



要求  
规格

# 第一部分：软件需求规格说明 (SRS) (第 10 和 11 章)



# SRS 作为需求规格说明书

- SRS描述了分析阶段的预期结果
  - 功能——软件在给定输入的情况下能够提供的功能（?）
  - 功能——软件能做什么和不能做什么
  - 功能特性——该软件与其他软件有何不同（独特的行为？还有哪些其他行为？）
  - 约束条件——软件必须遵循哪些规则/条例（业务规则和IT行业标准）
  - 系统行为在各种条件下（除特征之外的行为）
    - 特质——包含内在特质和外在特质
- SRS本身也是规范阶段的输出。
- 它是后续项目规划、设计和编码的基础，也是系统测试和用户文档的基础。
- 制定软件需求规格说明书（SRS）是明确需求的过程。



这张图展示了由四项功能支持的解决问题的能力。

质疑、探索、创造和实施。

# SRS 受众

- 客户（用户）——他们需要了解什么产品预计它们将由SRS交付。
- 项目经理——他们计划/监控/修改根据SRS中给出的要求，制定时间表、工作量和资源计划。
- 软件开发团队——他们需要知道要建造什么如SRS中所述。
- 测试人员——他们使用SRS来发展基于需求的测试、测试计划和测试程序。
- 维护和支持人员——他们使用SRS理解产品的每个部件应该做什么，以及它们与其他部件/系统之间的关系。
- 文档编写人员——他们编写用户手册以及 SRS 中规定的帮助屏幕，特别是用户界面提供的功能。
- 培训人员——他们使用SRS和用户文档制定培训计划教程。
- 法律人员——他们需要确保各项要求得到满足。遵守适用的法律法规。
- 分包商——他们已签订合同根据规定的要求

# SRS结构和内容

## 1. 引言部分

概述，帮助读者了解 SRS 的组织结构以及如何使用它。

- 目的

- 项目的目的，即在项目中要开发的产品或应用程序，以及与这些需求相关的要求（“开发……的产品”）。
- 文档的目标受众（请按上一张幻灯片中所述进行指定）
- 文档规范——文档中使用的格式标准/规范（例如 IOS/IEC/IEEE）
- 范围——项目将实现的目标以及需要完成的工作。**不是它的关注**
- 参考文献——本文档引用的资源

## 1. Introduction

- 1.1 Purpose
- 1.2 Document conventions
- 1.3 Project scope
- 1.4 References

## 2. Overall description

- 2.1 Product perspective
- 2.2 User classes and characteristics
- 2.3 Operating environment
- 2.4 Design and implementation constraints
- 2.5 Assumptions and dependencies

## 3. System features

- 3.x System feature X
  - 3.x.1 Description
  - 3.x.2 Functional requirements

## 4. Data requirements

- 4.1 Logical data model
- 4.2 Data dictionary
- 4.3 Reports
- 4.4 Data acquisition, integrity, retention, and disposal

## 5. External interface requirements

- 5.1 User interfaces
- 5.2 Software interfaces
- 5.3 Hardware interfaces
- 5.4 Communications interfaces

## 6. Quality attributes

- 6.1 Usability
- 6.2 Performance
- 6.3 Security
- 6.4 Safety
- 6.x [others]

## 7. Internationalization and localization requirements

## 8. Other requirements

## Appendix A: Glossary

## Appendix B: Analysis models

# SRS结构和内容

## 2. 总体描述部分

对以下内容进行高层次概述（概括性概述，无需细节）：产品以及环境产品将用于何处，预期用户并且已知限制条件，假设，和依赖项

- 产品视角——产品的背景和来源（为什么需要它）
- 用户类别和特征- 目标用户及其特征（谁会使用它以及他们会以何种方式使用）
- 运行环境——包括硬件平台；操作系统及其版本；用户、服务器和数据库的地理位置；以及托管相关数据库、服务器和网站的组织（现在很难拥有独立的系统）
- 设计和实施限制——任何会限制开发人员可用选项的因素以及每项限制的理由，例如要使用的编程语言和库的限制等。（技术限制（而不是业务规则））
- 假设- 功能相关（例如，所有手机都有触摸屏）
- 依赖项- 不可控的外部因素，例如特定版本的 .NET

# SRS结构和内容

## 3. 系统功能部分

为每个功能设置子标题（客户要求）并在相应的子标题中逐一描述这些特征。

- 3.x 系统功能 x (子标题)

