互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

第1章　为什么系统分析员需要学习UML

1.1　概述

系统分析员（System Analyst）的工作相当辛苦，他们站在用户与开发人员的中间，作为两者之间的沟通桥梁。系统分析员一方面需要向用户搜集并理清需求（Requirements），另一头又得急忙向开发人员提出清晰且明确的需求。

在项目进行期间，系统分析员除了得请神明保佑自己最好别误解或遗漏需求外，还得面对用户变更需求的反复性格，以及开发人员不愿因需求变更而白做工的强硬态度。这一切现象让系统分析员心力交瘁、焦头烂额。

在OO（Object-Oriented，面向对象的）与UML（Unified Modeling Language，统一建模语言）成了挡不住的潮流之后，程序员（Programmer）大量使用C++、Java等OO程序语言，同时也进一步带动设计师（System Designer）使用UML来表达关于OO设计。所以，设计师拿到系统分析文件后所做的第1件事情，便是将非OO文件转成OO的UML图，随后才能进行复杂的设计，并且生成各式的UML图，交由程序员按图编码。

然而，非OO的需求文件转成OO的UML图，不仅缺乏效率而且错误百出。许多公司开始意识到这样的问题，纷纷要求系统分析员学习OO概念，并且采用UML编写系分文件。这样一来，OO概念从分析开始，通过设计，一路贯穿到实现，沟通零误差。

UML是一套用来表达OO分析设计的国际标准语言，从1997年发展至今，吸引了相当多的爱好者，也发展出各式付费或免费的UML工具。挑选一套UML工具，作为系统分析员、设计师和程序员的工作平台，有助于提高工作效率。系统分析员生成的UML文件，可以交由设计师添加设计细节，最后再交由程序员按图编码。

1.2　UML并非万能

有些系统分析员对UML怀有高度期望，希望采用UML来搜集及编写需求之后，可以不再误解或遗漏需求，或者可以降低需求变更。不难想见，系统分析员经常得面对这些问题，当然期望学了UML之后，可以一劳永逸地解决掉这些问题。可是UML并非万能，无法根除这些本质性的问题，不过也不必悲观，总是有对策可以来处置需求误解、遗漏或变更的情况。

人跟人的沟通，本来就会产生误解，更何况用户与系统分析员的专业背景不同，所以当然

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

会在访谈的过程中充满大大小小的误解，既是在所难免，也就无法避免。再者，人脑并非计算机，谈着谈着，总是有多多少少的遗漏，无法在区区几次匆忙的访谈中，明确又清晰地条理分明。因而想在用户与系统分析师之间没有误会，这无疑是强人所难。

既然，访谈之中会有误解与遗漏，那日后反反复复多次地通过电话、电子邮件、会面来变更需求，或者是经过展示之后才发现错误而变更需求的烦人事件，也就不可抗拒地自然而然地发生了。

虽然，UML无法根除这些本质性的问题，但我们还是有对策可以面对这样的现象，试图减轻系统分析员的工作压力。对策如下：

- 使用UML图引导访谈，降低遗漏需求的情况。UML提供10多款不同功能的图，可以让系统分析员在访谈过程中，通过多款不同的图来理清需求各种不同角度的面貌，降低遗漏。而且，每款图都有它独特的组成图标，在绘制每一个图示时，都将引导系统分析员提出适时且重要的问题，以便搜集与理清需求。
- 快速生成可执行的程序片段，通过展示来凸显误解。
- 封装变化，让需求发生变化时，可以追踪到变化之处，迅速改版，并且不让变化起涟漪效应，向外扩散。

所以，在本书里，我们将要求系统分析员学习多款UML图，并且在编写需求时，不仅得生成让用户签字的文件，还得同时生成让设计师能接续设计的UML图件。这样的要求对于系统分析员确实相当为难，但是想要跟用户确认需求、签字同意，除了提交用户能够看懂的文字（Text）之外，别无选择。然而，想要满脑子OO概念的设计师和程序员，可以高效率且零误解地看懂系统分析员生成的需求文件，除了通过具OO概念的UML图外，目前看来似乎别无它法。

1.3 UML图

在本书中，系统分析员将学到两大类的UML图：行为图（Behavior Diagrams）与结构图（Structure Diagrams）。

行为图将引导系统分析员分析且理清“系统该做些什么”？系统分析员在绘制行为图时，可以聚焦在系统多方面的动作，像是系统与用户之间的交互，或者是某种对象（Object）因为事件的刺激以至于发生某些反应动作，以及一群对象交互完成某项服务，等等。系统分析员在访谈期间或结束之后，可以通过这些不同功能的行为图，获知系统多方面的行为。

系统分析员主要会学到下列的UML图，它们都归属于行为类：

- 用例图（Use Case Diagram）
- 活动图（Activity Diagram）
- 状态图（State Machine Diagram）
- 序列图（Sequence Diagram）

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

结构图将引导系统分析员获知“系统的重要组成元素是什么”？通常，通过结构图将引导系统分析员理清业务里（Business）的专有名词，以及专有名词之间的关系，以此作为系统的核心结构。其余，技术面的结构设计就留待设计师去伤脑筋了，这部分并不是系统分析员的工作职责。

系统分析员主要会学的UML图是类图（Class Diagram），属于结构图。

1.4　重要的OO及UML概念

OO是一种比较直接的设计思维，在设计软件时，让软件世界里的程序对象对应真实世界里的具体事物，以此来模拟真实世界的运作情况。观察真实世界可以发现，真实世界由各式各样的事物所组成，每种事物都有它特有的结构和行为，而且在联系起不同事物之后，还能够展现出丰富多元的能力。所以，OO软件也由各式各样的软件对象所组成，并且合力提供多样化的服务，以此参与企业运作过程。

OO概念是UML的基础，也就是说，系统分析员在学习使用UML的同时，其实就是在应用OO概念。换个角度来看，UML与OO两者互为表里，系统分析员脑子里运用的是OO概念，但是表达出来的需求文件内容却是使用UML图。

UML最大的特色在于它是图形语言，因此享有图形思考与表达的优势。所以在本书里，我们除了要求系统分析员在编写需求文件时需以UML图为主、文字为辅外，还要求系统分析员在访谈及讨论会议期间，皆需以UML图引导整个会议进行，并且以生成UML图作为其会议结果。

- 访谈即是绘制UML图。
- 讨论即是修改UML图。

想象一下，系统分析员带着安装了UML工具的笔记本电脑，到用户或领域专家的办公处所进行访谈，访谈的结果就是UML图。系统分析员启动UML工具的同时，就是访谈的开始。系统分析员开始动手绘制第一张UML图，也同步开始请教用户各项问题，一边绘图一边跟用户澄清认知，以便完成UML图。按照这样的过程，生成一张又一张的UML图。当这些UML图都绘制并且经过确认之后，系统分析员打印UML图交给用户，并且将UML文件传回给公司，完成此次访谈会议。

在访谈记录正式成为需求文件期间，用户有任何补充或者系统分析员有任何疑问，都切记以手上的UML图为讨论依据。同样地，系统分析员提交文件给设计师之后，如果设计师有任何疑问，需要开讨论会议的话，也必须以UML图为讨论依据。没有准备好UML图，就不要浪费时间开无谓的讨论会议，与会人员没有修正UML图，讨论会议就不算结束。

在了解OO与UML的相关性之后，接下来我们会介绍几项重要的OO概念，当然它们同时也是UML里的重要概念。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

1.4.1 对象

真实世界由琳琅满目的事物所构成，但并非所有的事物都适合对应成软件对象（Object）。对于系统分析员而言，候选对象应该同时符合下列两项条件：

1. 在企业运作过程中，业务人员会使用到的专业事物或概念。
2. 而且在信息化时，系统也会用到，或者需要保存。

系统分析员在访谈业务人员时，可以提出类似下述的问句，以便获知重要的对象。如：

- 在执行这项工作时，你们会用到哪些专业概念？（探问对象）
- 你们在执行这项工作时，会需要用到哪些数据？（探问对象）

诸如此类的问句，有助于找到重要的对象。软件公司可以多向资深的系统分析员搜集此类的智慧问句，甚至编写成系统分析员通用的工作手册。

1.4.2 属性与操作

当试图去了解真实世界中任一事物时，有多元的角度可以去探寻。不过，系统分析员并不需要这么样深入了解对象。对于任何一种对象，系统分析员只需要针对下列两项问题去探寻：

1. 对象需要记录哪些属性（Attributes）？
2. 对象可以提供哪些操作（Operations）？

上述的问题可以说得更白话些，或许系统分析员可以向业务人员作如下的提问：

- 某物会记录什么数据呢？（探问属性）
- 某物可以提供我们哪些数据呢？（探问属性）
- 通过某物，可以让我们查到哪些数据吗？（探问属性）
- 某物可以做什么用呢？（探问操作）
- 有了某物之后，我们可以拿它来做什么事呢？（探问操作）

