., and C. Anton, *ISO 9001 Survival Guide*, 3rd ed., AEM Consulting Group, 2006.

[AOS07] AOSD.net (Aspect-Oriented Software Development), glossary, available at http://aosd.net/wiki/index.php?title=Glossary.

[App00] Appleton, B., "Patterns and Software: Essential Concepts and Terminology," February 2000, available at www.cmcrossroads.com/bradapp/docs/patterns-intro.html.

[App13] Apple Computer, *Accessibility*, 2013, available at www.apple.com/accessibility/.

[Arl02] Arlow, J., and I. Neustadt, *UML and the Unified Process*, Addison-Wesley, 2002.

[Arn89] Arnold, R. S., "Software Restructuring," *Proceedings of the IEEE*, vol. 77, no. 4, April 1989, pp. 607–617.

[Art97] Arthur, L. J., "Quantum Improvements in Software System Quality," *CACM*, vol. 40, no. 6, June 1997, pp. 47–52.

[Ast04] Astels, D., *Test Driven Development: A Practical Guide*, Prentice Hall, 2004.

[ATK12] ATKearney, "A.T. Kearney Study of Global Wealth and Spending," 2012, available at http://www.atkearney.com/news-media/news-releases/news-release/-/asset_publisher/00OIL7Jc67KL/content/id/387464.

[Baa07] de Baar, B., "Project Risk Checklist," 2007, available at www.softwareprojects.org/project_riskmanagement_starting62.htm.

[Baa10] Baaz, A., et al., "Appreciating Lessons Learned," *IEEE Software*, vol. 27, no. 4, July–August, 2010, pp. 72–79.

[Bab09] Babar, M., and I. Groton, "Software Architecture Review: The State of Practice," *IEEE Computer*, vol. 42, no. 6, June 2009, pp. 1–8.

[Bab86] Babich, W. A., *Software Configuration Management*, Addison-Wesley, 1986.

[Bac97] Bach, J., "'Good Enough Quality: Beyond the Buzzword," *IEEE Computer*, vol. 30, no. 8, August 1997, pp. 96–98.

[Bac98] Bach, J., "The Highs and Lows of Change Control," *Computer*, vol. 31, no. 8, August 1998, pp. 113–115.

[Bae98] Baetjer, Jr., H., *Software as Capital*, IEEE Computer Society Press, 1998, p. 85.

[Bak72] Baker, F. T., "Chief Programmer Team Management of Production Programming," *IBM Systems Journal*., vol. 11, no. 1, 1972, pp. 56–73.

[Ban06a] Baniassad, E., et al., "Discovering Early Aspects," *IEEE Software*, vol. 23, no. 1, January–February, 2006, pp. 61–69.

[Bar06b] Baresi, L., E. DiNitto, and C. Ghezzi, "Toward Open-World Software: Issues and Challenges," *IEEE Computer*, vol. 39, no. 10, October 2006, pp. 36–43.

[Bas03] Bass, L., P. Clements, and R. Kazman, *Software Architecture in Practice*, 2nd ed., Addison-Wesley, 2003

[Bas84] Basili, V. R., and D. M. Weiss, "A Methodology for Collecting Valid Software Engineering Data," *IEEE Trans. Software Engineering*, vol. SE-10, 1984, pp. 728–738.

[Bea11] Beaird, J., The Principles of Beautiful Web Design, 2nd ed., Sitepoint, 2011.

[Bec00] Beck, K., *Extreme Programming Explained: Embrace Change*, Addison-Wesley, 1999.

[Bec01] Beck, K., et al., "Manifesto for Agile Software Development," www.agilemanifesto.org/.

[Bec04a] Beck, K., *Extreme Programming Explained: Embrace Change*, 2nd ed., Addison-Wesley, 2004.

[Bec04b] Beck, K., *Test-Driven Development: By Example*, 2nd ed., Addison-Wesley, 2004.

[Bee99] Beedle, M., et al., "SCRUM: An Extension Pattern Language for Hyperproductive Software Development," included in: *Pattern Languages of Program Design 4*, Addison-Wesley Longman, Reading MA, 1999, downloadable from http://jeffsutherland.com/scrum/scrum_plop.pdf.

[Beg10] Begel, A., R. DeLine, and T. Zimmermann, "Social Media for Software Engineering," *Proc. FoSER 2010*, ACM, November, 2010.

[Bei84] Beizer, B., *Software System Testing and Quality Assurance*, Van Nostrand-Reinhold, 1984.

[Bei90] Beizer, B., *Software Testing Techniques*, 2nd ed., Van Nostrand-Reinhold, 1990.

[Bei95] Beizer, B., *Black-Box Testing*, Wiley, 1995.

[Bel81] Belady, L., Foreword to *Software Design: Methods and Techniques* (L. J. Peters, author), Yourdon Press, 1981.

[Bel95] Bellinzona R., M. G. Gugini, and B. Pernici, "Reusing Specifications in OO Applications," *IEEE Software*, March 1995, pp. 65–75.

[Ben00] Bennatan, E., *Software Project Management: A Practitioner's Approach*, 3rd ed., McGraw-Hill, 2000.

[Ben99] Bentley, J., *Programming Pearls*, 2nd ed., Addison-Wesley, 1999.

[Ben02] Bennett, S., S. McRobb, and R. Farmer, *Object-Oriented Analysis and Design*, 2nd ed., McGraw-Hill, 2002.

[Ben10] Benaroch, M., and A. Appari, "Financial Pricing of Software Development Risk Facotrs," *IEEE Software*, vol. 27, no. 3, September–October, 2010, pp. 65–73.

[Ber80] Bersoff, E., V. Henderson, and S. Siegel, *Software Configuration Management*, Prentice Hall, 1980.

[Ber93] Berard, E., *Essays on Object-Oriented Software Engineering*, vol. 1, Addison-Wesley, 1993.

[Bes04] Bessin, J., "The Business Value of Quality," IBM developerWorks, June 15, 2004, downloadable from http://cm.techwell.com/articles/original/business-value-quality-and-testing.

[Bie94] Bieman, J. M., and L. M. Ott, "Measuring Functional Cohesion," *IEEE Trans. Software Engineering*, vol. SE-20, no. 8, August 1994, pp. 308–320.

[Bin93] Binder, R., "Design for Reuse Is for Real," *American Programmer*, vol. 6, no. 8, August 1993, pp. 30–37.

[Bin94a] Binder, R., "Testing Object-Oriented Systems: A Status Report," *American Programmer*, vol. 7, no. 4, April 1994, pp. 23–28.

[Bin94b] Binder, R. V., "Object-Oriented Software Testing," *Communications of the ACM*, vol. 37, no. 9, September 1994, p. 29.

[Bin99] Binder, R., *Testing Object-Oriented Systems: Models, Patterns, and Tools*, Addison-Wesley, 1999.

[Bir98] Biró, M., and T. Remzsö, "Business Motivations for Software Process Improvement," ERCIM News No. 32, January 1998, available at www.ercim.org/publication/Ercim_News/enw32/biro.html.

[Bla09] Black, S., et al. "Formal Versus Agile: Survival of the Fittest?" *IEEE Computer*, vol. 42, no. 9, September 2009, pp. 37–45.

[Bla10] Blair, S., et al., "Responsibility-Driven Architecture," *IEEE Software*, vol. 27, no. 3, March–April 2010, pp. 26–32.

[Bod09] Bode, S., et al., "Software Architectural Design Meets Security Engineering," *Proceedings of 16th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems*, 2009.

[Boe00] Boehm, B., et al., *Software Cost Estimation in COCOMO II*, Prentice Hall, 2000.

[Boe01a] Boehm, B., "The Spiral Model as a Tool for Evolutionary Software Acquisition," *CrossTalk*, May 2001, available at www.stsc.hill.af.mil/crosstalk/2001/05/boehm.html.

[Boe01b] Boehm, B., and V. Basili, "Software Defect Reduction Top 10 List," *IEEE Computer*, vol. 34, no. 1, January 2001, pp. 135–137.

[Boe08] Boehm, B., "Making a Difference in the Software Century," *IEEE Computer*, vol. 41, no. 3, March 2008, pp. 32–38.

[Boe81] Boehm, B., *Software Engineering Economics*, Prentice Hall, 1981.

[Boe88] Boehm, B., "A Spiral Model for Software Development and Enhancement," *Computer*, vol. 21, no. 5, May 1988, pp. 61–72.

[Boe89] Boehm, B. W., *Software Risk Management*, IEEE Computer Society Press, 1989.

[Boe96] Boehm, B., "Anchoring the Software Process," *IEEE Software*, vol. 13, no. 4, July 1996, pp. 73–82.

[Boe98] Boehm, B., "Using the WINWIN Spiral Model: A Case Study," *Computer*, vol. 31, no. 7, July 1998, pp. 33–44.

[Boh00] Bohl, M., and M. Rynn, *Tools for Structured Design: An Introduction to Programming Logic*, 5th ed., Prentice Hall, 2000.

[Boh66] Bohm, C., and G. Jacopini, "Flow Diagrams, Turing Machines and Languages with Only Two Formation Rules," *CACM*, vol. 9, no. 5, May 1966, pp. 366–371.

[Boi04] Boiko, B., *Content Management Bible*, 2nd ed., Wiley, 2004.

[Boo02] Booch, G., and A. Brown, "Collaborative Development Environments," Rational Software Corp., October 28, 2002.

[Boo05] Booch, G., J. Rumbaugh, and I. Jacobsen, *The Unified Modeling Language User Guide*. 2nd ed., Addison-Wesley, 2005.

[Boo06] Bootstrap-institute.com, 2006, www.cse.dcu.ie/espinode/directory/directory.html.

[Boo08] Booch, G., *Handbook of Software Architecture*, 2008, available at www.booch.com/architecture/systems.jsp.

[Boo11a] Booch, G., "Dominant Design," *IEEE Software*, vol. 28, no. 2, January–February 2011, pp. 8–9.

[Boo11b] Booch, G., "Draw Me a Picture," *IEEE Software*, vol. 28, no. 1, January–February 2011, pp. 6–7.

[Boo94] Booch, G., *Object-Oriented Analysis and Design*, 2nd ed., Benjamin Cummings, 1994.
[Bor01] Borchers, J., *A Pattern Approach to Interaction Design*, Wiley, 2001.
[Bos00] Bosch, J., *Design & Use of Software Architectures*, Addison-Wesley, 2000.
[Bos11] Bose, B., et al., "Morphing Smartphones into Automotive Application Platforms," *IEEE Computer*, vol. 44, no. 5, May 2011, pp. 28–29.
[Bra94] Bradac, M., D. Perry, and L. Votta, "Prototyping a Process Monitoring Experiment," *IEEE Trans. Software Engineering*, vol. 20, no. 10, October 1994, pp. 774–784.
[Bre02] Breen, P., "Exposing the Fallacy of 'Good Enough' Software," informit.com, February 1, 2002, available at www.informit.com/articles/article.asp?p=25141&rl=1.
[Bre03] Breu, R., et al., *Key Issues of a Formally Based Process Model for Security Engineering*, Proceedings of the 16th International Conference on Software & Systems Engineering and their Applications, *2003*.
[Bre10] Breu, R., *Ten Principles for Living Models—A Manifesto of Change-Driven Software Engineering*, International Conference on Complex, Intelligent and Software Intensive Systems (CISIS), Krakow, Poland, February 2010, pp. 1–8.
[Bro01] Brown, B., *Oracle9i Web Development*, 2nd ed., McGraw-Hill, 2001.
[Bro03] Brooks, F., "Three Great Challenges for Half-Century-Old Computer Science," *JACM*, vol. 50, no. 1, January 2003, pp. 25–26.
[Bro06] Broy, M., "The 'Grand Challenge' in Informatics: Engineering Software Intensive Systems," *IEEE Computer*, vol. 39, no. 10, October 2006, pp. 72–80.
[Bro10a] Brown, N., R. Nord, and I. Ozkaya, "Enabling Agility through Architecture," *Crosstalk*, November–December 2010, available at www.crosstalkonline.org/storage/issue-archives/…/201011-Brown.pdf.
[Bro10b] Broy, M., and R. Reussner, "Software Architecture Review: The State of Practice," *IEEE Computer*, vol. 43, no. 10, June 2010, pp. 88–91.
[Bro11] Brown, M., "The Best Tools for Mobile App Testing," available at http://www.mobileapptesting.com/the-best-tools-for-mobile-app-testing/2011/08/.
[Bro12] Brown, A., *The Architecture of Open Source Applications*, lulu.com, 2012.
[Bro95] Brooks, F., *The Mythical Man-Month,* Silver Anniversary edition, Addison-Wesley, 1995.
[Bro96] Brown, A., and K. Wallnau, "Engineering of Component Based Systems," *Component-Based Software Engineering*, IEEE Computer Society Press, 1996, pp. 7–15.
[Buc99] Bucanac, C., "The V-Model," University of Karlskrona/Ronneby, January 1999, downloadable from www.bucanac.com/documents/The_V-Model.pdf.
[Bud96] Budd, T., *An Introduction to Object-Oriented Programming*, 2nd ed., Addison-Wesley, 1996.
[Bus07] Buschmann, F., et al., *Pattern-Oriented Software Architecture, A System of Pattern*, Wiley, 2007.
[Bus10] Buschmann, F., "Learning from Failure, Part 2: Featuritis, Performitis, and Other Diseases," *IEEE Software*, vol. 27, no. 1, January–February 2010, pp. 10–11.
[Bus10a] Buschmann, F., "On Architecture Styles and Paradigms," *IEEE Software*, vol. 27, no. 5, September–October 2010, pp. 92–94.
[Bus10b] Buschmann, F., and K. Henley, "Five Considerations for Software Architecture, Part 1," *IEEE Software*, vol. 27, no. 3, May–June 2010, pp. 63–65.
[Bus10c] Buschmann, F., and K. Henley, "Five Considerations for Software Architecture, Part 2," *IEEE Software*, vol. 27, no. 4, July–August 2010, pp. 12–14.
[Bus96] Buschmann, F., et al., *Pattern-Oriented Software Architecture*, Wiley, 1996.
[Cac02] Cachero, C., et al., "Conceptual Navigation Analysis: a Device and Platform Independent Navigation Specification," *Proceedings of the Second International Workshop on Web-Oriented Technology*, June 2002, available at www.dsic.upv.es/~west/iwwost02/papers/cachero.pdf.
[Car08] Carrasco, M., "7 Key Attributes of High Performance Software Development Teams," June 30, 2008, available at http://www.realsoftwaredevelopment.com/7-key-attributes-of-high-performance-software-development-teams/.
[Car90] Card, D., and R. Glass, *Measuring Software Design Quality*, Prentice Hall, 1990.
[Cas06] Casey, V., and I. Richardson, "Uncovering the Reality within Virtual Software Teams," *Proc. GSD'06*, ACM, May 23, 2006.