### 3.x.1. 说明

- 特征 x 是什么？
- 功能 X 开发中的优先级

### 3.x.2. 功能需求 (顾客提出的)

- 与功能 x 相关的详细功能需求，即软件为实现此功能应具备的功能和能力。

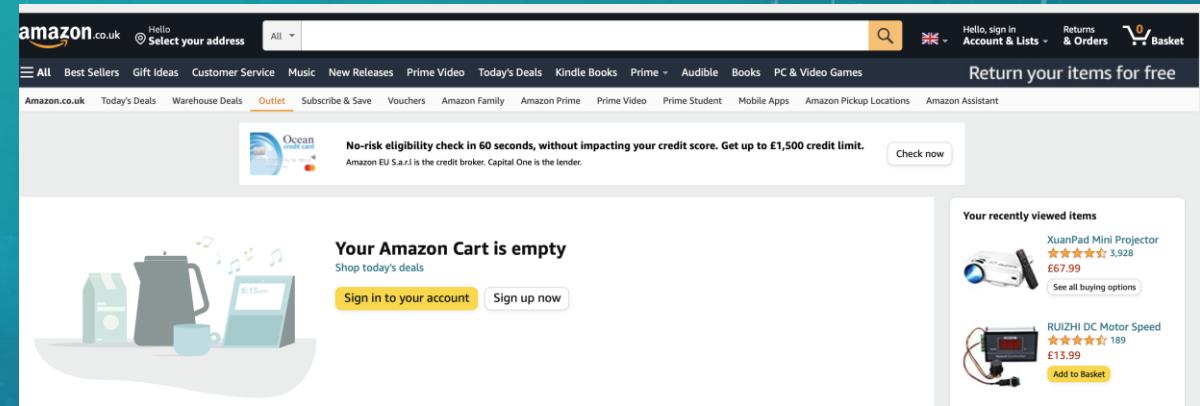
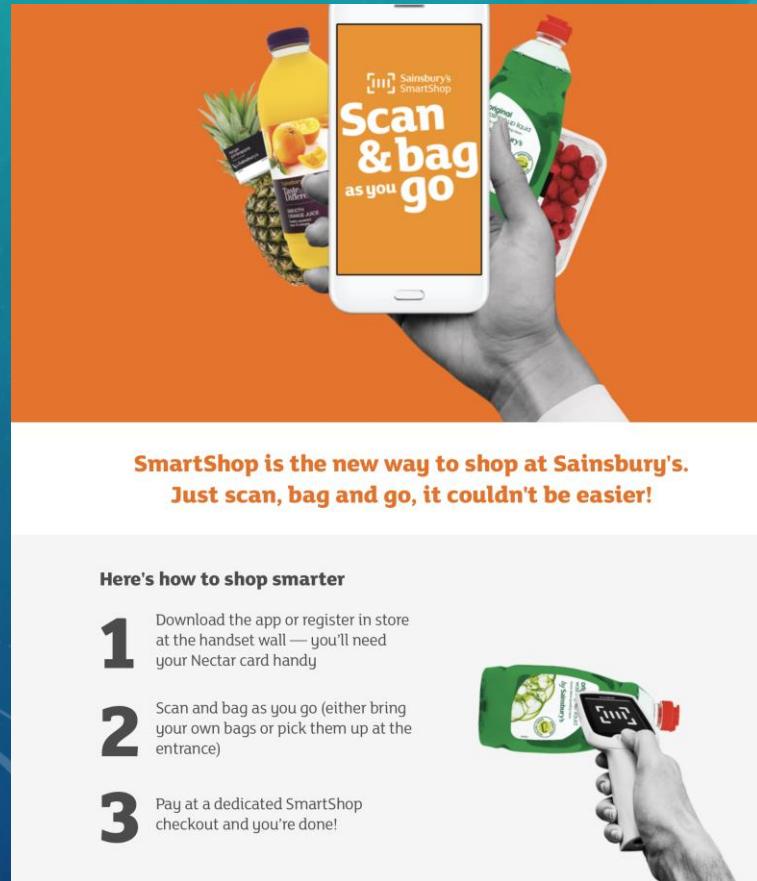
(以上步骤应重复用于所有功能。)

请注意，我们这里讨论的是需求，而不是系统规格！因此，这些特性和功能都是客户要求的。

# SRS结构和内容

练习1：购物车？

购物车有哪些功能？支持这些功能的功能有哪些？



练习2：使用智能扫描仪进行智能购物 想象一下并说出其中两个功能？  
设想并命名支持这些功能的函数？

# SRS结构和内容

## 4. 数据要求部分（更多详情请参见第二部分）

定义数据类型该软件将用作输入、中间变量和输出。

- 逻辑数据模型，例如实体关系图（每列的数据类型）和 UML 类图（日期类型）
- 数据字典——定义数据结构的组成，以及数据元素的含义、数据类型、长度、格式和允许的值。
- 报告——由IT系统生成的管理报告，包括其格式、大小、语言、表格、图表等。
- 数据采集（所有与数据保护相关的内容），包括同意（目的、生命周期、使用对象）、完整性（传输时保持质量）、保留（保存在安全的地方）和处置流程（销毁）