假设我们要开发基金交易平台，现在来向业务人员进行访谈，期间提到了“基金账户”。想象一下，系统分析员跟业务人员之间的模拟对话，大概会像这样子：

系统分析员问：基金账户会记录什么数据呢？（探问属性）

业务人员答：基金账户里头主要会记录总成本、总现值、总损益、总报酬率。

系统分析员问：有了基金账户之后，我们可以拿它来做什么？（探问操作）

业务人员答：有了基金账户之后，只要里头存有足够的钱，你就可以用它来申购基金，或者赎回基金。

访谈之后，系统分析员便得到了这个项目属性与操作的概貌，如图1-1所示。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

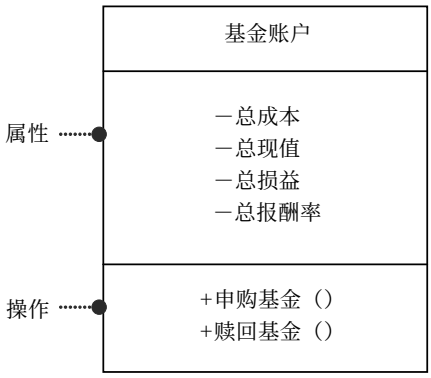

图1-1　基金账户的属性与操作

针对对象的各项属性，系统分析员还需要进一步了解它在企业里的定义、数据类型（Data Type）、可能的范围值或者初始值，更别忘了了解这项属性是怎么跑出来的？或许，系统分析员可以向业务人员作如下的提问：

- 可以请您（业务人员）用简单的一、两句话，解释某属性是什么吗？（探问属性定义）
- 可以请您举个例子吗？（判断属性的数据类型）
- 请问某属性有范围值吗？（判断属性的数据类型以及字段大小）
  - √ 可被接受的数字，最大最小为何？（数字类型）
  - √ 可被接受的字符串，最长最短为何？（字符串类型）
  - √ 预设的项目，有哪几个？项目异动的频率？（枚举类型）
- 请问某属性有初始值吗？（探问属性的初始值）
- 怎样做才能够得到某属性值（Attribute Value）？（探问属性值的获得方法）
  - √ 请问谁会提供这项属性值？（键入值）
  - √ 请问可以向哪里查询这项属性值？（查询值）
  - √ 请问计算公式为何？（计算值）
  - √ 请问可有独特的编码方式？（流水码或特定编码）

1.4.3　操作与方法

针对对象的操作，系统分析员还必须进一步探问how，了解操作的实现方法（Method）。简言之，操作是对象的what，而方法则是对象的how。然而，在我们用程序实现出对象之前，对象是死的，它哪会有什么方法来执行操作。所以，我们其实是想获知业务人员惯用的操作方法，然后将人为的操作方法转移给对象，成为对象的操作方法。

系统分析员访谈业务人员时，主要得获知方法的执行步骤（Procedure）、所需或者产生的数据、计算公式，以及企业的特殊约束。也许，系统分析员可以参考下列问题，向业务人员提问：

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

- 您（业务人员）通常都怎么执行某操作的呢？可以告诉我主要的执行步骤吗？（探问执行步骤）
- 请告诉我这些执行步骤会用到什么数据？以及会生成什么数据？（探问数据的输入及输出）
- 请告诉我这些执行步骤会需要使用到计算公式吗？（探问计算公式）
- 在执行某操作时，有没有什么重要的约束需要注意或遵守的？（探问特殊约束）

假设针对基金账户对象的“申购基金”这项操作的执行步骤，系统分析员向业务人员访谈的过程中，可能出现如下述的模拟对话：

系统分析员问：您通常都怎么执行“申购基金”这件事呢？可以告诉我，主要的执行步骤吗？（探问执行步骤）

业务人员答：是这样的，申购基金分为两种：一种是单笔申购，另一种是定期定额申购。

系统分析员问：谁在什么时候会决定是哪一种？还是可以同时包含两种？（判断是否要拆成两项操作）

业务人员答：客户一开始在申购基金时，就得二选一，决定是要单笔申购某档基金，还是定期定额申购。

（对于客户而言，单笔申购基金或者定期定额申购基金是两项不同的目的，所以系统分析员决定将原先的图1-1里的申购基金操作一分为二，改成图1-2的模样。）

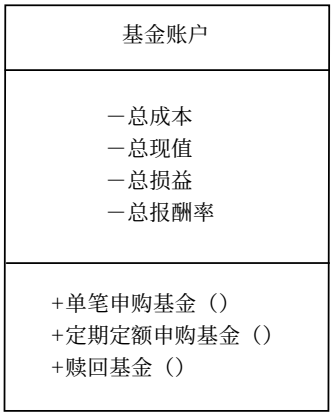

图1-2 申购基金操作一分为二

系统分析员问：请您挑比较简单好懂的一种，告诉我您主要的执行步骤？（探问执行步骤）

业务人员答：都很好懂啦，我就先说单笔申购的情况，好了。客户告诉我们单笔申购的基金名称、信托金额以及扣款账号，那我们这边只要确认这档基金有贩卖，然后客户指定的扣款账号里头有足够的余额可以付信托金额和申购手续费，这样就可以申购了。确认申购之后，我们会从客户指定的扣款账号中，支出信托金额和申购手续费。最后，我们会给客户一个申购交易的凭证号码，就完成单笔申购了。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

系统分析员记录“单笔申购基金”的主要执行步骤如下：

1. 客户告知（基金名称）、（信托金额）以及（扣款账号）。

2. 业务人员确认（该档基金销售中）。

3. 业务人员确认客户指定的（扣款账号）里头有足够的（余额），可以用来支付（信托金额）和（申购手续费）。

4. 确认申购之后，我们会从客户指定的（扣款账号）中，支出（信托金额）和（申购手续费）。

5. 最后，我们会给客户一个（申购交易的凭证号码），就完成单笔申购了。

随后，系统分析员可以请业务人员协助确认这份记录，同时给予补充。但是，对于单笔申购基金的操作方法还没访谈完，系统分析员还得获知并确认输出入数据、计算公式以及特殊约束。

1.4.4　封装

面对真实世界中的多数物品，我们是“不知”亦能用，多数事件，我们也是“不知”亦能行。当然，我们对多数的事物也不是全然无知，只不过通常是有限度的知。因此，在物品或事件所蕴含的细节都被封装（Encapsulation）起来的情况下，我们还是可以使用物品、参与事件，毫无困难。

所以，当我们打造软件对象来仿真真实世界里的事物时，也模拟了这种封装特性。由于软件对象的封装性，所以对象之间仅透露足以进行交互的低限信息。对于对象的封装性，系统分析员要掌握下列要点：

- 已知操作。对象通常仅对其他对象透露自身的操作，彼此之间通过调用（Call）已知的操作来交互。
- 封装属性。每个对象封装属性值，不透露给其他对象。
- 封装方法。每个对象封装方法，仅对其他对象透露操作，但不透露其方法。

因此，系统分析员要特别注意，在分析规划对象的方法时，如果需要与其他对象交互，甚至是使用到对象本身的属性或操作时，切记要严守下列三条：

1. 不得直接提及对象的属性。

2. 也不得假设对象的执行方法。

3. 仅能够使用对象的操作。

从上述可以发现，我们为了保护对象的封装特性，会牺牲掉许多便利性。好比喝冰冻饮料时不能打开杯盖直接畅饮，而非得戳一个小洞，通过吸管小口小口吸取，十分不便。但使用吸管有一个好处，不小心打翻时，不会溅得四处都是，然后还得花钱再买一罐。

严守对象的封装性，有一个好处，当需求发生变化而需要改写代码时，变化会被局限在对象的属性和方法中，不会起涟漪效应，也不会发生牵一发而动全身的连锁反应。由于软件内部

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

的组成对象易于汰旧换新，所以软件的使用寿命延长，后续的维护成本也偏低，企业因此得到高投资报酬率，不过眼前首先要付出的代价是较高的开发成本。

1.4.5 类

“物以类聚”点出了对象（Object）与类（Class）之间的关联。针对一群相似的对象，我们本能地将它们视为一类（同一类）。访谈时，系统分析员听着业务人员谈论某一事物，甚至请他举出两三个具体且容易理解的实例，试图通过理解且分析这两三个同类型的实例，定义出适用于这些实例的类。

简言之，系统分析员通过分析真实世界中同类型的实例，以便定义出适用于软件世界的类。经过设计之后，程序员会为这些类编码。随后在软件系统执行期间，系统内部将遵照各类独自的定义，创建各类不同功用的对象。最后，这些不同类的对象将调用彼此的操作，合力完成各项系统功能。基金账户类与对象如图1-3所示。