[Cas89] Cashman, M., "Object Oriented Domain Analysis," *ACM Software Engineering Notes*, vol. 14, no. 6, October 1989, p. 67.

[Cav78] Cavano, J., and J. McCall, "A Framework for the Measurement of Software Quality," *Proc. ACM Software Quality Assurance Workshop*, November 1978, pp. 133–139.

[CCS02] CS3 Consulting Services, 2002, available at www.cs3inc.com/DSDM.htm.

[Cec06] Cechich, A., et al., "Trends on COTS Component Identification," *Proc. Fifth Intl. Conf. on COTS-Based Software Systems*, IEEE, 2006.

[Cha89] Charette, R., *Engineering Risk Analysis and Management*, McGraw-Hill/Intertext, 1989.

[Cha92] Charette, R., "Building Bridges over Intelligent Rivers," *American Programmer*, vol. 5, no. 7, September 1992, pp. 2–9.

[Cha93] de Champeaux, D., D. Lea, and P. Faure, *Object-Oriented System Development*, Addison-Wesley, 1993.

[Chi94] Chidamber, S., and C. Kemerer, "A Metrics Suite for Object-Oriented Design," *IEEE Trans. Software Engineering*, vol. SE-20, no. 6, June 1994, pp. 476–493.

[Cho89] Choi, S., and W. Scacchi, "Assuring the Correctness of a Configured Software Description," *Proceedings of the Second International Workshop on Software Configuration Management*, ACM, Princeton, NJ, October 1989, pp. 66–75.

[Chu09] Chung, L., and J. Leite, "On Non-Functional Requirements in Software Engineering," published in *Conceptual Modeling: Foundations and Applications*, Springer-Verlag, 2009.

[Chu95] Churcher, N., and M. Shepperd, "Towards a Conceptual Framework for Object-Oriented Metrics," *ACM Software Engineering Notes*, vol. 20, no. 2, April 1995, pp. 69–76.

[Cia09] Ciafrani, C., et al., *ISO 9000:2008 Explained*, 3rd ed., ASQ Quality Press, 2009.

[Cig07] Cigital, Inc., "Case Study: Finding Defects Earlier Yields Enormous Savings," 2007, available at www.cigital.com/solutions/roi-cs2.php.

[Cla05] Clark, S., and E. Baniasaad, *Aspect-Oriented Analysis and Design*, Addison-Wesley, 2005.

[Cle03] Clements, P., R. Kazman, and M. Klein, *Evaluating Software Architectures: Methods and Case Studies*, Addison-Wesley, 2003.

[Cle06] Clemmons, R., "Project Estimation with Use Case Points," *CrossTalk*, February 2006, pp. 18–222, available at www.stsc.hill.af.mil/crosstalk/2006/02/0602Clemmons.pdf.

[Cle10] Clements, P., and L. Bass, "The Business Goals Viewpoint," *IEEE Software*, vol. 27, no. 6, November–December 2010, pp. 38–45.

[CMM07] *Capability Maturity Model Integration (CMMI)*, Software Engineering Institute, 2007, available at www.sei.cmu.edu/cmmi/.

[CMM08] *People Capability Maturity Model Integration (People CMM)*, Software Engineering Institute, 2008, available at www.sei.cmu.edu/cmm-p/.

[Coa91] Coad, P., and E. Yourdon, *Object-Oriented Analysis*, 2nd ed., Prentice Hall, 1991.

[Coa99] Coad, P., E. Lefebvre, and J. DeLuca, *Java Modeling in Color with UML*, Prentice Hall, 1999.

[Coc01a] Cockburn, A., and J. Highsmith, "Agile Software Development: The People Factor," *IEEE Computer*, vol. 34, no. 11, November 2001, pp. 131–133.

[Coc01b] Cockburn, A., *Writing Effective Use-Cases*, Addison-Wesley, 2001.

[Coc02] Cockburn, A., *Agile Software Development*, Addison-Wesley, 2002.

[Coc04] Cockburn, A., "What the Agile Toolbox Contains," *CrossTalk*, November 2004, available at www.stsc.hill.af.mil/crosstalk/2004/11/0411Cockburn.html.

[Coc05] Cockburn, A., *Crystal Clear*, Addison-Wesley, 2005.

[Coc11] Cochran, C., *ISO 9001 in Plain English*, Paton Professional, 2011.

[Coh05] Cohn, M., "Estimating with Use Case Points," *Methods & Tools*, Fall, 2005, available at http://www.mountaingoatsoftware.com/articles/estimating-with-use-case-points.

[Col09] Collaris, R-A., and E. Dekker, "Software Cost Estimation Using Use Case Points, " IBM Developer Works, March 15, 2009, available at http://www.ibm.com/developerworks/rational/library/edge/09/mar09/collaris_dekker/.

[Con93] Constantine, L., "Work Organization: Paradigms for Project Management and Organization, *CACM*, vol. 36, no. 10, October 1993, pp. 34–43.

[Con95] Constantine, L., "What DO Users Want? Engineering Usability in Software," *Windows Tech Journal*, December 1995, available from www.forUse.com.

[Con96] Conradi, R., "Software Process Improvement: Why We Need SPIQ," NTNU, October 1996, downloadable from www.idi.ntnu.no/grupper/su/publ/pdf/nik96-spiq.pdf.

[Con99] Constantine, L., and L. Lockwood, "Learning the Laws of Usability," *Software Development*, vol. 7, no. 10, October, 1999.

[Con02] Conradi, R., and A. Fuggetta, "Improving Software Process Improvement," *IEEE Software*, July–August 2002, pp. 2–9, available at http://citeseer.ist.psu.edu/conradi02improving.html.

[Cop05] Coplien, J., "Software Patterns," 2005, available at http://hillside.net/patterns/definition.html.

[Cou00] Coulouris, G., J. Dollimore, and T. Kindberg, *Distributed Systems: Concepts and Design*, 3rd ed., Addison-Wesley, 2000.

[Cri92] Christel, M., and K. Kang, "Issues in Requirements Elicitation," Software Engineering Institute, CMU/SEI-92-TR-12 7, September 1992.

[Crn11] Crnkovic, I., et al., "A Classification Framework for Software Component Models," *IEEE Transactions on Software Engineering*, vol. 37, no. 5, September–October 2011, pp. 593–615.

[Cro79] Crosby, P., *Quality Is Free*, McGraw-Hill, 1979.

[Cur01] Curtis, B., W. Hefley, and S. Miller, *People Capability Maturity Model*, Addison-Wesley, 2001.

[Cur86] Curritt, P., M. Dyer, and H. Mills, "Certifying the Reliability of Software," *IEEE Trans, Software Engineering*, vol. SE-12, no. 1, January 1994.

[Cur90] Curtis, B., and D. Walz, "The Psychology of Programming in the Large: Team and Organizational Behavior," *Psychology of Programming*, Academic Press, 1990.

[CVS07] Concurrent Versions System, Ximbiot, 2007, available at http://ximbiot.com/cvs/wiki/index.php?title=Main_Page.

[CVS12] *Open CVS*, 2012, available at http://web.archive.org/web/20041220041434/http://www.opencvs.org/.

[DAC03] "An Overview of Model-Based Testing for Software," Data and Analysis Center for Software, CR/TA 12, June 2003, available from www.goldpractices.com/dwnload/practice/pdf/Model_Based_Testing.pdf.

[Dah72] Dahl, O., E. Dijkstra, and C. Hoare, *Structured Programming*, Academic Press, 1972.

[Dan09] Dang, A., "How Correct is Security by Correctness?" Tom's Hardware, July 16, 2009, available at http://www.tomshardware.com/reviews/joanna-rutkowska-rootkit,2356-7.html.

[Dar01] Dart, S., *Spectrum of Functionality in Configuration Management Systems*, Software Engineering Institute, 2001, available at www.sei.cmu.edu/legacy/scm/tech_rep/TR11_90/TOC_TR11_90.html.

[Dar91] Dart, S., "Concepts in Configuration Management Systems," *Proc. Third International Workshop on Software Configuration Management*, ACM SIGSOFT, 1991, available at www.sei.cmu.edu/legacy/scm/abstracts/abscm_concepts.html.

[Dar99] Dart, S., "Change Management: Containing the Web Crisis," *Proc. Software Configuration Management Symposium*, Toulouse, France, 1999, available at www.perforce.com/perforce/conf99/dart.html.

[Dav90] Davenport, T. H., and J. E. Young, "The New Industrial Engineering: Information Technology and Business Process Redesign," *Sloan Management Review*, Summer 1990, pp. 11–27.

[Dav93] Davis, A., et al., "Identifying and Measuring Quality in a Software Requirements Specification," *Proceedings of the First International Software Metrics Symposium*, IEEE, Baltimore, MD, May 1993, pp. 141–152.

[Dav95a] Davis, M., "Process and Product: Dichotomy or Duality," *Software Engineering Notes*, ACM Press, vol. 20, no. 2, April 1995, pp. 17–18.

[Dav95b] Davis, A., *201 Principles of Software Development*, McGraw-Hill, 1995.

[Day99] Dayani-Fard, H., et al., "Legacy Software Systems: Issues, Progress, and Challenges," IBM Technical Report: TR-74.165-k, April 1999, available at www.cas.ibm.com/toronto/publications/TR-74.165/k/legacy.html.

[DeM02] DeMarco, T., and B. Boehm, "The Agile Methods, Fray," *IEEE Computer*, vol. 35, no. 6, June 2002, pp. 90–92.
[DeM79] DeMarco, T., *Structured Analysis and System Specification*, Prentice Hall, 1979.
[DeM95] DeMarco, T., *Why Does Software Cost So Much?* Dorset House, 1995.
[DeM98] DeMarco, T., and T. Lister, *Peopleware*, 2nd ed., Dorset House, 1998.
[Dem86] Deming, W., *Out of the Crisis*, MIT Press, 1986.
[Den73] Dennis, J., "Modularity," in *Advanced Course on Software Engineering* (F. L. Bauer, ed.), Springer-Verlag, 1973, pp. 128–182.
[Des08] de Sá, M., and L. Carriço, "Lessons from Early Stages Design of Mobile Applications," *Proceedings of 10th International Conference on Human Computer Human with Mobile Services and Devices*, September 2008, pp. 127–136.
[deS09] de Sousa, C., et al, "Cooperative and Human Aspects of Software Engineering," *IEEE Software*, vol. 26, no. 6, pp. 17–19.
[Dev00] Devanbu, P., and S. Stubblebine, "Software Engineering for Security: A Roadmap," *Proc. ICSE*, IEEE, 2000, available at http://www0.cs.ucl.ac.uk/staff/A.Finkelstein/fose/finaldevanbu.pdf.
[Dev01] Devedzik, V., "Software Patterns," in *Handbook of Software Engineering and Knowledge Engineering*, World Scientific Publishing Co., 2001.
[Dha95] Dhama, H., "Quantitative Metrics for Cohesion and Coupling in Software," *Journal of Systems and Software*, vol. 29, no. 4, April 1995.
[Dij65] Dijkstra, E., "Programming Considered as a Human Activity," in *Proc. 1965 IFIP Congress*, North-Holland Publishing Co., 1965.
[Dij72] Dijkstra, E., "The Humble Programmer," 1972 ACM Turing Award Lecture, *CACM*, vol. 15, no. 10, October 1972, pp. 859–866.
[Dij76a] Dijkstra, E., "Structured Programming," in *Software Engineering, Concepts and Techniques* (J. Buxton et al., ed.), Van Nostrand-Reinhold, 1976.
[Dij76b] Dijkstra, E., *A Discipline of Programming*, Prentice Hall, 1976.
[Dij82] Dijksta, E., "On the Role of Scientific Thought," *Selected Writings on Computing: A Personal Perspective*, Springer-Verlag, 1982.
[Dix99] Dix, A., "Design of User Interfaces for the Web," *Proc. User Interfaces to Data Systems Conference*, September 1999, available at www.comp.lancs.ac.uk/computing/users/dixa/topics/webarch/.
[Don99] Donahue, G., S. Weinschenck, and J. Nowicki, "Usability Is Good Business," Compuware Corp., July 1999, available at www.compuware.com.
[Dre99] Dreilinger, S., "CVS Version Control for Web Site Projects," 1999, available at www.durak.org/cvswebsites/howto-cvs/howto-cvs.html.
[Dru75] Drucker, P., *Management*, W. H. Heinemann, 1975.
[DSi08] D'Silva, V., et al., "Interview: Software Security in the Real World," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 7, July 2008, pp. 1165–1178.
[Duc01] Ducatel, K., et al., *Scenarios for Ambient Intelligence in 2010*, ISTAG-European Commission, 2001, downloadable from ftp://ftp.cordis.europa.eu/pub/ist/docs/istagscenarios2010.pdf.
[Dun01] Dunaway, D., and S. Masters, *CMM-Based Appraisal for Internal Process Improvement (CBA IPI Version 1,2 Method Description)*, Software Engineering Institute, 2001, available at www.sei.cmu.edu/publications/documents/01.reports/01tr033.html.
[Dun02] Dunn, W., *Practical Design of Safety-Critical Computer Systems*, William Dunn, 2002.
[Dun82] Dunn, R., and R. Ullman, *Quality Assurance for Computer Software*, McGraw-Hill, 1982.
[Duy02] VanDuyne, D., J. Landay, and J. Hong, *The Design of Sites*, Addison-Wesley, 2002.
[Dye92] Dyer, M., *The Cleanroom Approach to Quality Software Development*, Wiley, 1992.
[Edg95] Edgemon, J., "Right Stuff: How to Recognize It When Selecting a Project Manager," *Application Development Trends*, vol. 2, no. 5, May 1995, pp. 37–42.
[Eis01] Eisenstein, J., et al., "Applying Model-Based Techniques to the Development of UIs for Mobile Computers," *Proceedings of Intelligent User Interfaces*, January 2001.
[Eji91] Ejiogu, L., *Software Engineering with Formal Metrics*, QED Publishing, 1991.

[Elr01] Elrad, T., R. Filman, and A. Bader (ed.), "Aspect Oriented Programming," *Comm. ACM*, vol. 44, no. 10, October 2001, special issue.

[Erd09] Ergomus, H., "The Seven Traits of Superprofessionals," *IEEE Software*, vol. 26, no. 4, July–August, 2009, pp. 4–6.

[Erd10] Ergogmus, H., "Déjà vu: The Life of Software Engineering Ideas," *IEEE Software*, vol. 27, no. 1, January–February 2010, pp. 2–3.

[Eri05] Ericson, C., *Hazard Analysis Techniques for System Safety*, Wiley-Interscience, 2005.

[Eri08] Erickson, T., *The Interaction Design Patterns Page*, May 2008, available at www.visi .com/~snowfall/InteractionPatterns.html.

[Eva04] Evans, E., *Domain Driven Design*, Addison-Wesley, 2004.

[Eve09] Everett, G., and B. Meyer, "Point/Counterpoint," *IEEE Software*, vol. 26, no. 4, July–August 2009, pp. 62–65.

[Fag86] Fagan, M., "Advances in Software Inspections," *IEEE Trans. Software Engineering*, vol. 12, no. 6, July 1986.

[Fal10] Falessi, D., et al., "Peaceful Coexistence: Agile Developer Perspectives on Software Architecture," *IEEE Software*, vol. 27, no. 3, March–April 2010, pp. 23–25.

[Fel07] Feller, J., et al. (eds.), *Perspectives on Free and Open Source Software*, The MIT Press, 2007.

[Fel89] Felican, L., and G. Zalateu, "Validating Halstead's Theory for Pascal Programs," *IEEE Trans. Software Engineering*, vol. SE-15, no. 2, December 1989, pp. 1630–1632.

[Fen91] Fenton, N., *Software Metrics*, Chapman and Hall, 1991.

[Fen94] Fenton, N., "Software Measurement: A Necessary Scientific Basis," *IEEE Trans. Software Engineering*, vol. SE-20, no. 3, March 1994, pp. 199–206.

[Fer00] Fernandez, E. B., and X. Yuan, "Semantic Analysis Patterns," *Proceedings of the 19th International Conerence on Conceptual Modeling, ER2000*, Lecture Notes in Computer Science 1920, Springer, 2000, pp. 183–195. Also available from www.cse.fau.edu/~ed/SAPpaper2.pdf.

[Fer97] Ferguson, P., et al., "Results of Applying the Personal Software Process," *IEEE Computer*, vol. 30, no. 5, May 1997, pp. 24–31.

[Fer98] Ferdinandi, P. L., "Facilitating Communication," *IEEE Software*, September 1998, pp. 92–96.

[Fil05] Filman, R., et al., *Aspect-oriented Software Development*, Addison-Wesley, 2005.

[Fir12] Firesmith, D., *Security and Safety Requirements for Software-Intensive Systems*, Auerbach, 2012.

[Fir93] Firesmith, D. G., *Object-Oriented Requirements Analysis and Logical Design*, Wiley, 1993.

[Fis11] Fisher, R., et al., *Getting to Yes: Negotiating without Giving In*, Penguin Books, 2011.

[Fle98] Fleming, Q., and J. Koppelman, "Earned Value Project Management," *CrossTalk*, vol. 11, no. 7, July 1998, p. 19.

[Fok10] Fokaefs, M., et al., WikiDev 2.0: Facilitating Software Development Teams, *Proceedings of the 14th European Conference on Software Maintenance and Reengineering*, March 15–18, 2010, pp. 276–277.

[Fos06] Foster, E., "Quality Culprits," InfoWorld Grip Line Weblog, May 2, 2006, available at http://weblog.infoworld.com/gripeline/2006/05/02_a395.html.

[Fow00] Fowler, M., et al., *Refactoring: Improving the Design of Existing Code*, Addison-Wesley, 2000.

[Fow01] Fowler, M., and J. Highsmith, "The Agile Manifesto," *Software Development Magazine*, August 2001, available at www.sdmagazine.com/documents/s=844/sdm0108a/0108a.htm.

[Fow02] Fowler, M., "The New Methodology," June 2002, available at www.martinfowler .com/articles/newMethodology.html#N8B.

[Fow03] Fowler, M., et al., *Patterns of Enterprise Application Architecture*, Addison-Wesley, 2003.

[Fow04] Fowler, M., *UML Distilled*, 3rd ed., Addison-Wesley, 2004.

[Fow97] Fowler, M., *Analysis Patterns: Reusable Object Models*, Addison-Wesley, 1997.

[Fra93] Frankl, P., and S. Weiss, "An Experimental Comparison of the Effectiveness of Branch Testing and Data Flow," *IEEE Trans. Software Engineering*, vol. SE-19, no. 8, August 1993, pp. 770–787.

[Fre80] Freeman, P., "The Context of Design," in *Software Design Techniques*, 3rd ed. (P. Freeman and A. Wasserman, eds.), IEEE Computer Society Press, 1980, pp. 2–4.

[Fre90] Freedman, D., and G. Weinberg, *Handbook of Walkthroughs, Inspections and Technical Reviews*, 3rd ed., Dorset House, 1990.

[Fri10] Fricker, S., "Handshaking with Implementation Proposals: Negotiating Requirements Understanding," *IEEE Software*, vol. 27, no. 2, March–April 2010, pp. 72–80.

[Gag04] Gage, D., and J. McCormick, "We Did Nothing Wrong," *Baseline Magazine*, March 4, 2004, available at www.baselinemag.com/article2/0,1397,1544403,00.asp.

[Gai95] Gaines, B., "Modeling and Forecasting the Information Sciences," Technical Report, University of Calgary, Calgary, Alberta, September 1995.

[Gam95] Gamma, E., et al., *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.

[Gar08] GartnerGroup, "Understanding Hype Cycles," 2008, available at www.gartner.com/pages/story.php.id.8795.s.8.jsp.

[Gar09] Gardner, D., "Can Software Development Aspire to the Cloud?" *ZDNet.com*, April 28, 2009, available at http://www.zdnet.com/blog/gardner/can-software-development-aspire-to-the-cloud/2915.

[Gar09a] Garlan, D., et al., "Architectural Mismatch: Way Reuse is Still So Hard," *IEEE Software*, vol. 26, no. 4, July–August 2009, pp. 66–69.

[Gar10] Garcia-Crespo, A. et al., "A Qualitative Study of Hard Decision Making in Managing Global Software Development Teams," *Journal of Management Information Systems*, vol. 27, no. 3, June 2010, pp. 247–252.

[Gar12] GartnerGroup, "Gartner Says Worldwide Media Tablets Sales to Reach 119 Million Units in 2012," April 2012, available at http://www.gartner.com/it/page.jsp?id=1980115.

[Gar84] Garvin, D., "What Does 'Product Quality' Really Mean?" *Sloan Management Review*, Fall 1984, pp. 25–45.

[Gar87] Garvin D., "Competing on the Eight Dimensions of Quality," *Harvard Business Review*, November 1987, pp. 101–109. A summary is available at www.acm.org/crossroads/xrds6-4/software.html.

[Gar95] Garlan, D., and M. Shaw, "An Introduction to Software Architecture," *Advances in Software Engineering and Knowledge Engineering*, vol. I (V. Ambriola and G. Tortora, eds.), World Scientific Publishing Company, 1995.

[Gau89] Gause, D., and G. Weinberg, *Exploring Requirements: Quality Before Design*, Dorset House, 1989.

[Gav11] Gavalas, D., and D. Economou, "Development Platforms for Mobile Applications," *IEEE Software*, vol. 28, no. 1, January–February, 2011, pp. 77–86.

[Gey01] Geyer-Schulz, A., and M. Hahsler, "Software Engineering with Analysis Patterns," Technical Report 01/2001, Institut für Informationsverarbeitung und -wirtschaft, Wirschaftsuniversität Wien, November 2001, downloadable from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.70.8415&rep=rep1&type=pdf.

[Gho01] Ghoshm, A., and Swaminatha, T. "Software Security and Privacy Risks in Mobile E-Commerce," *Communication of the ACM*, vol. 44, no. 2, February 2001, pp. 51–57.

[Gil06] Gillis, D., "Pattern-Based Design," tehan + lax blog, September 14, 2006, available at www.teehanlax.com/blog/?p=96.

[Gil88] Gilb, T., *Principles of Software Project Management*, Addison-Wesley, 1988.

[Gil95] Gilb, T., "What We Fail to Do in Our Current Testing Culture," *Testing Techniques Newsletter* (online edition, ttn@soft.com), Software Research, January 1995.

[Gla02] Gladwell, M., *The Tipping Point*, Back Bay Books, 2002.

[Gla98] Glass, R., "Defining Quality Intuitively," *IEEE Software*, May 1998, pp. 103–104, 107.

[Gli07] Glinz, M., and R. Wieringa, "Stakeholders in Requirements Engineering," *IEEE Software*, vol. 24, no. 2, March–April 2007, pp. 18–20.

[Gli09] Glinz, M., "A Risk-Base, Value-Oriented Approach to Quality Requirements," *IEEE Software*, vol. 26, no. 5, March–April 2009, pp. 34–41.

[Glu94] Gluch, D., "A Construct for Describing Software Development Risks," CMU/SEI-94-TR-14, Software Engineering Institute, 1994.

[Gna99] Gnaho, C., and F. Larcher, "A User-Centered Methodology for Complex and Customizable Web Engineering," *Proceedings of the First ICSE Workshop on Web Engineering*, ACM, Los Angeles, May 1999.

[Gon04] Gonzales, R., "Requirements Engineering," Sandia National Laboratories, a slide presentation, available at www.incose.org/enchantment/docs/04AprRequirementsEngineering.pdf.

[Gor06] Gorton, I., *Essential Software Architecture*, Springer, 2006.