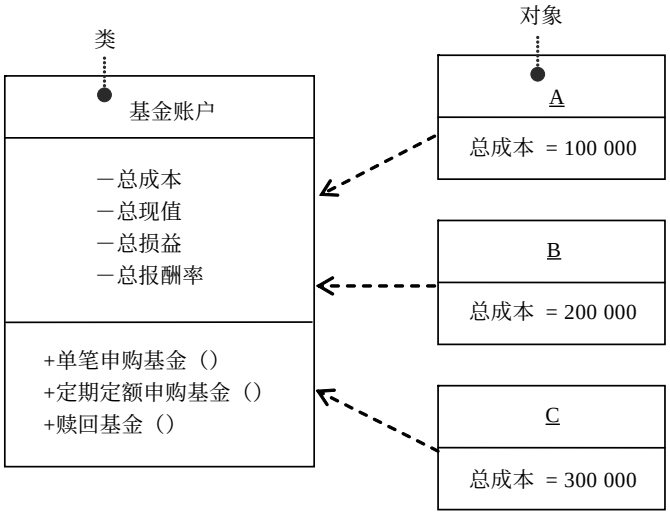

图1-3 基金账户类与对象

所以，一个简化后的开发程序以及系统运作情况，大致如上所述。归结起来，类与其对象之间细微的关联如下：

- （类）定义属性与操作，且所属（对象）共有这些属性与操作。
- 虽然同类（对象）共有属性，可是每一个（对象）却独有属性值。每个基金账户对象共有“总成本”这项属性，可是每个基金账户对象的总成本值都不同。请看图1-3，A基金账户的总成本值是100 000元，B基金账户的总成本值是200 000元，C基金账户的总成本值是300 000元，每个基金账户对象独有自己的总成本值。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

- 因为同类（对象）共有操作和方法，所以它们可以做相同的事情，而且有相同的作法。
- 稍后我们还会提到，（类）也定义关系（Relationship），且所属（对象）共有这些关系。不过，如同属性与属性值的情况，虽然同类（对象）共有关系，可是每一个（对象）却独有关系值。

1.4.6　泛化关系

系统分析员通过类定义属性和操作，所以日后针对同类的一群对象，就可以使用相同的方式对待它们。不过有时候，这些对象并不是全然相同，可能大部分的属性和操作相同，但是少部分的属性和操作却不同。在这种情况下，类之间的泛化关系（Generalization）就派上用场了。

现在我们先来看一段模拟情景，体会系统分析员在访谈期间可能遇到的情况，随后我们才说明如何使用泛化关系。当系统分析员访谈到申购基金的主题时，立即理解到“申购交易”是一项很重要的概念。所以，系统分析员检查了成为候选对象的两项条件，我们在前面曾提及：

1. 在企业运作过程中，业务人员会使用到的专业事物或概念。

2. 而且，在信息化时，系统也会用到，或者需要保存。

客户申购基金时，业务人员确实会使用到申购交易的概念，符合前述条件1。而且，在信息化时，系统也需要保存每一笔的申购交易，符合前述条件2。“申购交易”同时符合上述两项条件，所以系统分析员将它列为候选对象，并且让所有的申购交易对象归属于同一类，如图1-4所示。

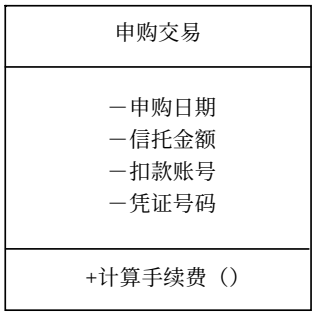

图1-4　申购交易类

业务人员说：申购交易有两种，一种是单笔申购，另一种是定期定额申购。

系统分析员问：这两种有什么不同呢？

业务人员答：单笔申购比较简单，就跟我们平常买东西，一次买断的情况相同。客户单笔申购某档基金100 000元，买断，我们给客户申购交易的凭证号码，交易就结束。定期定额申购，比较像是一种约定，客户可以跟我们约定每月的15号，自动从指定扣款账户支出5 000元申购某档基金。每月15日定期，每次5 000元定额，指定某档基金，指定某个扣款账户，经过

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

这样的约定之后，每个月就会定期定额自动支出一笔资金来申购基金。

系统分析员问：那客户可以更改定期定额的约定吗？

业务人员答：随时可以改啊！

系统分析员问：可以更改哪些项目？

业务人员答：可以改扣款日期、投资金额、扣款账号，还有扣款情况。

系统分析员定义出三个申购类，相较起来，申购交易属于一般类（Generalized Class），单笔申购与定期定额申购属于特殊类（Specialized Class），如图1-5所示。

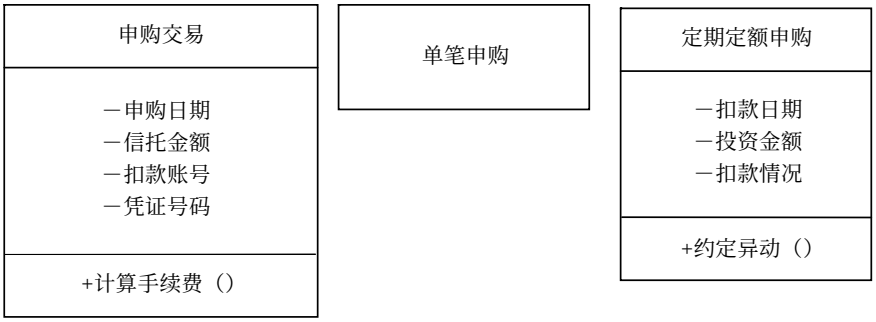

图1-5 三个申购类

因为，系统分析员把单笔申购与定期定额申购里，通用的属性与操作记录到申购交易类，所以申购交易类定义了通用且一般性的属性与操作。单笔申购与定期定额申购两者独有且特殊的属性与操作，当然还是记录在两者自己的类里。所以，相对于申购交易的一般性，我们将申购交易归属于一般类，而将单笔申购与定期定额申购归属于特殊类。

系统分析员可以通过检查下列两项条件，判断是否采用泛化关系：

1. 在企业领域的专业概念里，特殊对象必须“是一种”（a kind of）一般对象。

2. 多种特殊对象里，有部分通用的属性与操作，也有部分独有的属性与操作。

上述的“是一种”条件，必须遵守业务概念。举例来说，我们可以接受“休旅车是一种汽车”，但是很难接受“遥控车是一种汽车”。因为遥控车是玩具，不是汽车，或许说“遥控车是一种玩具汽车”，会比较符合企业领域里的专业概念。

请看图1-6，系统分析员经过下列思考决定使用泛化关系：

- 单笔申购是一种申购交易，定期定额申购也是一种申购交易。或者说，申购交易可以分为两种，一种是单笔申购，另一种是定期定额申购。（符合上述条件1）
- 单笔申购与定期定额申购里，都需要记录申购日期、信托金额、扣款账号、凭证号码这四项属性，以及必须执行计算手续费这项操作，这些属性与操作属于通用部分。看起来单笔申购确实比较单纯，没有特殊的属性和操作。不过，定期定额申购就不是这么单纯了，它多了扣款日期、投资金额、扣款情况这三项属性，以及多了一个约定异动的操作。（符合上述条件2）

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

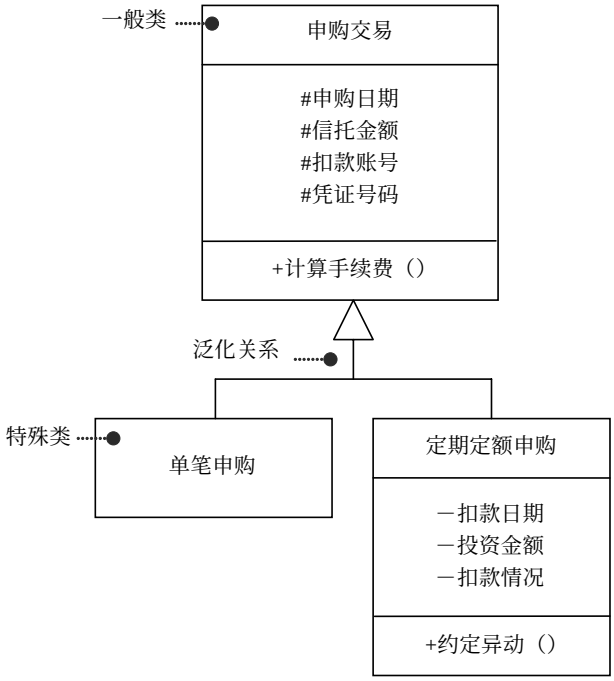

图1-6　类之间的泛化关系

在泛化的过程中，我们将特殊类里头通用的属性和操作记录到一般类里，因此通过泛化关系，特殊类可以继承（Inheritance）一般类里的通用属性与操作。由于特殊类通过继承，可以直接重用（Reuse）一般类里的属性、操作和方法，节省开发成本。所以当发生变化需要改版时，也只需要改版一般类，节省维护成本。