[Got11] Gotel, O., and S. Morris, "Requirements Tracery," *IEEE Software*, vol. 28, no. 5, September–October 2011, pp. 92–94.

[Gra03] Gradecki, J., and N. Lesiecki, *Mastering AspectJ: Aspect-Oriented Programming in Java*, Wiley, 2003.

[Gra87] Grady, R. B., and D. L. Caswell, *Software Metrics: Establishing a Company-Wide Program*, Prentice Hall, 1987.

[Gra92] Grady, R. G., *Practical Software Metrics for Project Management and Process Improvement*, Prentice Hall, 1992.

[Gru00] Gruia-Catalin, R., et al., "Software Engineering for Mobility: A Roadmap," *Proceedings of the 22nd International Conference on the Future of Software Engineering*, 2000.

[Gru02] Grundy, J., "Aspect-Oriented Component Engineering," 2002, www.cs.auckland.ac.nz/~john-g/aspects.html.

[Gub09] Gube, J., "40+ Helpful Resources on User Interface Design Patterns," June 15, 2009, http://www.smashingmagazine.com/2009/06/15/40-helpful-resources-on-user-interface-design-patterns/.

[Gus89] Gustavsson, A., "Maintaining the Evolution of Software Objects in an Integrated Environment," *Proceedings of the Second International Workshop on Software Configuration Management*, ACM, Princeton, NJ, October 1989, pp. 114–117.

[Gut93] Guttag, J., and J. Horning, *Larch: Languages and Tools for Formal Specification*, Springer-Verlag, 1993.

[Hac98] Hackos, J., and J. Redish, *User and Task Analysis for Interface Design*, Wiley, 1998.

[Hai02] Hailpern, B., and P. Santhanam, "Software Debugging, Testing and Verification," *IBM Systems Journal*, vol. 41, no. 1, 2002, available at www.research.ibm.com/journal/sj/411/hailpern.html.

[Hal77] Halstead, M., *Elements of Software Science*, North-Holland, 1977.

[Hal90] Hall, A., "Seven Myths of Formal Methods," *IEEE Software*, September 1990, pp. 11–20.

[Hal98] Hall, E. M., *Managing Risk: Methods for Software Systems Development*, Addison-Wesley, 1998.

[Ham90] Hammer, M., "Reengineer Work: Don't Automate, Obliterate," *Harvard Business Review*, July–August 1990, pp. 104–112.

[Han95] Hanna, M., "Farewell to Waterfalls," *Software Magazine*, May 1995, pp. 38–46.

[Har11] Harris, N., and P. Avgeriou, "Pattern-Based Architecture Reviews," *IEEE Software*, vol. 28, no. 6, November–December 2011, pp. 66–71.

[Har12] Hardy, T., *Software and System Safety*, Authorhouse, 2012.

[Har98b] Harrison, R., S. Counsell, and R. Nithi, "An Evaluation of the MOOD Set of Object-Oriented Software Metrics," *IEEE Trans. Software Engineering*, vol. SE-24, no. 6, June 1998, pp. 491–496.

[Hen10] Hendler, J., "Web 3.0: The Dawn of Semantic Search," *IEEE Computer*, January, 2010, p. 77.

[Her00] Herrmann, D., *Software Safety and Reliability*, Wiley-IEEE Computer Society Press, 2000.

[Het84] Hetzel, W., *The Complete Guide to Software Testing*, QED Information Sciences, 1984.

[Het93] Hetzel, W., *Making Software Measurement Work*, QED Publishing, 1993.

[Hev93] Hevner, A., and H. Mills, "Box Structure Methods for System Development with Objects," *IBM Systems Journal*, vol. 31, no. 2, February 1993, pp. 232–251.

[Hig01] Highsmith, J. (ed.), "The Great Methodologies Debate: Part 1," *Cutter IT Journal*., vol. 14, no. 12, December 2001.

[Hig02a] Highsmith, J. (ed.), "The Great Methodologies Debate: Part 2," *Cutter IT Journal*, vol. 15, no. 1, January 2002.

[Hig95] Higuera, R., "Team Risk Management," *CrossTalk*, U.S. Dept. of Defense, January 1995, pp. 2–4.

[Hil05] Hildreth, S., "Buggy Software: Up from a Low Quality Quagmire," *Computerworld*, July 25, 2005, available at www.computerworld.com/developmenttopics/development/story/0,10801,103378,00.html.

[Hil13] Hillside.net, *Patterns Catalog*, 2013, available at http://hillside.net/patterns/patterns-catalog.

[Hne11] Hneif, M., and S. Lee, "Using Guidelines to Improve Quality in Software Nonfunctional Attributes," *IEEE Software*, vol. 28, no. 5, November–December 2011, pp. 72–73.

[Hof00] Hofmeister, C., R. Nord, and D. Soni, *Applied Software Architecture*, Addison-Wesley, 2000.

[Hof01] Hofmann, C., et al., "Approaches to Software Architecture," 2001, downloadable from http://citeseer.nj.nec.com/84015.html.

[Hol06] Holzner, S., *Design Patterns for Dummies*, For Dummies Publishers, 2006.

[Hoo96] Hooker, D., "Seven Principles of Software Development," September 1996, available at http://c2.com/cgi/wikiSevenPrinciplesOfSoftwareDevelopment.

[Hoo12] Hoober, S., and E. Berkman, *Designing Mobile Interfaces*, O'Reilly Media, 2012.

[Hop90] Hopper, M., "Rattling SABRE, New Ways to Compete on Information," *Harvard Business Review*, May–June 1990.

[Hor03] Horch, J., *Practical Guide to Software Quality Management*, 2nd ed., Artech House, 2003.

[Hoy09] Hoyle, D., *ISO 9000 Quality Systems Handbook*, 6th ed., Taylor & Francis, 2009.

[Hum00] Humphrey, W., *Introduction to the Team Software Process*, Addison-Wesley, 2000.

[Hum05] Humphrey, W., *A Self-Improvement Process for Software Engineers*, Addison-Wesley, 2005.

[Hum95] Humphrey, W., *A Discipline for Software Engineering*, Addison-Wesley, 1995.

[Hum96] Humphrey, W., "Using a Defined and Measured Personal Software Process," *IEEE Software*, vol. 13, no. 3, May–June 1996, pp. 77–88.

[Hum97] Humphrey, W., *Introduction to the Personal Software Process*, Addison-Wesley, 1997.

[Hum98] Humphrey, W., "The Three Dimensions of Process Improvement, Part III: The Team Process," *CrossTalk*, April 1998, available at www.stsc.hill.af.mil/crosstalk/1998/apr/dimensions.asp.

[Hun99] Hunt, A., D. Thomas, and W. Cunningham, *The Pragmatic Programmer*, Addison-Wesley, 1999.

[Hya96] Hyatt, L., and L. Rosenberg, "A Software Quality Model and Metrics for Identifying Project Risks and Assessing Software Quality," NASA SATC, 1996, available at http://satc.gsfc.nasa.gov/support/STC_APR96/qualtiy/stc_qual.html.

[IBM13] IBM, *Web Services Globalization Model*, 2013, available at http://www.ibm.com/developerworks/webservices/library/ws-global/.

[IBM81] "Implementing Software Inspections," course notes, IBM Systems Sciences Institute, IBM Corporation, 1981.

[IEE93a] *IEEE Standards Collection: Software Engineering*, IEEE Standard 610.12-1990, IEEE, 1993.

[IEE93b] *IEEE Standard Glossary of Software Engineering Terminology*, IEEE, 1993.

[IEE00] IEEE Standard Association, IEEE-Std-1471-2000, *Recommended Practice for Architectural Description of Software-Intensive Systems*, 2000, available at http://standards.ieee.org/reading/ieee/std_public/description/se/1471-2000_desc.html.

[IEE05] IEEE Std. 982.1-2005, *IEEE Standard Dictionary of Measures of the Software Aspects of Dependability*, 2005, available at http://standards.ieee.org/findstds/standard/982.1-2005.html.

[IEE09] *IEEE Software* (special issue), "Human Aspects of Software Engineering," vol. 28, no 6, November–December, 2009.

[IFP01] *Function Point Counting Practices Manual*, 2001, downloadable from: perun.pmf.uns.ac.rs/old/repository/…/se/functionpoints.pdf.

[IFP12] *Function Point Bibliography/Reference Library*, International Function Point Users Group, 2012, available from http://www.ifpug.org/?page_id=237.

[Isk08] Iskold, A., "Top Ten Concepts that Every Software Engineer Should Know," ReadWrite, July 2008, available from http://readwrite.com/2008/07/22/top_10_concepts_that_every_software_engineer_should_know.

[ISO00] *ISO 9001: 2000 Document Set*, International Organization for Standards, 2000, available at www.iso.ch/iso/en/iso9000-14000/iso9000/iso9000index.html.

[ISO02] *Z Formal Specification Notation—Syntax, Type System and Semantics*, ISO/IEC 13568:2002, Intl. Standards Organization, 2002.

[ISO08] ISO SPICE, 2008, available at www.isospice.com/categories/SPICE-Project/.

[Ivo01] Ivory, M., R. Sinha, and M. Hearst, " Empirically Validated Web Page Design Metrics," ACM *SIGCHI'01*, March 31–April 4, 2001, available at http://webtango.berkeley.edu/papers/chi2001/.

[Jac02a] Jacobson, I., "A Resounding 'Yes' to Agile Processes—But Also More," *Cutter IT Journal*, vol. 15, no. 1, January 2002, pp. 18–24.

[Jac02b] Jacyntho, D., D. Schwabe, and G. Rossi, "An Architecture for Structuring Complex Web Applications," 2002, available at www2002.org/CDROM/alternate/478/.

[Jac04] Jacobson, I., and P. Ng, *Aspect-Oriented Software Development*, Addison-Wesley, 2004.

[Jac75] Jackson, M. A., *Principles of Program Design*, Academic Press, 1975.

[Jac92] Jacobson, I., *Object-Oriented Software Engineering*, Addison-Wesley, 1992.

[Jac98] Jackman, M., "Homeopathic Remedies for Team Toxicity," *IEEE Software*, July 1998, pp. 43–45.

[Jac99] Jacobson, I., G. Booch, and J. Rumbaugh, *The Unified Software Development Process*, Addison-Wesley, 1999.

[Jal04] Jalote, P., et al., "Timeboxing: A Process Model for Iterative Software Development," *Journal of Systems and Software*, vol. 70, issue 2, 2004, pp. 117–127. Available at www.cse.iitk.ac.in/users/jalote/papers/Timeboxing.pdf.

[Jec06] Jech, T., *Set Theory*, 3rd ed., Springer, 2006.

[Jon04] Jones, C., "Software Project Management Practices: Failure Versus Success," *CrossTalk*, October 2004. Available at www.stsc.hill.af.mil/crossTalk/2004/10/0410Jones.html.

[Jon86] Jones, C., *Programming Productivity*, McGraw-Hill, 1986.

[Jon91] Jones, C., *Systematic Software Development Using VDM*, 2nd ed., Prentice Hall, 1991.

[Jon96] Jones, C., "How Software Estimation Tools Work," *American Programmer*, vol. 9, no. 7, July 1996, pp. 19–27.

[Jon98] Jones, C., *Estimating Software Costs*, McGraw-Hill, 1998.

[Joy00] Joy, B., "The Future Doesn't Need Us," *Wired*, vol. 8, no. 4, April 2000.

[Kai02] Kaiser, J., "Elements of Effective Web Design," *About, Inc.*, 2002, available at http://webdesign.about.com/library/weekly/aa091998.htm.

[Kan01] Kaner, C., "Pattern: Scenario Testing" (draft), 2001, available at www.testing.com/test-patterns/patterns/pattern-scenario-testing-kaner.html.

[Kan93] Kaner, C., J. Falk, and H. Q. Nguyen, *Testing Computer Software*, 2nd ed., Van Nostrand-Reinhold, 1993.

[Kan95] Kaner, C., "Lawyers, Lawsuits, and Quality Related Costs," 1995, available at www.badsoftware.com/plaintif.htm.

[Kanll] Kannan, N., "Mobile Testing: Nine Strategy Tests You'll Want to Perform," 2011, available at http://searchsoftwarequality.techtarget.com/tip/Mobile-testing-Nine-strategy-tests-youll-want-to-perform.

[Kar94] Karten, N., *Managing Expectations*, Dorset House, 1994.

[Kau95] Kauffman, S., *At Home in the Universe*, Oxford, 1995.

[Kaz03] Kazman, R., and A. Eden, "Defining the Terms Architecture, Design, and Implementation," *news@sei interactive*, Software Engineering Institute, vol. 6, no. 1, 2003, available at www.sei.cmu.edu/news-at-sei/columns/the_architect/2003/1q03/architect-1q03.htm.

[Kaz98] Kazman, R., et al., *The Architectural Tradeoff Analysis Method*, Software Engineering Institute, CMU/SEI-98-TR-008, July 1998, summarized at http://www.sei.cmu.edu/architecture/tools/evaluate/atam.cfm.

[Kea07] Keane, "Testing Mobile Business Applications," a white paper, 2007, available at www.keane.com.

[Kei98] Keil, M., et al., "A Framework for Identifying Software Project Risks," *CACM*, vol. 41, no. 11, November 1998, pp. 76–83.

[Kel00] Kelly, D., and R. Oshana, "Improving Software Quality Using Statistical Techniques, Information and Software Technology," *Elsevier*, vol. 42, August 2000, pp. 801–807, available at www.eng.auburn.edu/~kchang/comp6710/readings/Improving_Quality_with_Statistical_Testing_InfoSoftTech_August2000.pdf.

[Ker05] Kerievsky, J., *Industrial XP: Making XP Work in Large Organizations*, Cutter Consortium, Executive Report, vol. 6., no. 2, 2005, available at www.cutter.com/content-and-analysis/resource-centers/agile-project-management/sample-our-research/apmr0502.html.

[Ker11] Kershbaum, F., et al., "Secure Collaborative Supply-Chain Management," *IEEE Computer*, vol. 44, no. 9, September 2011, pp. 38–43.

[Ker78] Kernighan, B., and P. Plauger, *The Elements of Programming Style*, 2nd ed., McGraw-Hill, 1978.

[Kho12a] Khode, A., "Getting Started with Mobile Apps Testing," 2012, available at http://www.mobileappstesting.com/getting-started-with-mobile-apps-testing/.

[Kho12b] Khode, A., "Checklist for Mobile Test Automation Tools," 2012, available at http://www.mobileappstesting.com/getting-started-with-mobile-apps-testing/

[Kir94] Kirani, S., and W. Tsai, "Specification and Verification of Object-Oriented Programs," Technical Report TR 94-64, Computer Science Department, University of Minnesota, December 1994.

[Kiz05] Kizza, J., *Computer Network Security*, Springer, 2005.

[Knu98] Knuth, D., *The Art of Computer Programming*, three volumes, Addison-Wesley, 1998.

[Koe12] Koester, J., "The Seven Deadly Sins of MobileApp Design," *Venture Beat/Mobile*, May31,2012,availableathttp://venturebeat.com/2012/05/31/the-7-deadly-sins-of-mobile-app-design/

[Kon02] Konrad, S., and B. Cheng, "Requirements Patterns for Embedded Systems," *Proceedings of the 10th Anniversary IEEE Joint International Conference on Requirements Engineering*, IEEE, September 2002, pp. 127–136, available at http://citeseer.ist.psu.edu/669258.html.

[Kor03] Korpipaa, P., et al., "Managing Context Information in Mobile Devices," *IEEE Pervasive Computing*, vol. 2, no. 3, July–September 2003, pp. 42–51.

[Kra88] Krasner, G., and S. Pope, "A Cookbook for Using the Model-View Controller User Interface Paradigm in Smalltalk-80," *Journal of Object-Oriented Programming*, vol. 1, no. 3, August–September 1988, pp. 26–49.

[Kra95] Kraul, R., and L. Streeter, "Coordination in Software Development," *CACM*, vol. 38, no. 3, March 1995, pp. 69–81.

[Kru05] Krutchen, P., "Software Design in a Postmodern Era," *IEEE Software*, vol. 22, no. 2, March–April 2005, pp. 16–18.

[Kru06] Kruchten, P., H. Obbink, and J. Stafford (eds.), "Software Architectural" (special issue), *IEEE Software*, vol. 23, no. 2, March–April 2006.

[Kru09] Kruchten, P., et al. "The Decision View's Role in Software Architecture Practice," *IEEE Software*, vol. 26, no. 2, March–April 2009, pp. 70–72.

[Kur05] Kurzweil, R., *The Singularity Is Near*, Penguin Books, 2005.

[Kur13] Kurzweil, R., *How to Create a Mind*, Viking, 2013.

[Kyb84] Kyburg, H., *Theory and Measurement*, Cambridge University Press, 1984.

[Laa00] Laakso, S., et al., "Improved Scroll Bars," *CHI 2000 Conf. Proc.*, ACM, 2000, pp. 97–98, available at www.cs.helsinki.fi/u/salaakso/patterns/.

[Lag10] Lago, P., et al., "Software Architecture: Framing Stakeholders' Concerns," *IEEE Software*, vol. 27, no. 6, November–December 2010, pp. 20–24.

[Lai02] Laitenberger, A., "A Survey of Software Inspection Technologies," in *Handbook on Software Engineering and Knowledge Engineering*, World Scientific Publishing Company, 2002.

[Lam01a] Lam, W., "Testing E-Commerce Systems: A Practical Guide," *IEEE IT Pro*, March–April 2001, pp. 19–28.

[Lam01b] Lamsweerde, A. "Goal-Oriented Requirements Engineering: A Guided Tour," *Proceedings of 5th IEEE International Symposium on Requirements Engineering*, Toronto, August 2009, pp. 249–263.

[Lan01] Lange, M., "It's Testing Time! Patterns for Testing Software," June 2001, available at www.testing.com/test-patterns/patterns/index.html.

[Lan02] Land, R., "A Brief Survey of Software Architecture," technical report, Dept. of Computer Engineering, Mälardalen University, Sweden, February 2002.

[Lan10] Lanubile, F., C. Ebert, R. Prikladnicki, and A. Vizcaino, "Collaboration Tools for Global Software Engineering," *IEEE Software*, vol. 27, 2010, pp. 52–55.

[Laz11] Lazzaroni, M., et al., *Reliability Engineering*, Springer, 2011.

[Leh97a] Lehman, M., and L. Belady, *Program Evolution: Processes of Software Change*, Academic Press, 1997.

[Leh97b] Lehman, M., et al., "Metrics and Laws of Software Evolution—The Nineties View," *Proceedings of the 4th International Software Metrics Symposium (METRICS '97)*, IEEE, 1997, available at www.ece.utexas.edu/~perry/work/papers/feast1.pdf.

[Let01] Lethbridge, T., and R. Laganiere, *Object-Oriented Software Engineering: Practical Software Development Using UML and Java*, McGraw-Hill, 2001.

[Let03a] Lethbridge, T., Personal communication on domain analysis, May 2003.

[Let03b] Lethbridge, T., Personal communication on software metrics, June 2003.

[Lev01] Levinson, M., "Let's Stop Wasting $78 billion a Year," *CIO Magazine*, October 15, 2001, available at www.cio.com/archive/101501/wasting.html.

[Lev95] Leveson, N., *Safeware: System Safety and Computers*, Addison-Wesley, 1995.

[Lew09] Lewicki, R., B. Barry, and D. Saunders, *Negotiation*, McGraw-Hill, 2009.

[Lie03] Lieberherr, K., "Demeter: Aspect-Oriented Programming," May 2003, available at www.ccs.neu.edu/home/lieber/LoD.html.

[Lin79] Linger, R., H. Mills, and B. Witt, *Structured Programming*, Addison-Wesley, 1979.

[Lin88] Linger, R., and H. Mills, "A Case Study in Cleanroom Software Engineering: The IBM COBOL Structuring Facility," *Proc. COMPSAC '88*, Chicago, October 1988.

[Lin94] Linger, R., "Cleanroom Process Model," *IEEE Software*, vol. 11, no. 2, March 1994, pp. 50–58.

[Lip10] Lipner, S., "The Security Development Life Cycle," June 24, 2010, available at https://www.owasp.org/images/7/78/OWASP_AppSec_Research_2010_Keynote_2_by_Lipner.pdf.

[Lis88] Liskov, B., "Data Abstraction and Hierarchy," *SIGPLAN Notices*, vol. 23, no. 5, May 1988.

[Liu98] Liu, K., et al., "Report on the First SEBPC Workshop on Legacy Systems," Durham University, February 1998, available at www.dur.ac.uk/CSM/SABA/legacy-wksp1/report.html.

[Lon02] Longstreet, D., "Fundamental of Function Point Analysis," Longstreet Consulting, Inc., 2002, available at www.ifpug.com/fpafund.htm.

[Lor94] Lorenz, M., and J. Kidd, *Object-Oriented Software Metrics*, Prentice Hall, 1994.

[Lup08] Lupton, E., and J. Cole, *Graphic Design: The New Basics*, Princeton Architectural Press, 2008.

[Maa07] Maassen, O., and S. Stelting, "Creational Patterns: Creating Objects in an OO System," 2007, available at www.informit.com/articles/article.asp?p=26452&rl=1.

[Mad10] Madison, J., "Agile-Architecture Interactions," *IEEE Software*, vol. 27, no. 3, March–April 2010, pp. 41–48.

[Mai10a] Maiden, N., and S. Jones, "Agile Requirements—Can We Have Our Cake and Eat It Too?" *IEEE Software*, vol. 27, no. 3, May–June 2010, pp. 20–24.

[Mai10b] Maiden, N., "Service Design: It's All in the Brand," *IEEE Software*, vol. 27, no. 5, September–October 2010, pp. 18–19.

[Man81] Mantai, M., "The Effect of Programming Team Structures on Programming Tasks," *CACM*, vol. 24, no. 3, March 1981, pp. 106–113.

[Man97] Mandel, T., *The Elements of User Interface Design*, Wiley, 1997.
[Mar00] Martin, R., "Design Principles and Design Patterns," 2000, available at www.objectmentor.com.
[Mar01] Marciniak, J. (ed.), *Encyclopedia of Software Engineering*, 2nd ed., Wiley, 2001.
[Mar02a] Marick, B., "Software Testing Patterns," 2002, available at www.testing.com/test-patterns/index.html.
[Mar94] Marick, B., *The Craft of Software Testing*, Prentice Hall, 1994.
[Mas02] Mascolo, C., et al., "Mobile Computing Middleware," appears in *Advanced Lectures on Networking* (E. Georgi, ed.), Springer-Verlag, 2002.
[Mat94] Matson, J., et al, "Software Cost Estimation Using Function Points," IEEE Trans. Software Engineering, vol. SE-20, no. 4, April 1994, pp. 275–287.
[McC04] McConnell, S., *Code Complete*, 2nd. ed., Microsoft Press, 2004.
[McC09] McCaffrey, J., "Analyzing Risk Exposure and Risk using PERIL," *MSDN Magazine*, January 2009, available at http://msdn.microsoft.com/en-us/magazine/dd315417.aspx
[McC76] McCabe, T., "A Software Complexity Measure," *IEEE Trans. Software Engineering*, vol. SE-2, December 1976, pp. 308–320.
[McC77] McCall, J., P. Richards, and G. Walters, "Factors in Software Quality," three volumes, NTIS AD-A049-014, 015, 055, November 1977.
[McC94] McCabe, T. J., and A. H. Watson, "Software Complexity," *CrossTalk*, vol. 7, no. 12, December 1994, pp. 5–9.
[McC96] McConnell, S., "Best Practices: Daily Build and Smoke Test," *IEEE Software*, vol. 13, no. 4, July 1996, pp. 143–144.
[McC98] McConnell, S., *Software Project Survival Guide*, Microsoft Press, 1998.
[McC99] McConnell, S., "Software Engineering Principles," *IEEE Software*, vol. 16, no. 2, March–April 1999, available at www.stevemcconnell.com/ieeesoftware/eic04.htm.
[McD93] McDermid, J., and P. Rook, "Software Development Process Models," in *Software Engineer's Reference Book*, CRC Press, 1993, pp. 15/26–15/28.
[McG91] McGlaughlin, R., "Some Notes on Program Design," *Software Engineering Notes*, vol. 16, no. 4, October 1991, pp. 53–54.
[McG94] McGregor, J., and T. Korson, "Integrated Object-Oriented Testing and Development Processes," *Communications of the ACM*, vol. 37, no. 9, September, 1994, pp. 59–77.
[McN10] McNeil, P., *The Web Designer's Idea Book*, vol. 2, How Publishers, 2010.
[Mea10] Mead, N., and Jarzombek, J., "Advancing Software Assurance with Public-Private Collaboration," *IEEE Computer*, vol. 43, no. 9, September 2010, pp. 21–30.
[Mei06] Meier, J., "Web Application Security Engineering," *IEEE Security and Privacy*, July–August 2006, pp. 16–24.
[Mei09] Meier, J., et al., *Microsoft Application Architecture Guide*, 2nd ed., Microsoft Press, 2009, available at http://msdn.microsoft.com/en-us/library/ff650706.
[Mei12] Meier, J., et al., "Chapter 19: Mobile Applications," *Application Architecture Guide, 2.0*, 2012, available at http://apparchguide.codeplex.com/wikipage?title=Chapter%2019%20-%20Mobile%20Applications
[Men01] Mendes, E., N. Mosley, and S. Counsell, "Estimating Design and Authoring Effort," *IEEE Multimedia*, vol. 8, no. 1, January–March 2001, pp. 50–57.
[Mer93] Merlo, E., et al., "Reengineering User Interfaces," *IEEE Software*, January 1993, pp. 64–73.
[Mes08] Messeguer, R., et al. "Communication and Coordination Patterns to Support Mobile Collaboration," *Proceedings of 12th International Conference on Collaborative and Cooperative Work*, April 2008, pp. 565–570.
[Mey09] Meyer, B., et al., "Programs that Test Themselves," *IEEE Computer*, vol. 42, no. 9, September 2009, pp. 46–55.
[Mic04] Microsoft, "Prescriptive Architecture: Integration and Patterns," *MSDN*, May 2004, available at http://msdn2.microsoft.com/en-us/library/ms978700.aspx.
[Mic12] "Principles of Service-Oriented Design," Microsoft, 2012, available at http://msdn.microsoft.com/en-us/library/bb972954.aspx
[Mic13a] Microsoft, "Patterns and Practices," *MSDN*, available at http://msdn.microsoft.com/en-us/library/ff647589.aspx.