请看图1-7里的特殊类，我特别将继承来的属性与操作一并列出，而且使用底纹区分。

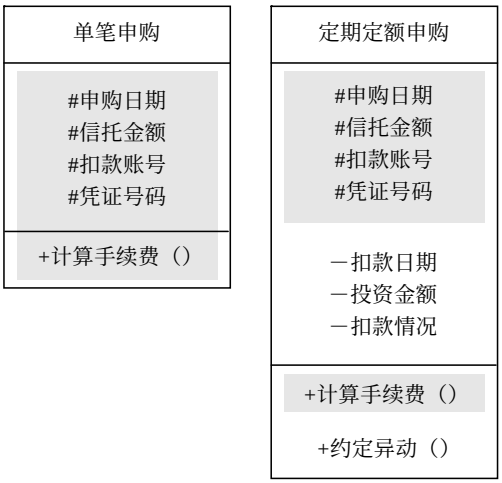

图1-7　从申购交易类继承而来的属性与操作

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

1.4.7 关联关系

类之间最常见的关系，不是上述的泛化关系，而是此处的关联关系（Association）。系统分析员可以通过检查下列两项条件，判断是否采用关联关系：

1. 在企业领域的专业概念里，两种对象之间有一种固定不变且需要保存的静态关系。
2. 在信息化时，系统会用到这些静态关系，而且必须将它们存到数据库。

请看图1-8，针对基金账户与申购交易之间的关联关系，系统分析员做了如下的思考：

- 发生任何一笔申购交易，都必须确实记录在某一基金账户底下。申购交易与基金账户之间的关系，固定不变，且在该基金账户被结清之前，两者之间的关系需要被永久保存下来。（符合上述条件1）
- 由于，申购交易与基金账户之间的关系需要被保存下来，所以会被存到数据库里。（符合上述条件2）
- 每一个基金账户底下，可以有零到多笔的申购交易。所以，在申购交易端标示（*），代表一个基金账户对象可以链接（Link）零到多个申购交易对象。
- 每一笔申购交易，只能登记在某一个基金账户底下。所以，在基金账户端标示（1），代表一个申购交易对象只能链接1个基金账户对象。

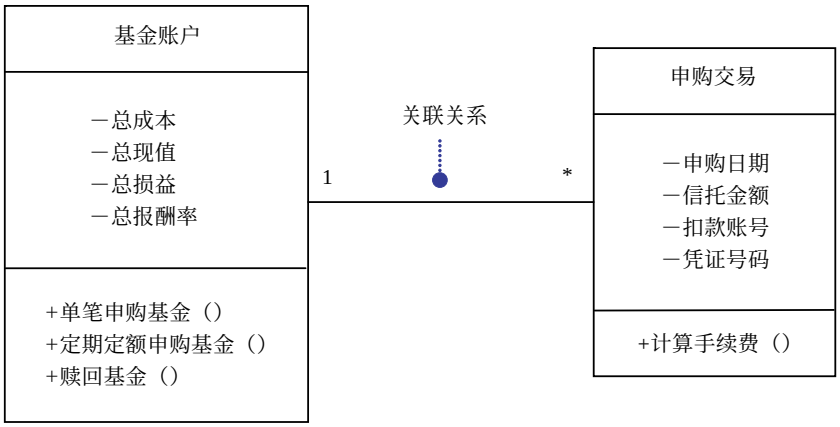

图1-8 关联关系

1.4.8 聚合关系

聚合关系（Aggregation）是一种特殊的关联关系，所以它继承了关联关系的特质，而且还独有“整体-部分”（Whole-Part）的特质。简言之，聚合关系两端的对象，需具有Whole-Part的关系。系统分析员可以通过检查下列三项条件，判断是否采用聚合关系：

1. 在企业领域的专业概念里，两种对象之间有一种固定不变且需要保存的静态关系。（继

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

承自关联关系的条件）

2. 在信息化时，系统会用到这些静态关系，而且必须将它们存到数据库。（继承自关联关系的条件）

3. 在企业领域的专业概念里，两种对象之间有Whole-Part的静态关系。（聚合关系独有的条件）

在我们的基金模拟项目中，有“基金看台”的概念。每一个基金看台可以设置多档自选基金，如此一来，客户就可以同时观看并比较多档基金的报酬率。请看图1-9，假设我在彰化银行、新光银行、永丰银行都有申购基金，所以我设置了三个基金看台，这样就可以同时间观看这三个基金账户里头的基金资料。由于，三个基金账户里头都有申购过安本亚太基金，所以这档基金会同时出现在三个基金看台中。

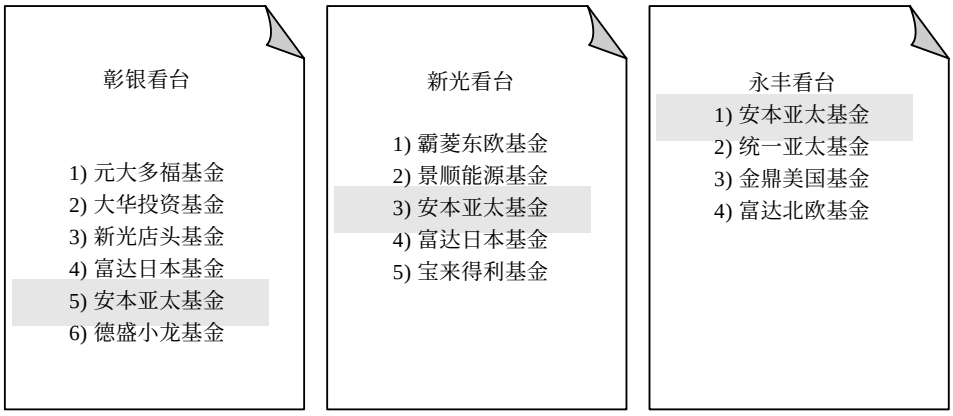

图1-9　一个Part对象可以链接多个Whole对象

请看图1-10，针对基金看台与自选基金之间的聚合关系，系统分析员做了如下的思考：

- 基金看台与自选基金之间的关系，固定不变，且在该基金看台被整个删除，或者某自选基金从该基金看台除名之前，两者之间的关系需要被永久保存下来。（符合上述条件1）
- 由于基金看台与自选基金之间的关系需要被保存下来，所以会被存到数据库里。（符合上述条件2）
- 基金看台聚集了多档自选基金。自选基金是基金看台里很重要的一项元素，而且可以说，如果没有自选基金，基金看台几乎也就失去了存在价值。所以，两者之间有着Whole-Part的特质，基金看台是Whole，自选基金是Part。（符合上述条件3）
- 每一个基金看台底下，可以有零到多档的自选基金。所以，在自选基金端标示（*），代表一个基金看台对象可以链接零到多个自选基金对象。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

• 每一档自选基金，可以设定在多个基金看台底下。所以，在基金看台端标示（*），代表一个自选基金对象可以链接零到多个基金看台对象。

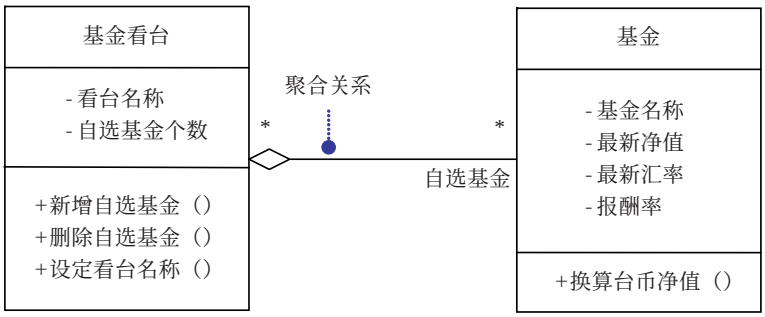

图1-10 聚合关系

1.4.9 组合关系

组合关系（Composition）是一种特殊的聚合关系，所以它继承了关联关系，以及聚合关系的“整体-部分”（Whole-Part）的特质，还独有全然拥有Part对象的特质。系统分析员可以通过检查下列四项条件，判断是否采用组合关系：

1. 在企业领域的专业概念里，两种对象之间有一种固定不变且需要保存的静态关系。（继承自关联关系的条件）

2. 在信息化时，系统会用到这些静态关系，而且必须将它们存到数据库。（继承自关联关系的条件）

3. 在企业领域的专业概念里，两种对象之间有Whole-Part的静态关系。（继承自聚合关系的条件）

4. Part对象只能链接一个Whole对象，且Whole对象被注销（Destroy）时，Part对象必须一块被注销。（组合关系独有的条件）

在基金模拟项目中，定期定额申购是很重要的概念。客户约定了定期定额申购之后，银行每月就会按照约定自动成立一笔小额的申购交易，而且每期申购的基金单位数都会被累加在一起。特别的是，自动成立的单期交易没有自己的凭证号码，而是共有原先定期定额申购约定时的凭证号码。

请看图1-11，针对定期定额申购与单期交易之间的组合关系，系统分析员做了如下的思考：

• 定期定额申购与单期交易之间的关系，固定不变且需要被永久保存下来。（符合上述条件1）

• 由于，定期定额申购与单期交易之间的关系需要被保存下来，所以会被存到数据库里。（符合上述条件2）

• 客户约定了定期定额申购之后，每月就会按照约定自动生成一笔小额的申购交易，所有

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

的单期交易都归属于同一个定期定额申购交易底下。两者之间有着Whole-Part的特质，定期定额申购是Whole，单期交易是Part。（符合上述条件3）

- 每一个单期交易对象只能链接一个定期定额申购对象，且定期定额申购对象被注销时，单期交易对象必须一块被注销。（符合上述条件4）
- 每一个定期定额申购底下，可以有零到多个的单期交易。所以，在单期交易端标示（*），代表一个定期定额申购对象可以链接零到多个单期交易对象。
- 每一笔单期交易，只能隶属于某一个定期定额申购底下。所以，在定期定额申购端标示（1），代表一个单期交易对象只能链接1个定期定额申购对象。