[Mic13b] *Microsoft Accessibility Technology for Everyone*, 2013, available at www.microsoft.com/enable/.

[Mil00a] Miller, E., "WebSite Testing," 2000, available at www.soft.com/eValid/Technology/White.Papers/website.testing.html.

[Mil04] Miler, J., and J. Gorski, "Risk Identification Patterns for Software Projects," *Foundations of Computing and Decision Sciences*, vol. 29, no. 1, 2004, pp. 115–131, available at http://iag.pg.gda.pl/iag/download/Miler-Gorski_Risk_Identification_Patterns.pdf.

[Mil72] Mills, H., "Mathematical Foundations for Structured Programming," Technical Report FSC 71-6012, IBM Corp., Federal Systems Division, Gaithersburg, MD, 1972.

[Mil77] Miller, E., "The Philosophy of Testing," in *Program Testing Techniques*, IEEE Computer Society Press, 1977, pp. 1–3.

[Mil87] Mills, H., M. Dyer, and R. Linger, "Cleanroom Software Engineering," *IEEE Software*, September 1987, pp. 19–25.

[Mil88] Mills, H., "Stepwise Refinement and Verification in Box Structured Systems," *Computer*, vol. 21, no. 6, June 1988, pp. 23–35.

[Min95] Minoli, D., *Analyzing Outsourcing*, McGraw-Hill, 1995.

[Mob11] Mobile Labs, "Mobile Application Test Automation: Best Practices for Best Results," a white paper, 2011, available at http://mobilelabsinc.com/wp-content/uploads/2012/01/Mobile-Application-Test-Automation-Best-Practices-White-Paper.pdf.

[Mob12] "Mobile UI Patterns," 2012, available at http://mobile-patterns.com/.

[Mor05] Morales, A., "The Dream Team," *Dr. Dobbs Portal*, March 3, 2005, available at www.ddj.com/dept/global/184415303.

[Mor81] Moran, T., "The Command Language Grammar: A Representation for the User Interface of Interactive Computer Systems," *Intl. Journal of Man-Machine Studies*, vol. 15, pp. 3–50.

[Mus87] Musa, J., A. Iannino, and K. Okumoto, *Engineering and Managing Software with Reliability Measures*, McGraw-Hill, 1987.

[Mye78] Myers, G., *Composite Structured Design*, Van Nostrand, 1978.

[Mye79] Myers, G., *The Art of Software Testing*, Wiley, 1979.

[Myl11] Malavarapu, V., and M. Inanamdar, "Taking Testing to the Cloud," 2012, available at http://www.cognizant.com/InsightsWhitepapers/Taking-Testing-to-the-Cloud.pdf.

[NAS07] NASA, *Software Risk Checklist*, Form LeR-F0510.051, March 2007, available at http://osat-ext.grc.nasa.gov/rmo/spa/SoftwareRiskChecklist.doc.

[Nei11] Neil, T., "A Look Inside Mobile Design Patterns," UXBooth.com, November 12, 2011, available at http://www.uxbooth.com/blog/mobile-design-patters/.

[Nei12] Neil, T., *Mobile Design Pattern Gallery*, O'Reilly Media, 2012.

[Ngu00] Nguyen, H., "Testing Web-Based Applications," *Software Testing and Quality Engineering*, May–June 2000, available at www.stqemagazine.com.

[Ngu01] Nguyen, H., *Testing Applications on the Web*, Wiley, 2001.

[Nie00] Nielsen, J., *Designing Web Usability*, New Riders Publishing, 2000.

[Nie92] Nierstrasz, O., S. Gibbs, and D. Tsichritzis, "Component-Oriented Software Development," *CACM*, vol. 35, no. 9, September 1992, pp. 160–165.

[Nie94] Nielsen, J., and J. Levy, "Measuring Usability: Preference vs. Performance," *CACM*, vol. 37, no. 4, April 1994, pp. 65–75.

[Nie96] Nielsen, J., and A. Wagner, "User Interface Design for the WWW," *Proc. CHI '96 Conf. on Human Factors in Computing Systems*, ACM Press, 1996, pp. 330–331.

[Nir10] Niranjan, P., and C. V. Guru Rao, "A Mockup Tool for Software Component Reuse Library," *Intl. J. Software Engineering & Applications*, vol. 1, no. 2, April 2010, available at http://airccse.org/journal/ijsea/papers/0410ijsea1.pdf.

[Nok13] "Category: Mobile Design," Nokia Developer, 2013, available at http://www.developer.nokia.com/Community/Wiki/Category:Mobile_Design_Patterns.

[Nog00] Nogueira, J., C. Jones, and Luqi, "Surfing the Edge of Chaos: Applications to Software Engineering," Command and Control Research and Technology Symposium, Naval Post Graduate School, Monterey, CA, June 2000, available at www.dodccrp.org/2000CCRTS/cd/html/pdf_papers/Track_4/075.pdf.

[Nor70] Norden, P., "Useful Tools for Project Management" in *Management of Production*, M. K. Starr (ed.), Penguin Books, 1970.

[Nor86] Norman, D. A., "Cognitive Engineering," in *User Centered Systems Design*, Lawrence Earlbaum Associates, 1986.

[Nor88] Norman, D., *The Design of Everyday Things*, Doubleday, 1988.

[Nov04] Novotny, O., "Next Generation Tools for Object-Oriented Development," *The Architecture Journal*, January 2005, available at http://msdn2.microsoft.com/en-us/library/aa480062.aspx.

[Noy02] Noyes, B., "Rugby, Anyone?" *Managing Development* (an online publication of Fawcette Technical Publications), June 2002, available at www.fawcette.com/resources/managingdev/methodologies/scrum/.

[Nun11] Nunes, N., L. Constantine, and R. Kazman, "iUCP: Estimating Interactive Software Project Size with Enhanced Use Case Points," *IEEE Software*, vol. 28, no. 4, July–August 2011, pp. 64–73.

[Obj10] "The Dependency Inversion Principle," *Objectmentor.com*, 2010, available at www.objectmentor.com/resources/articles/dip.pdf>.

[Off02] Offutt, J., "Quality Attributes of Web Software Applications," *IEEE Software*, March–April 2002, pp. 25–32.

[Ols06] Olsen, G., "From COM to Common," *Component Technologies*, ACM, vol. 4, no. 5, June 2006, available at http://acmqueue.com/modules.php?name=Content&pa=showpage&pid=394.

[Ols99] Olsina, L., et al., "Specifying Quality Characteristics and Attributes for Web Sites," *Proc. 1st ICSE Workshop on Web Engineering*, ACM, Los Angeles, May 1999.

[OMG03a] Object Management Group, *OMG Unified Modeling Language Specification*, version 1.5, March 2003, available at www.rational.com/uml/resources/documentation/.

[OMG03b] "Object Constraint Language Specification," in *Unified Modeling Language*, v2.0, Object Management Group, September 2003, available at www.omg.org.

[Orf99] Orfali, R., D. Harkey, and J. Edwards, *Client/Server Survival Guide*, 3rd ed., Wiley, 1999.

[Osb90] Osborne, W. M., and E. J. Chikofsky, "Fitting Pieces to the Maintenance Puzzle," *IEEE Software*, January 1990, pp. 10–11.

[OSO12] *OpenSource.org*, 2012, available at www.opensource.org/.

[Pag85] Page-Jones, M., *Practical Project Management*, Dorset House, 1985, p. vii.

[Par72] Parnas, D., "On Criteria to Be Used in Decomposing Systems into Modules," *CACM*, vol. 14, no. 1, April 1972, pp. 221–227.

[Par96a] Pardee, W., *To Satisfy and Delight Your Customer*, Dorset House, 1996.

[Par96b] Park, R. E., W. B. Goethert, and W. A. Florac, *Goal Driven Software Measurement—A Guidebook*, CMU/SEI-96-BH-002, Software Engineering Institute, Carnegie Mellon University, August 1996.

[Par10] Parnas, D., "Interview: Software Security in the Real World," *IEEE Computer*, vol. 43, no. 1, January 2010, pp. 28–34.

[Par11] Pardo, C., et al., "Harmonizing Quality Assurance Processes and Product Characteristics," *IEEE Computer*, June 2011, pp. 94–96.

[Pas10] Passos, L., et al., "Static Architecture-Conformance Checking: An Illustrative Overview," *IEEE Software*, vol. 27, no. 5, September–October 2010, pp. 82–89,

[Pat07] Patton, J., "Understanding User Centricity," *IEEE Software*, vol. 24, no. 6, November–December 2007, pp. 9–11.

[Pau94] Paulish, D., and A. Carleton, "Case Studies of Software Process Improvement Measurement," *Computer*, vol. 27, no. 9, September 1994, pp. 50–57.

[Pea11] Pearson, S., and M. Mont, "Sticky Policies: An Approach for Managing Privacy across Multiple Parties," *IEEE Computer*, vol. 44, no. 9, September 2011, pp. 60–68.

[Pee11] Peeters, J., "Agile Security Requirements Engineering," *Proceedings of First International Workshop on Empirical Requirements Engineering*, 2011.

[Per74] Persig, R., *Zen and the Art of Motorcycle Maintenance*, Bantam Books, 1974.

[Pha89] Phadke, M., *Quality Engineering Using Robust Design*, Prentice Hall, 1989.

[Pha97] Phadke, M., "Planning Efficient Software Tests," *CrossTalk*, vol. 10, no. 10, October 1997, pp. 11–15.

[Phi02] Phillips, M., "CMMI V1.1 Tutorial," April 2002, available at www.sei.cmu.edu/cmmi/.

[Phi98] Phillips, D., *The Software Project Manager's Handbook*, IEEE Computer Society Press, 1998.

[Pol45] Polya, G., *How to Solve It*, Princeton University Press, 1945.

[Poo88] Poore, J., and H. Mills, "Bringing Software Under Statistical Quality Control," *Quality Progress*, November 1988, pp. 52–55.

[Poo93] Poore, J., H. Mills, and D. Mutchler, "Planning and Certifying Software System Reliability," *IEEE Software*, vol. 10, no. 1, January 1993, pp. 88–99.

[Pop08] Popcorn, F., *Faith Popcorn's Brain Reserve*, 2008, available at www.faithpopcorn .com/.

[Pot04] Potter, M., *Set Theory and Its Philosophy: A Critical Introduction*, Oxford University Press, 2004.

[Pow02] Powell, T., *Web Design*, 2nd ed., McGraw-Hill/Osborne, 2002.

[Pow98] Powell, T., *Web Site Engineering*, Prentice Hall, 1998.

[Pra07] Pratt, M., "Five Tips for Building an Incident Response Plan," *Computerworld*, May 16, 2007, available at http://www.computerworld.com/s/article/9019558/Five_tips_for_building_an_incident_response_pla.

[Pre05] Pressman, R., *Adaptable Process Model*, Version 2.0, R. S. Pressman & Associates, 2005, available at www.rspa.com/apm/index.html.

[Pre08] Pressman, R., and D. Lowe, *Web Engineering: A Practitioner's Approach*, McGraw-Hill, 2008.

[Pre88] Pressman, R., *Making Software Engineering Happen*, Prentice Hall, 1988.

[Pre94] Premerlani, W., and M. Blaha, "An Approach for Reverse Engineering of Relational Databases," *CACM*, vol. 37, no. 5, May 1994, pp. 42–49.

[Pri10] Prince, B., "10 Most Dangerous Web App Security Flaws," *eWeek.com*, April, 19, 2010, available at http://www.eweek.com/c/a/Security/10-Most-Dangerous-Web-App-Security-Risks-730757/.

[Put78] Putnam, L., "A General Empirical Solution to the Macro Software Sizing and Estimation Problem," *IEEE Trans. Software Engineering*, vol. SE-4, no. 4, July 1978, pp. 345–361.

[Put92] Putnam, L., and W. Myers, *Measures for Excellence*, Yourdon Press, 1992.