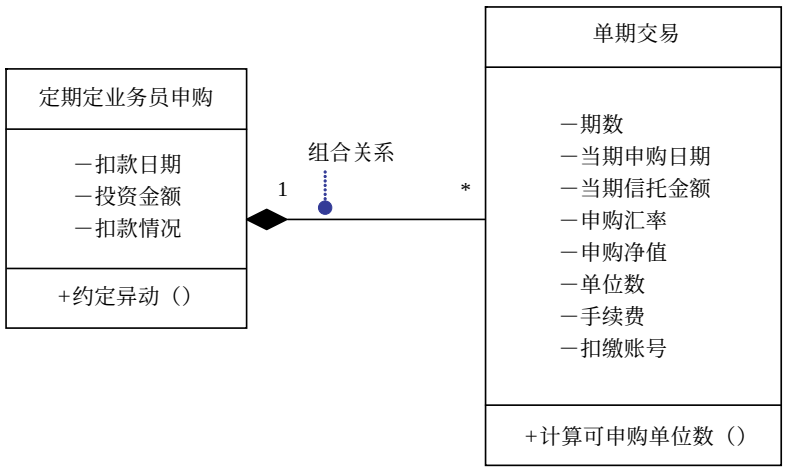

图1-11　组合关系

1.4.10　用例与执行者

用户因为某种目的而来使用系统，这样的一段交互（Interaction）过程，就是一个用例（Use Case）。而且，因为用户在这过程中会与系统交互、参与用例，所以特别称为执行者（Actor）。

采用用例的技术，可以引导系统分析员站在用户的角度来描绘系统，开发出用户合意的系统。用户通常在意系统的What，而非系统的How。他们想要知道系统提供什么样服务，以及该如何与系统交互才能够获取这些服务，但是不需要知道系统内部的执行细节。

请看图1-12，例如在基金模拟项目中，投资人会在意网上基金系统有没有提供申购基金的服务，如果有，那需要通过什么样的操作步骤才能够网上下单申购基金。但是，投资人不会想要知道网上基金系统内部的执行细节，既然不知亦能用，那又何必知道呢！

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

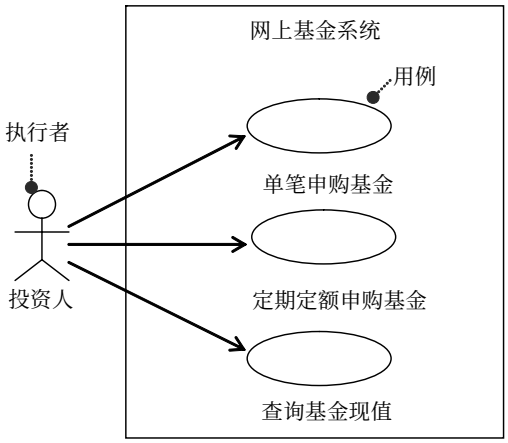

图1-12 用例图

1.4.11 业务用例与系统用例

用例可以用来表达用户与信息系统（System）的交互过程，也可以用来表达顾客与企业组织（Business）的交互过程，为区分两者，特将前者称为“系统用例”（System Use Case），后者称为“业务用例”（Business Use Case）。想当然尔，执行者也跟着区分为“系统执行者”（System Actor）与“业务执行者”（Business Actor）。

在基金模拟项目中，有些投资人不会使用计算机，所以不可能使用网上基金系统申购基金。这些投资人会到银行柜台，委托理财专员代为申购基金，其间的交互就是一项业务用例。请看图1-13里的申购基金（业务用例），表达了投资人（业务执行者）为了申购基金，与银行（企业）交互的过程。

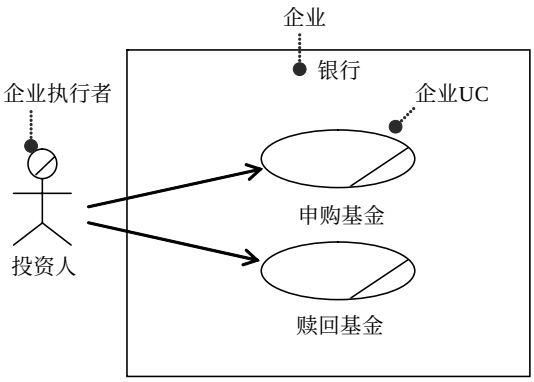

图1-13 业务用例图

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

在系统开发初期，系统系统分析员将定义并描述业务用例，不过不是全部的业务用例，而是日后系统会涉及到的业务用例。接着，系统分析员进一步分析每一个业务用例内部的执行活动，从中圈选出可以交给系统执行的活动，并将这些可自动化的活动定义成系统用例。

接着，这些系统用例将引导整个开发程序。通常，项目会从中挑选一批系统用例作为首次发布（Release）的范围，随后系统分析员才针对这些系统用例进行深度访谈，理清需求细节且编写文件，递交给其他开发人员进行后续的系统设计及编码工作。

1.5　MDA开发程序

本书采用MDA（Model-Driven Architecture）开发程序，作为系统分析员进行分析工作，以及生成UML模型的依据。MDA与UML同为OMG（Object Management Group）机构之标准。MDA主要将生成的UML模型，分为下列三个阶段：

- CIM（Computation Independent Model）—— 聚焦于系统环境及需求，但不涉及系统内部的结构与运作细节。
- PIM（Platform Independent Model）—— 聚焦于系统内部细节，但不涉及实现系统的具体平台（Platform）。
- PSM（Platform Specific Model）—— 聚焦于系统落实于特定具体平台的细节。例如，Spring、EJB2或.NET都是一种具体平台。

最后，程序员会依据PSM的UML模型内容，按图施工，编写出适用于特定具体平台的代码。

1.5.1　MDA的主张

如何应对企业与技术的快速变化，一直是软件界的专家学者们伤脑筋的问题。然而，此刻OMG（Object Management Group）所提出的MDA（Model-Driven Architecture），便是为了解决这个变化问题。

- MDA欲解决的问题 —— 如何应对企业与技术的快速变化？
- MDA所用的工具 —— 运用OMG现有的标准及技术，主要包括有：UML（Unified Modeling Language）、MOF（Meta-Object Facility）、CWM（Common Warehouse Metamodel）、UML Profile、XMI（XML Metadata Interchange）以及CORBA。
- MDA提出的解决方法 —— 将企业及应用系统与实现技术平台分离，且以统一建模语言UML来表达与平台无关的PIM（Platform Independent Model），然后再设计出适用于特定平台的模型PSM（Platform Specific Model）。如此一来，因为分隔且封装了企业与技术两方面的变化，所以降低了两者之间的牵动。

MDA主张将设计切分成PIM和PSM，除此之外，MDA其实没有额外提出其他的标准或技术，取而代之的是，它善用且整合多项已经存在的标准及技术。请看图1-14，这是MDA官方网

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

站（http://www.omg.org/mda/）首页上的图片，充分呈现了MDA的期望与相关技术。

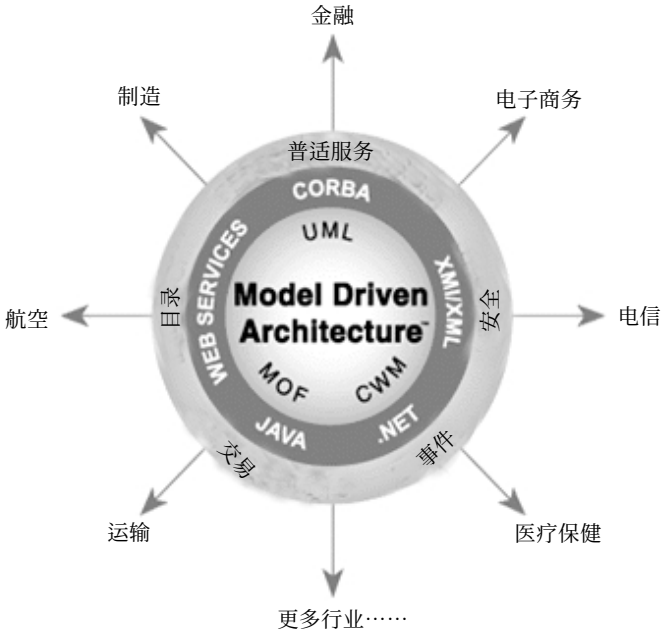

图1-14 MDA

从图1-14最内圈看起，MDA主要使用了UML、MOF及CWM这三项建模标准（Modeling Standard），作为PIM及PSM的建模基础。不过，对于一般的开发人员而言，只要熟知UML这套统一建模语言，就可以开发MDA项目了。

往外看图1-14中的第二内圈，代表公开标准或私有的实现技术平台，有CORBA、XMI/XML、.NET、Java及Web Services，等等。也就是说，MDA希望制定出各式独特的具体平台专属的PSM转换规则，并且最好可以由厂商配合设计出MDA开发工具，以便能够将中立的PIM自动转出特定平台的模型PSM。

接下来再往外看到图1-14中的第三内圈，代表跨平台的普适服务（Pervasive Services），有目录服务（Directory Services）、交易服务（Transaction Services）、安全服务（Security Services）以及分布式事件及通知服务（Distributed Event and Notification Services）。OMG计划定义四项普适服务，让任何平台上的应用程序或Client端都可以通过MDA环境，取得跨平台的服务。

至于图1-14中的最外围，则代表MDA可以应用在各式不同的领域环境中，诸如电子商务（E-Commerce）、电信（Telecom）、运输（Transportation）、制造（Manufacturing）、医疗保健（Healthcare）、金融（Finance）以及航空（Space）等领域。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

1.5.2　程序

MDA项目开发的第一步骤从CIM开始。不同于PIM与PSM，CIM试图表达信息系统的应用环境，而非信息系统本身。以银行的基金系统为例，CIM表达的对象是银行的基金业务及组织运作，而PIM与PSM则表达支持银行基金业务的信息系统。

开发团队在进行CIM时，关切的是与企业相关的营运目标、实现条件及运作流程等，先了解信息系统的应用环境，才有可能为企业量身打造出完善的信息系统。

在经历建构CIM的过程中，开发团队除了可以逐步了解企业，同时也建立与业务人员之间的沟通方式及默契，还让业务人员可以参与信息系统的开发。这就好比在进行室内装潢之前，好的室内设计师会与业主沟通，充分了解业主的期望，以及家中成员的目前状况与未来展望，甚至邀请业主一块参与设计，如此才能够量身订做出妥贴的室内空间。

PIM与PSM之间的界限，比较容易混淆，两者所关切的主体都是信息系统，分野的界限在于“平台”（Platform）一词。PIM重视信息系统里重要的运作与结构，CIM旨在记录企业领域里的重要需求与概念，两者之间的界限十分清楚。以平台作为PIM与PSM两者的分野，PIM表达的设计必须无关乎或独立于任何一个特定的平台，而PSM恰好相反，它必须要能够真正落实及适合某一个特定的平台。

其实，平台的概念可以再进一步细分为抽象平台（Abstract Platform）与具体平台（Concrete Platform）。一般而言，PIM可以建构于抽象平台上，但绝对不可以建构于具体平台上，只有PSM则才可以建构于具体平台上。如果没有特别区分两者，通常指的是具体平台，像是前述的平台以及MDA所指称的平台，都是归类为具体平台。

开发MDA项目时，开发团队通常会先以CIM与抽象平台为信息来源，整合设计出PIM。随后，才以PIM与具体平台为信息来源，整合设计出PSM。CIM、PIM、PSM与平台之间的关连，如图1-15所示。

简言之，PIM与PSM的界限在于，是否支持特定的具体平台。前者与具体平台无关，后者则得适合于某一个特定的具体平台。如此一来，PIM阶段所生成的UML模型便可以由日后不同的具体平台所支持，也因此产生不同具体平台版本的PSM设计。

以室内设计为例，设计师先规划出如图1-16的设计图，但不指定实际施工的建材与细节，所以这是一张PIM阶段的室内设计图。在这张设计图里的右下角有一个衣柜组，但此处并不指定该衣柜组的实际施工的建材及方式。假如，业主的预算不够宽裕，可以选择系统家具的方式来施工；若是比较讲究的业主，可能会选择手工打造。此处参考的图1-16和图1-17这两张设计图，下载自某3D室内设计公司（http://www.3899.idv.tw/）的网站。

请看图1-17的衣柜施工图，详细描述了衣柜的规格和建材，像是衣柜的拉门外面必须贴上胡桃木夹板，所以这是一张PSM阶段的衣柜施工图。依据这张详细的衣柜施工图，木工师傅就可以按图施工。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

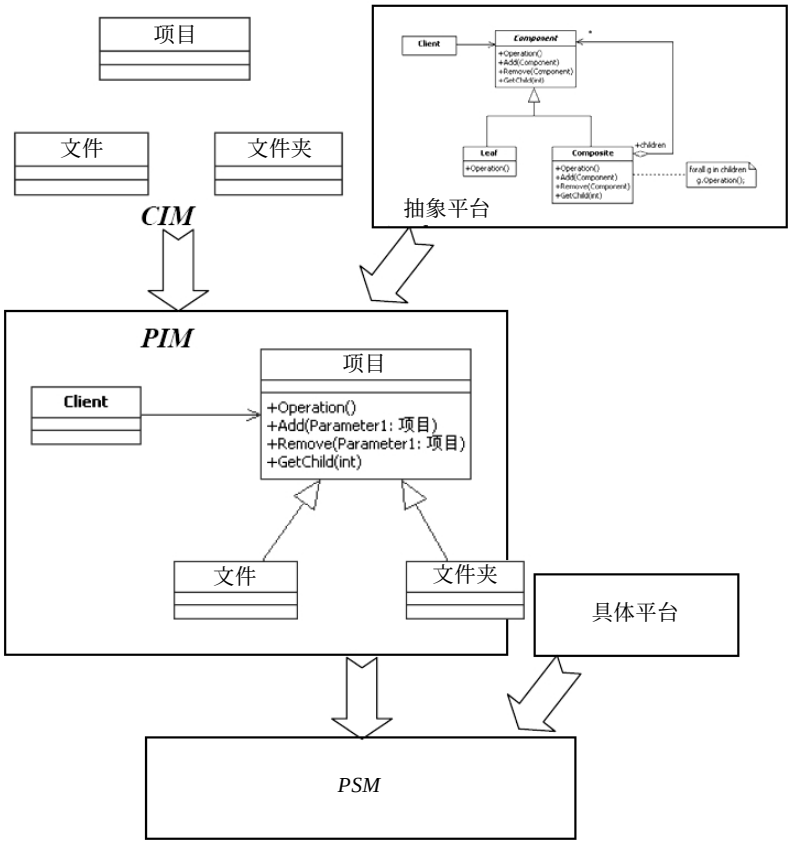

图1-15 从CIM经PIM到PSM

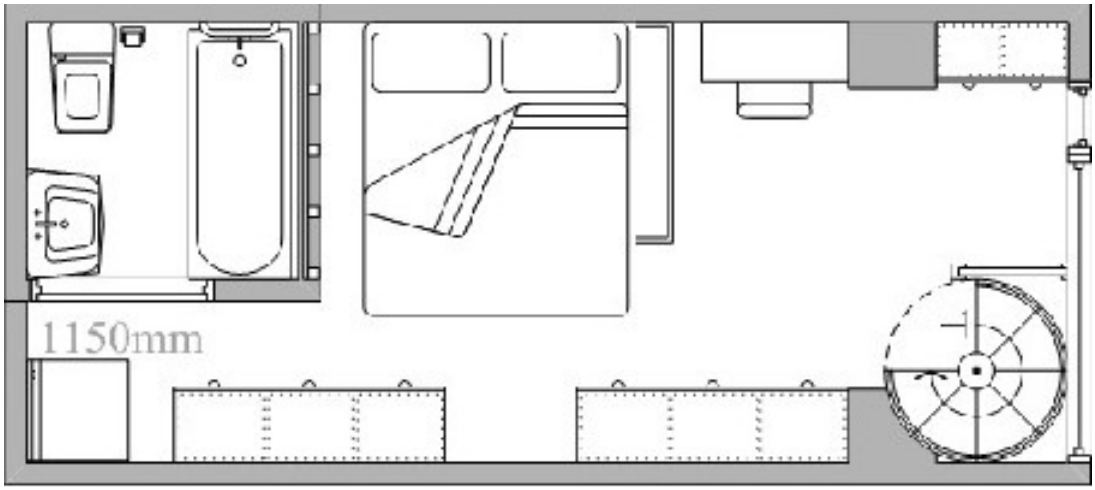

图1-16 PIM阶段的室内设计图

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

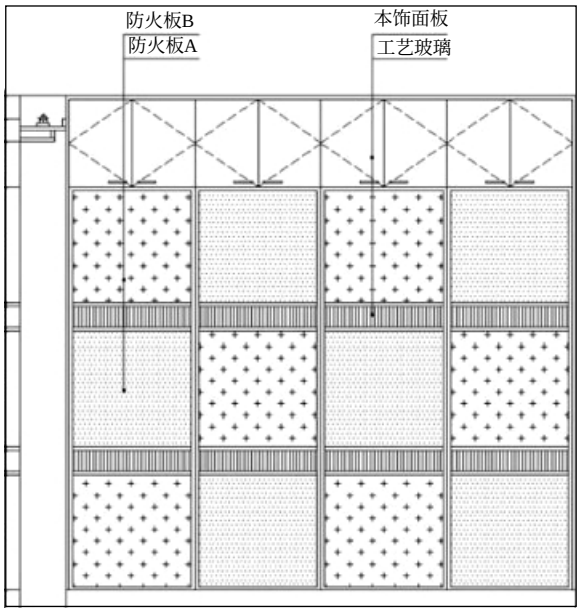

图1-17　PSM阶段的衣柜施工图

1.5.3　MDA在芯片设计的应用

虽然MDA的应用起源于一般的商用信息系统，可是它的魅力不断扩大，所以其他领域的应用也急起直追，例如在芯片设计的应用上，就是令人惊艳的后起之秀，值得我们观摩学习之。