[Put97a] Putnam, L., and W. Myers, "How Solved Is the Cost Estimation Problem?" *IEEE Software*, November 1997, pp. 105–107.

[Put97b] Putnam, L., and W. Myers, *Industrial Strength Software: Effective Management Using Measurement*, IEEE Computer Society Press, 1997.

[Pyz03] Pyzdek, T., *The Six Sigma Handbook*, McGraw-Hill, 2003.

[QAI08] *A Software Engineering Curriculum*, QAI, 2008, information available at www.qaie-school.com/innerpages/offer.asp.

[QSM02] "QSM Function Point Language Gearing Factors," Version 2.0, *Quantitative Software Management*, 2002, available at www.qsm.com/FPGearing.html.

[Qur09] Qureshi N., and A. Perini, "Engineering Adaptive Requirements," *Proceedings of Workshop on Software Engineering for Adaptive and Self-Managing Systems*, Vancouver, May 2009, pp. 126–131.

[Rad02] Radice, R., *High-Quality Low Cost Software Inspections*, Paradoxicon Publishing, 2002.

[Rai06] Raiffa, H., *The Art and Science of Negotiation*, Belknap Press, 2005.

[Rat12] Raatikainen, M., et al., "Mobile Content as a Service: A Blueprint for a Vendor-Neutral Cloud of Mobile Devices," *IEEE Software*, vol. 29, no. 4, July–August, 2012, pp. 28–32.

[Red10] Redwine, S., "Fitting Software Assurance into Higher Education," *IEEE Computer*, vol. 43, no. 9, September 2010, pp. 41–66.

[Ree99] Reel, J., "Critical Success Factors in Software Projects," *IEEE Software*, May 1999, pp. 18–23.

[Reu12] Reuveni, D. "Crowdsourcing Provides Answer to App Testing Dilemma," 2012, available at http://www.wirelessweek.com/Articles/2010/02/Mobile-Content-Crowdsourcing-Answer-App-Testing-Dilemma-Mobile-Applications/.

[Ric01] Ricadel, A., "The State of Software Quality," *InformationWeek*, May 21, 2001, available at www.informationweek.com/838/quality.htm.

[Ric04] Rico, D., *ROI of Software Process Improvement*, J. Ross Publishing, 2004. A summary article can be found at http://davidfrico.com/rico03a.pdf.

[Rob10] Robinson, W., "A Roadmap for Comprehensive Requirements Monitoring," *IEEE Computer*, vol. 43, no. 5, May 2010, pp. 64–72.

[Roc06] *Graphic Design That Works*, Rockport Publishers, 2006.

[Roc11] Rocha, F., S, Abreus, and M. Correia, "The Final Frontier: Confidentiality and Privacy in the Cloud," *IEEE Computer*, vol. 44, no. 9, September 2011, pp. 44–50.

[Roc94] Roche, J. M., "Software Metrics and Measurement Principles," *Software Engineering Notes*, ACM, vol. 19, no. 1, January 1994, pp. 76–85.

[Rod12] Rodriguez, S., "Half of Americans Now Have Smartphones," *The Los Angeles Times*, August 14, 2012, available at http://www.latimes.com/business/technology/la-fi-tn-americans-smartphones-20120814,0,6077673.story.

[Rod98] Rodden, T., et al., "Exploiting Context in HCI Design for Mobile Systems," *Proceedings of Workshop on Human Computer Interaction with Mobile Devices*, 1998.

[Roe00] Roetzheim, W., "Estimating Internet Development," *Software Development*, August 2000, available at www.sdmagazine.com/documents/s=741/sdm0008d/0008d.htm.

[Rog12] Rogers, A., "Software Quality Assurance Engineers Are the Happiest Workers in America," *Business Insider*, April 16, 2012, available at http://www.businessinsider.com/happiest-jobs-in-america-2012-4.

[Rom11] Roman, R., et al., "Securing the Internet of Things," *IEEE Computer*, vol. 44, no. 9, September 2011, pp. 51–58.

[Roo09] Rooksby, J., et al., "Testing in the Wild: The Social and Organizational Dimensions of Real World Practice," *Journal of Computer Supported Work*, vol. 18, no. 5–6, December 2009, pp. 559–580.

[Roo96] Roos, J., "The Poised Organization: Navigating Effectively on Knowledge Landscapes," 1996, available at www.imd.ch/fac/roos/paper_po.html.

[Ros04] Rosenhainer, L., "Identifying Crosscutting Concerns in Requirements Specifications," 2004, available at http://trese.cs.utwente.nl/workshops/oopsla-early-aspects-2004/Papers/Rosenhainer.pdf.

[Ros75] Ross, D., J. Goodenough, and C. Irvine, "Software Engineering: Process, Principles and Goals," *IEEE Computer*, vol. 8, no. 5, May 1975.

[Rot02] Roth, J., "Seven Challenges for Developers of Mobile Groupware," in *Proceedings of Computer Human Interaction Workshop on Mobile Ad Hoc Collaboration*, 2002.

[Rou02] Rout, T. (project manager), *SPICE: Software Process Assessment—Part 1: Concepts and Introductory Guide*, 2002, available at www.sqi.gu.edu.au/spice/suite/download.html.

[Roy70] Royce, W., "Managing the Development of Large Software Systems: Concepts and Techniques," *Proc. WESCON*, August 1970.

[Roz11] Rozanski, N., and E. Woods, *Software Systems Architecture*, second edition, Addison-Wesley, 2011.

[Rum91] Rumbaugh, J., et al., *Object-Oriented Modeling and Design*, Prentice Hall, 1991.

[Rya11a] Ryan, M., "Cloud Computing Privacy Concerns on Our Doorstep," *Communication of the ACM*, vol. 44, no. 2, January 2011, pp. 36–38.

[Sae11] Saeed, A., T. Chen, and O. Alzubi, "Malicious and Spam Posts in Online Social Networks," *IEEE Computer*, vol. 44, no. 9, September 2011, pp. 23–28.

[Saf08] Safanov, V., *Using Aspect-Oriented Programming for Trustworthy Software Development*, Wiley-Interscience, 2008.

[Sai11] Saieddian, H., and D. Broyles, "Security Vulnerabilities in the Same Origin Policy Implications and Alternatives," *IEEE Computer*, vol. 44, no. 9, September 2011, pp. 29–36.

[Sal09] Saleh, K., and G. Elshahry, "Modeling Security Requirements for Trustworthy Systems," *Encyclopedia of Information Science and Technology*, 2nd ed., 2009.

[Sar06] Sarwate, A., "Hot or Not: Web Application Vulnerabilities," *SC Magazine*, December 27, 2006, available at http://www.scmagazine.com/hot-or-not-web-application-vulnerabilities/article/34315/.

[Saw08] Sawyer, S., et al., "Social Interactions of Information Systems Development Teams: A Performance Perspective," *Information Systems Journal*, vol. 20, 2008, pp. 81–107.

[Sca00] Scacchi, W., "Understanding Software Process Redesign Using Modeling, Analysis, and Simulation," *Software Process Improvement and Practice*, Wiley, 2000, pp. 185–195, available at www.ics.uci.edu/~wscacchi/Papers/Software_Process_Redesign/SPIP-ProSim99.pdf.

[Sce02] Sceppa, D., *Microsoft ADO.NET*, Microsoft Press, 2002.

[Sch01b] Schwaber, K., and M. Beedle, *Agile Software Development with SCRUM*, Prentice Hall, 2001.

[Sch03] Schlickman, J., *ISO 9001: 2000 Quality Management System Design*, Artech House Publishers, 2003.

[Sch06] Schmidt, D., "Model-Driven Engineering," *IEEE Computer*, vol. 39, no. 2, February 2006, pp. 25–31.

[Sch09] Schumacher, R. (ed.), *Handbook of Global User Research*, Morgan-Kaufmann, 2009.

[Sch11] Schilit, B. "Mobile Computing: Looking to the Future," *IEEE Computer*, vol. 44, no. 5, May 2011. pp. 28–29.

[Sch98a] Schneider, G., and J. Winters, *Applying Use Cases*, Addison-Wesley, 1998.

[Sch98c] Schulmeyer, G., and J. McManus (eds.), *Handbook of Software Quality Assurance*, 3rd ed., Prentice Hall, 1998.

[Sch99] Schneidewind, N., "Measuring and Evaluating Maintenance Process Using Reliability, Risk, and Test Metrics," *IEEE Trans. SE*, vol. 25, no. 6, November–December 1999, pp. 768–781, available at www.dacs.dtic.mil/topics/reliability/IEEETrans.pdf.

[SDS08] Spice Document Suite, "The SPICE and ISO Document Suite," *ISO-Spice*, 2008, available at www.isospice.com/articles/9/1/SPICE-Project/Page1.html.

[SEE03] The Software Engineering Ethics Research Institute, "UCITA Updates," 2003, available at http://seeri.etsu.edu/default.htm.

[SEI00] *SCAMPI, V1.0 Standard CMMI®Assessment Method for Process Improvement: Method Description*, Software Engineering Institute, Technical Report CMU/SEI-2000-TR-009, available at www.sei.cmu.edu/publications/documents/00.reports/00tr009.html.

[SEI02] "Maintainability Index Technique for Measuring Program Maintainability," SEI, 2002, available at www.sei.cmu.edu/str/descriptions/mitmpm_body.html.

[SEI08] "The Ideal Model," Software Engineering Institute, 2008, available at www.sei.cmu.edu/ideal/.

[SEI13] "Software Product Lines—Overview," Software Engineering Institute, 2013, available at www.sei.cmu.edu/productlines/.

[Sha05] Shalloway, A., and J. Trott, *Design Patterns Explained*, 2nd ed., Addison-Wesley, 2005.

[Sha09] Shaw, M., "Continuing Prospects for an Engineering Discipline of Software," *IEEE Software*, vol. 26, no. 8, November–December 2009, pp. 64–67.

[Sha95a] Shaw, M., and D. Garlan, "Formulations and Formalisms in Software Architecture," *Volume 1000—Lecture Notes in Computer Science*, Springer-Verlag, 1995.

[Sha95b] Shaw, M., et al., "Abstractions for Software Architecture and Tools to Support Them," *IEEE Trans. Software Engineering*, vol. SE-21, no. 4, April 1995, pp. 314–335.

[Sha96] Shaw, M., and D. Garlan, *Software Architecture*, Prentice Hall, 1996.

[She10] Sheldon, F., and Vishik, C. "Moving Toward Trustworthy Systems: R&D Essentials," *IEEE Computer*, vol. 43, no. 9, September 2010, pp. 31–40.

[Shn04] Shneiderman, B., and C. Plaisant, *Designing the User Interface*, 4th ed., Addison-Wesley, 2004.

[Shn09] Shneiderman, B., et al., *Designing the User Interface*, 5th ed., Addison-Wesley, 2009.

[Shn80] Shneiderman, B., *Software Psychology*, Winthrop Publishers, 1980, p. 28.

[Sho83] Shooman, M., *Software Engineering*, McGraw-Hill, 1983.

[Shu12] Shull, F., "Designing a World at Your Fingertips: A Look at Mobile User Interfaces," *IEEE Software*, vol. 29, no. 4, July–August, 2012, pp. 4–7.

[Sin08] Sinn, R. H., *Software Security Technologies: A Programmatic Approach*, Addison-Wesley, 2008.

[Sne03] Snee, R., and R. Hoerl, *Leading Six Sigma*, Prentice Hall, 2003.

[Sne95] Sneed, H., "Planning the Reengineering of Legacy Systems," *IEEE Software*, January 1995, pp. 24–25.

[Soa10] Soares, G., et al., "Making Program Refactoring Safer," *IEEE Software*, vol. 37, no. 4, July–August 2010, pp. 52–57.

[Soa11] Soasta, "Five Strategies for Performance Testing Mobile Applications," a white paper, 2011, available at http://info.soasta.com/performance-testing-mobile-apps-sem2.html.

[Sob10] Sobel, A., and G. McGraw, "Interview: Software Security in the Real World," *IEEE Computer*, vol. 43, no. 9, September 2010, pp. 47–53.

[Sol99] van Solingen, R., and E. Berghout, *The Goal/Question/Metric Method*, McGraw-Hill, 1999.

[Som05] Somerville, I., "Integrating Requirements Engineering: A Tutorial," *IEEE Software*, vol. 22, no. 1, January–February 2005, pp. 16–23.

[Som97] Somerville, I., and P. Sawyer, *Requirements Engineering*, Wiley, 1997.

[Sou08] de Sousa, C., and D. Redmiles, "An Empirical Study of Software Developer's Management of Dependencies and Changes," *ICSE Proceedings*, May 2008, available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.158.427&rep=rep1&type=pdf.

[Spa11] Spagnolli, B., et al., "Eco-Feedback on the Go: Motivating Energy Awareness," *IEEE Computer*, vol. 44, no. 5, May 2011, pp. 38–45.