在芯片系统的开发上，采用了MDA开发程序之后，同样会生成下列三阶段的UML模型：

- CIM（Computation Independent Model）—— 聚焦于芯片系统环境及需求，但不涉及芯片系统内部的结构与运作细节。
- PIM（Platform Independent Model）—— 聚焦于芯片系统内部细节，但不涉及芯片系统的具体平台（Platform）。
- PSM（Platform Specific Model）—— 聚焦于芯片系统落实于特定具体平台的细节。例如，Java、C/C++、SystemC、Verilog等都是一种具体平台。

随后，软硬件工程师会依据PSM的UML模型内容，按图施工，编写出适用于特定具体平台的代码，或者通过支持MDA的开发工具，自动生成诸如Java、C/C++、SystemC、Verilog等软硬整合的代码或电路图。

接着，我们来看一、两项比较有名的应用。首先，我们来看法国国立信息与自动化研究所（INRIA）在一份2004年的《MDA for SoC Design, UML to SystemC Experiment》研究报告中，提到他们在开发ISP（Intensive Signal Processing）项目时，不仅采用了MDA技术，他们还进一步提出一个Y型方法来搭配运作，如图1-18所示。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

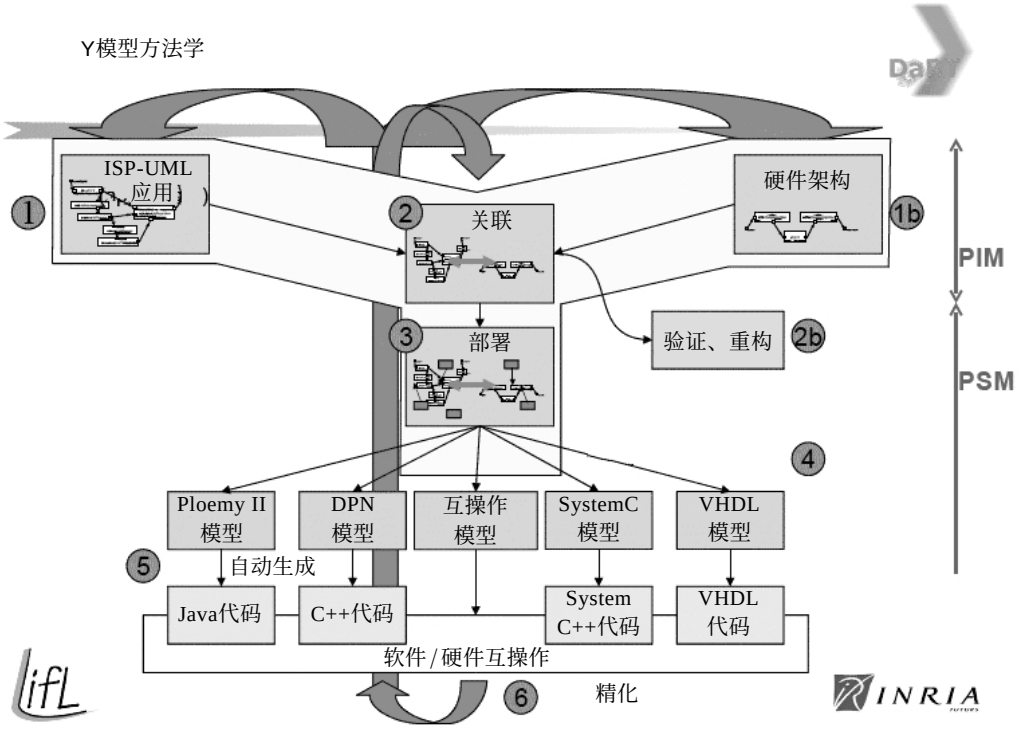

图1-18 Y型方法

而且，这个团队还开发了一个可以转换PSM模型的软件，若您有兴趣深入研究，可到ModelTransf网站（http://www.lifl.fr/west/mdaTransf）下载相关数据，以及这个模型转换工具的源代码。

最后，我们来看业界的另一项应用实例。在2006年召开的“UML for SoC Design Workshop”会议上，意法半导体公司在《A SoC design flow based on UML 2.0 and SystemC》的报告中，提到他们成功地扩展了EA（Enterprise Architect）这套UML开发工具，使得能够从UML模型自动生成SystemC代码，达成MDA中PSM模型自动转出代码的部分，如图1-19所示。

1.5.4 本书所采用的分析步骤

依据MDA，本书所提及的步骤及生成，归属于CIM与PIM阶段，并未涉及PSM阶段。如下：

- CIM-1：定义业务流程，产生业务用例模型。
- CIM-2：分析业务流程，产生活动图。
- CIM-3：定义系统范围，产生系统用例图。
- PIM-1：分析系统流程，产生系统用例叙述。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

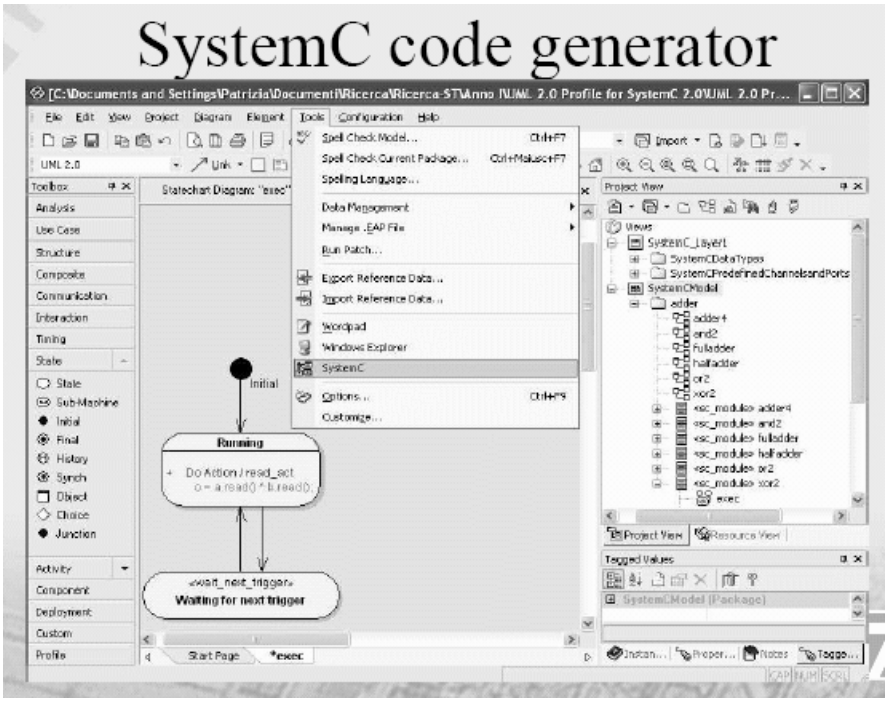

图1-19　UML自动生成SystemC

- PIM-2：分析业务规则，产生状态图。
- PIM-3：定义静态结构，产生类图。
- PIM-4：定义操作及方法，生成序列图。

在CIM阶段，系统分析员大约花1~2周的时间，尽快生成初步的系统用例，以便让相关的决策人员可以从中挑选出首期开发的系统用例，而这也就是首期的系统范围。

随后，项目正式进入PIM阶段，也是正式进入分析阶段，所以系统分析员将投入更多的时间，针对首期的系统用例详述细部规格，作为正式需求文件的一部分，也作为业务人员与开发人员之间的沟通文件。

此外，系统分析员需多加注意，CIM阶段与PIM阶段的生成方式略有不同。系统分析员在结束CIM阶段之后，才决定出PIM阶段的系统范围，也同时正式进入PIM阶段。但是，在进入到PIM阶段之后，系统分析员将所有系统用例依相关性分成若干组，以组别方式生成该组系统用例涉及的PIM-1~4结果，随后交给后续的开发人员进行设计、编码及测试。然后，逐步生成一组一组的PIM-1~4结果，跟CIM的生成方式不同。

1.6　UML对MDA的帮助

UML所具备的多项特性，更使得它成为开发MDA项目不可或缺的要素之一。UML有助于

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

发展MDA项目的几项重要特性，包含有：

- 中立机构负责维护UML —— UML是OMG的公开标准，并非私人的知识产权。
- 中立的建模语言 —— UML是统一建模语言（Unified Modeling Language）的缩写，顾名思义，它是建构模型（Model）的专用语言，并不指定使用于任何特殊的应用领域、具体平台、实现语言或实现方法。
- Profile支持定制化UML方言 —— 虽然UML天性中立，可是通过UML所提供的Profile机制，却可以让我们自创独特的UML方言，以便应用于特定应用领域、具体平台、实现语言或实现方法上。

1.6.1 中立机构负责维护UML

由于UML非私人企业所拥有，它是OMG（Object Management Group）的公开标准，欢迎公众使用。因此，许多专家学者喜欢自行开发工具来支持UML，很多工具都是免费的，甚至是开放源码（Open Source）。

一旦这些UML工具有提供Profile机制，便可以让我们创出各式UML方言（UML based-language），用以表达不同平台或不同领域中特殊的概念。换言之，所有这些付费、免费或开放源码的UML工具，都是MDA的雏形工具，也都具备了成为MDA工具的基础功能。

请看图1-20网页的最上方，中间UML立体字样的图片是UML的商标，UML左边第一个则是MDA的商标。UML是OMG相当重要且成功的一个标准，所以我们可以从OMG网站（http://www.omg.org/）找到UML官方网站的入口。

图1-20 OMG的网站

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

当然，我们也可以不通过OMG网站，直接连到UML的官方网站（http://www.uml.org/）。在UML官方网站中，列出了许多UML资源，像是UML的标准规格文件、最新消息、相关文章、开发工具或论坛网站等。

1.6.2　中立的建模语言

UML是统一建模语言（Unified Modeling Language）的缩写，顾名思义，它是建立模型（Model）的专用语言，并不指定使用于任何特殊的应用领域、具体平台、实现语言或实现方法，因此UML天生就特别适合用来表达MDA的CIM和PIM模型。

UML包含一套有明确定义的图示，方便设计师用来绘制设计蓝图，建造软件模型。UML 2.0最新版定义了13款图，每种图适合表达一种设计观点。请看图1-21的例子，这是一张UML用例图，非常适合用来表达PIM阶段的系统功能。

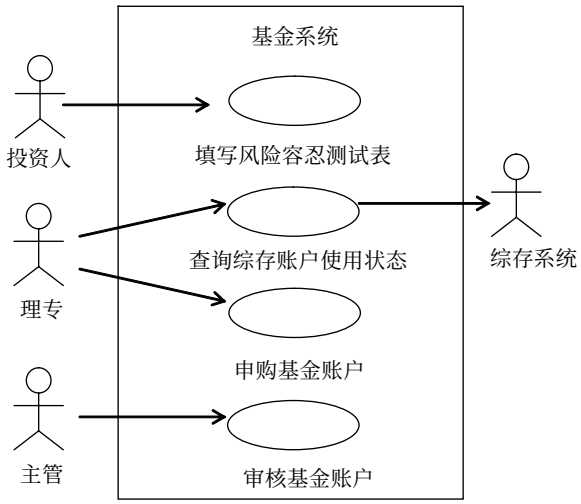

图1-21　UML的用例图

再看另一款序列图（Sequence Diagram），则适合用来表达软件内部对象交互的情况。请看图1-22的例子，我们用序列图表达了申购基金流程。

有了UML之后，会方便设计师表达对软件的规划，但是UML标准文件里头，并不会说明该如何设计软件。简言之，虽然UML定义了标准的建模语言，有助于设计师使用相同的图标元素表达软件设计，但是UML没有规范，也没说明该如何设计软件，因此我们还得熟知分析设计技术，才能够建构出高质量的UML模型。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

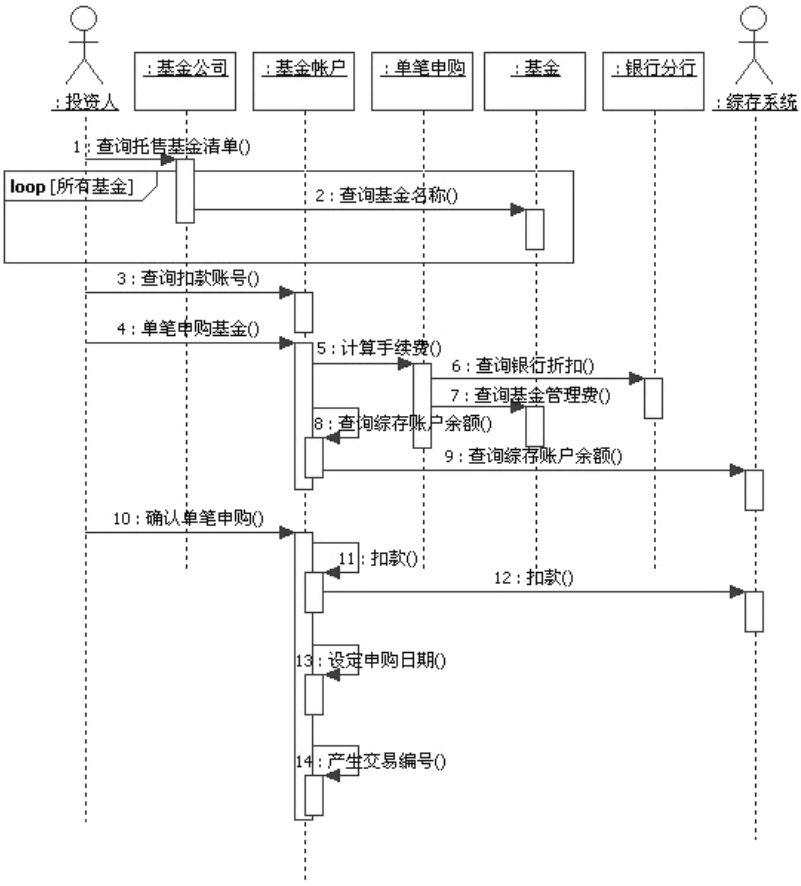

图1-22 UML的序列图

1.6.3 Profile支持定制化UML方言

虽然UML天性中立，可是通过UML所提供的Profile机制，却可以让我们自创独特的UML方言，以便应用于特定应用领域、具体平台、实现语言或实现方法上，也因此可以表达MDA的PSM设计模型。请看图1-23，这是一个简单的EJB2的UML Profile设计，节录自UML规格文件里的设计图。

其实，UML Profile的概念不难，使用也容易。回头来解释EJB2的例子，简单来说，使用EJB2具体平台时，一定会接触到两项重要的概念，其一是Bean的概念，另一是Interface的概念。

在EJB2里，最常用的Bean有Entity Bean与Session Bean，前者主要提供数据存取的服务，后者则提供流程控制的服务。由于UML中立于任何一项特定的具体平台，所以也就没有定义Bean这个跟EJB2具体平台有关的特殊元素。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

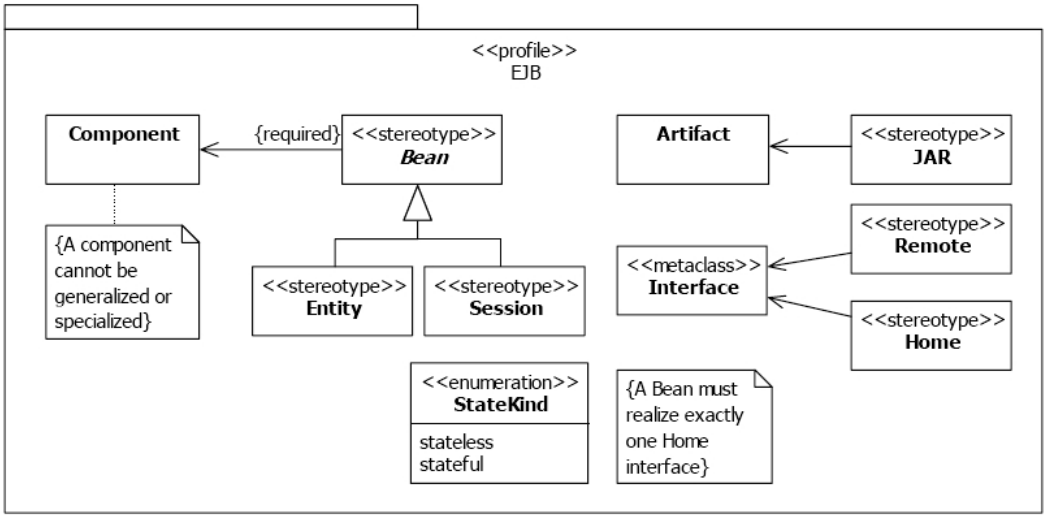

图1-23　UML Profile for EJB2

但是，Bean的概念与UML里的Component概念十分相近，所以通过UML的扩展机制，便能够以UML现有的Component元素为基底，扩展出适用于EJB2具体平台的Entity Bean及Session Bean，如图1-24所示。同理，我们以UML里的Interface为基底，扩展出EJB2具体平台里的Remote Interface和Home Interface，如图1-25所示。

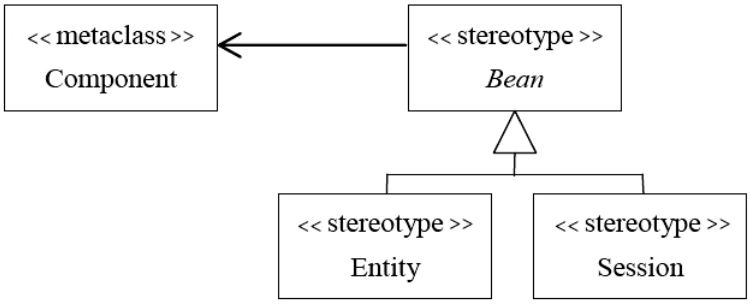

图1-24　Bean扩展自Component

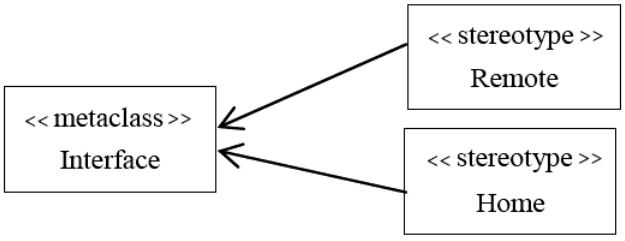

图1-25　Remote和Home扩展自Interface