[SPI99] "SPICE: Software Process Assessment, Part 1: Concepts and Introduction," Version 1.0, ISO/IEC JTC1, 1999.

[Spl01] Splaine, S., and S. Jaskiel, *The Web Testing Handbook*, STQE Publishing, 2001.

[Spo02] Spolsky, J., "The Law of Leaky Abstractions," November 2002, available at www.joelonsoftware.com/articles/LeakyAbstractions.html.

[Spr04] Spriestersbach, A., and T. Springer, "Quality Attributes in Mobile Web Application Development," 2004, available at http://atlas.tk.informatik.tu-darmstadt.de/Publications/2004/profes.pdf.

[Sri01] Sridhar, M., and N. Mandyam, "Effective Use of Data Models in Building Web Applications," 2001, available at www2002.org/CDROM/alternate/698/.

[SSO08] Software-Supportability.org, 2008, available at www.software-supportability.org/.

[Sta10] Stafford, T., and R. Poston, "Online Security Threats and Computer User Intentions," *IEEE Computer*, vol. 43, no. 1, January 2010, pp. 58–64.

[Sta97] Stapleton, J., *DSDM—Dynamic System Development Method: The Method in Practice*, Addison-Wesley, 1997.

[Sta97b] Statz, J., D. Oxley, and P. O'Toole, "Identifying and Managing Risks for Software Process Improvement," *CrossTalk*, April 1997, available at www.stsc.hill.af.mil/crosstalk/1997/04/identifying.asp.

[Ste93] Stewart, T., "Reengineering: The Hot New Managing Tool," *Fortune*, August 23, 1993, pp. 41–48.

[Sto05] Stone, D., et al., *User Interface Design and Evaluation*, Morgan Kaufman, 2005.

[Str08] Stickland, J., "How Cloud Computing Works," http://computer.howstuffworks.com/cloud-computing/cloud-computing.htm, April, 2008.

[Ste10] Stephens, M., and D. Rosenberg, *Design Driven Testing*, Apress, 2010.

[Tai89] Tai, K., "What to Do Beyond Branch Testing," *ACM Software Engineering Notes*, vol. 14, no. 2, April 1989, pp. 58–61.

[Tai12] Taivalsaari, A., and K. Systa, "Mobile Content as a Service: A Blueprint for a Vendor-Neutral Cloud of Mobile Devices," *IEEE Software*, vol. 29, no. 4, July–August 2012, pp. 28–33.

[Tan01] Tandler, P., "Aspect-Oriented Model-Driven Development for Mobile Context-Aware Computing," *Proceedings of UbiComp 2001: Ubiquitous Computing*, 2001.

[Tay09] Taylor, R., N. Medvidovic, and E. Dashofy, *Software Architecture*, Wiley, 2009.

[Tay90] Taylor, D., *Object-Oriented Technology: A Manager's Guide*, Addison-Wesley, 1990.

[The01] Thelin, T., H. Petersson, and C. Wohlin, "Sample Driven Inspections," *Proc. of Workshop on Inspection in Software Engineering (WISE'01)*, Paris, France, July 2001, pp. 81–91. Available at http://www.cas.mcmaster.ca/wise/wise01/ThelinPeterssonWohlin.pdf.

[The13] "The Emergence of Cloud Computing," 2013, available at http://www.opensourcery.com/clouddev.htm.

[Tho04] Thomas, J., et al., *Java Testing Patterns*, Wiley, 2004.

[Tho92] Thomsett, R., "The Indiana Jones School of Risk Management," *American Programmer*, vol. 5, no. 7, September 1992, pp. 10–18.

[Tic05] *TickIT*, 2005, available at www.tickit.org/.

[Tid02] Tidwell, J., "IU Patterns and Techniques," May, 2002, available at http://designinginterfaces.com/.

[Til00] Tillman, H., "Evaluating Quality on the Net," Babson College, May 30, 2000, available at www.hopetillman.com/findqual.html#2.

[Tog01] Tognozzi, B., "First Principles," *askTOG*, 2001, available at www.asktog.com/basics/firstPrinciples.html.

[Tra95] Tracz, W., "Third International Conference on Software Reuse—Summary," *ACM Software Engineering Notes*, vol. 20, no. 2, April 1995, pp. 21–22.

[Tyr05] Tyree, J., and A. Akerman, "Architectural Decisions: Demystifying Architecture," *IEEE Software*, vol. 22, no. 2, March–April, 2005.

[Uem99] Uemura, T., S. Kusumoto, and K. Inoue, "A Function Point Measurement Tool for UML Design Specifications," *Proc. of Sixth International Symposium on Software Metrics*, IEEE, November 1999, pp. 62–69.

[Ull97] Ullman, E., *Close to the Machine: Technophilia and its Discontents*, City Lights Books, 2002.

[Uni03] Unicode, Inc., *The Unicode Home Page*, 2003, available at www.unicode.org/.

[USA87] *Management Quality Insight*, AFCSP 800-14 (U.S. Air Force), January 20, 1987.

[Ute12] UTest, E-book: *Essential Guide to Mobile App Testing, 2012*, available at http://www.utest.com/landing-blog/essential-guide-mobile-app-testing.

[UXM10] "Four Best User Interface Pattern Libraries," *UX Movement.com*, available at http://uxmovement.com/resources/4-best-design-pattern-libraries/.

[Vac06] Vacca, J., *Practical Internet Security*, Springer, 2006.

[Van02] Van Steen, M., and A. Tanenbaum, *Distributed Systems: Principles and Paradigms*, Prentice Hall, 2002.

[Van89] Van Vleck, T., "Three Questions About Each Bug You Find," *ACM Software Engineering Notes*, vol. 14, no. 5, July 1989, pp. 62–63.

[Ven03] Venners, B., "Design by Contract: A Conversation with Bertrand Meyer," *Artima Developer*, December 8, 2003, available at www.artima.com/intv/contracts.html.

[Vin11] Vinson, L., "Mobile Application Testing: Process, Tools, and Techniques," 2011, available at http://threeminds.organic.com/2011/05/mobile-application-testing-process-tools-techniques.html.

[Voa12] Voas, J., et al., "Mobile Software App Takeover," *IEEE Software*, vol. 29, no. 4, July–August 2012, pp. 25–27.

[Wal03] Wallace, D., I. Raggett, and J. Aufgang, *Extreme Programming for Web Projects*, Addison-Wesley, 2003.

[Wal12] Walker, J., "Computer Programmers Learn Tough Lesson in Sharing," *The Wall Street Journal*, vol. 260, no. 48, August 27, 2012, p. 1.

[War07] Ward, M., "Using VoIP Software Building zBlocks—A Look at the Choices," TMNNet, 2007, available at www.tmcnet.com/voip/0605/featurearticle-using-voip-software-building-blocks.htm.

[War74] Warnier, J. D., *Logical Construction of Programs*, Van Nostrand-Reinhold, 1974.

[Was10] Wasserman, A., "Software Engineering Issues for Mobile Application Development," *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research*, 2010.

[Web05] Weber, S., *The Success of Open Source*, Harvard University Press, 2005.

[Web13] Web Application Security Consortium, 2013, available at http://www.webappsec.org/.

[Wee11] Weevers, I., "Seven Guidelines for Designing High Performance Mobile User Experiences," *Smashing Magazine*, July 18, 2011, available at http://uxdesign.smashingmagazine.com/2011/07/18/seven-guidelines-for-designing-high-performance-mobile-user-experiences/

[Wei86] Weinberg, G., *On Becoming a Technical Leader*, Dorset House, 1986.

[Wel01] vanWelie, M., "Interaction Design Patterns," 2001, available at www.welie.com/patterns/.

[Wel99] Wells, D., "XP—Unit Tests," 1999, available at www.extremeprogramming.org/rules/unittests.html.

[Wev11] Wever, A., and N. Maiden, "Requirements Analysis: The Next Generation," *IEEE Software*, vol. 28, no. 2, March–April 2011, pp. 22–23.

[Whi07] Whitehead, J., "Collaboration in Software Engineering: A Roadmap," in *Future of Software Engineering 2007*, L. Briand and A. Wolf (eds.), IEEE-CS Press, 2007.

[Whi08] White, J., "Start Your Engines: Mobile Application Development," April 22, 2008, available at http://www.devx.com/SpecialReports/Article/37693.

[Whi12] Whittaker, J., et al., *How Google Tests Software*, Addison-Wesley, 2012.

[Whi97] Whitmire, S., *Object-Oriented Design Measurement*, Wiley, 1997.

[Wie02] Wiegers, K., *Peer Reviews in Software*, Addison-Wesley, 2002.

[Wie03] Wiegers, K., *Software Requirements*, 2nd ed., Microsoft Press, 2003.

[Wik12] Wikipedia, "Data Modeling," 2012, available at http://en.wikipedia.org/wiki/Data_modeling.

[Wik13] "Cloud Computing," January 2013, available at http://en.wikipedia.org/wiki/Cloud_computing.

[Wil00] Williams, L., and R. Kessler, "All I Really Need to Know about Pair Programming I Learned in Kindergarten," *CACM*, vol. 43, no. 5, May 2000, available at http://collaboration.csc.ncsu.edu/laurie/Papers/Kindergarten.PDF.

[Wil05] Willoughby, M., "Q&A: Quality Software Means More Secure Software," *Computerworld*, March 21, 2005, available at www.computerworld.com/securitytopics/security/story/0,10801,91316,00.html.

[Wil97] Williams, R., J. Walker, and A. Dorofee, "Putting Risk Management into Practice," *IEEE Software*, May 1997, pp. 75–81.

[Wil99] Wilkens, T., "Earned Value, Clear and Simple," *Primavera Systems*, April 1, 1999, p. 2.

[Win90] Wing, J., "A Specifier's Introduction to Formal Methods," *IEEE Computer*, vol. 23, no. 9, September 1990, pp. 8–24.

[Wir71] Wirth, N., "Program Development by Stepwise Refinement," *CACM*, vol. 14, no. 4, 1971, pp. 221–227.

[Wir90] Wirfs-Brock, R., B. Wilkerson, and L. Weiner, *Designing Object-Oriented Software*, Prentice Hall, 1990.

[WMT02] *Web Mapping Testbed Tutorial*, 2002, available at www.webmapping.org/vcgdocuments/vcgTutorial/.

[Woh94] Wohlin, C., and P. Runeson, "Certification of Software Components," *IEEE Trans. Software Engineering*, vol. SE-20, no. 6, June 1994, pp. 494–499.

[Wor04] World Bank, *Digital Technology Risk Checklist*, 2004, available at www.moonv6.org/lists/att-0223/WWBANK_Technology_Risk_Checklist_Ver_6point1.pdf.

[Wri11] Wright, A., "Lessons Learned: Architects Are Facilitators, Too!" *IEEE Software*, vol. 28, no. 2, January–February 2011, pp. 70–72.

[W3C03] World Wide Web Consortium, *Web Content Accessibility Guidelines*, 2003, available at www.w3.org/TR/2003/WD-WCAG20-20030624/.

[Yac03] Yacoub, S., et al., *Pattern-Oriented Analysis and Design*, Addison-Wesley, 2003.

[Yah13] Yahoo Developer Network, Yahoo! Design Pattern Library, 2013, available at http://developer.yahoo.com/ypatterns/.

[Yau11] Yau, S., and H. An, "Software Engineering Meets Services and Cloud Computing," *IEEE Computer*, vol. 44, no. 10, October 2011, pp. 47–53.

[You01] Young, R., *Effective Requirements Practices*, Addison-Wesley, 2001.

[You75] Yourdon, E., *Techniques of Program Structure and Design*, Prentice Hall, 1975.

[You95] Yourdon, E., "When Good Enough Is Best," *IEEE Software*, vol. 12, no. 3, May 1995, pp. 79–81.

[Yau02] Yaun, M. "Best Tools for Mobile Application Development," *Java World*, available at www.javaworld.com/javaworld/jw-10-2002/jw-1018-wireless.html.

[Zah90] Zahniser, R., "Building Software in Groups," *American Programmer*, vol. 3, no. 7–8, July–August 1990.

[Zah94] Zahniser, R., "Timeboxing for Top Team Performance," *Software Development*, March 1994, pp. 35–38.

[Zha05] Zhang, W., and S. Jarzabek, "Reuse without Compromising Performance: Industrial Experience from RPG Software Product Line for Mobile Devices," *Proceedings of 9th Software Product Line Conference*, September 2005, pp. 57–69.

[Zim11] Zimmermann, O., "Architectural Decisions as Reusable Design Assets," *IEEE Software*, vol. 28, no. 1, January–February 2011, pp. 64–69.

[Zul92] Zultner, R., "Quality Function Deployment for Software: Satisfying Customers," *American Programmer*, February 1992, pp. 28–41.

[Zus90] Zuse, H., *Software Complexity: Measures and Methods*, DeGruyter, 1990.

[Zus97] Zuse, H., *A Framework of Software Measurement*, DeGruyter, 1997.