# SRS结构和内容

## 5. 外部接口要求部分

6. 非功能性需求部分（更多内容见第三部分） 明确系统如何与用户以及外部软件和硬件进行通信

- 性能——例如响应速度、吞吐量、执行时间和存储容量
- 安全——信息安全现在是一个大问题。
- 无障碍设施——为残障人士提供服务

## 7. 国际化和本地化部分

国际化和本地化要求确保产品适用于其创建地以外的国家、文化和地理位置。（试想一下，如果京东只有中文版本，那就无法实现国际化。）

## 8. 其他要求部分

- 法律、监管或财务合规和标准要求
- 产品安装、配置、启动和关闭；以及日志记录、监控和审计跟踪要求。

# SRS验证

收集到的各项需求需要根据以下标准进行验证：

- 完整版——包含所有内容**详细的信息**（例如底色——款式、尺寸、颜色、效果）
- 正确——没有说错话。
- 可行性——在系统及其运行环境的能力和限制，以及时间、预算、人力资源等条件下可实现。
- 必要性——能够带来预期的商业价值，使产品在市场上脱颖而出，或者符合外部标准、政策或法规的要求。
- 优先级排序——根据对实现预期价值的重要性对业务需求进行排序。
- 明确无误——不会误导或造成误解（“他在煮苹果”）
- 可验证性——最后可以检查要求是否已实现。

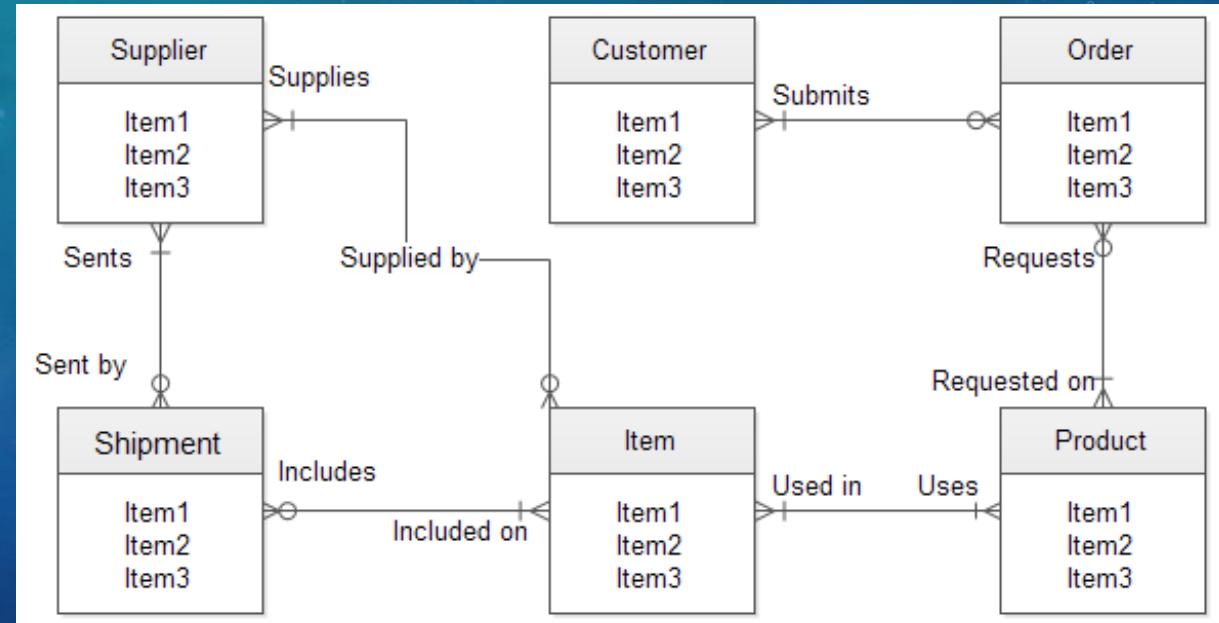
## 第二部分：数据（第13章）

# 数据建模

数据规范是指软件系统中将要操作哪些数据。它至关重要，因为软件功能的核心在于数据——数据的创建、读取、更新和删除。

## ER模型

- 旨在明确数据实体本身及其相互关系的分析过程
- 实体关系图作为一种通用数据模型（ER模型）
- 实体（数据项）——数据聚合或物理项目——属性——数据项的特征
- 关系/关联——数据项对之间的逻辑联系
- 多重性——关联关系一端可以存在的实体类型实例的数量。



# 数据字典

- 日期字典是一个元数据存储库——包含有关数据的信息，例如含义、与其他数据的关系、来源、用途和格式，或者是关于元数据的表格集合。
- 例子：

**DATA**

employee_id	first_name	last_name	nin	dept_id
44	Simon	Martinez	HH 45 09 73 D	1
45	Thomas	Goldstein	SA 75 35 42 B	2
46	Eugene	Cornelsen	NE 22 63 82	2
47	Andrew	Petculescu	XY 29 87 61 A	1
48	Ruth	Stadick	MA 12 89 36 A	15
49	Bamy	Scardelis	AT 20 73 18	2
50	Sidney	Hunter	HW 12 94 21 C	6
51	Jeffrey	Evans	LX 13 26 39 B	6
52	Doris	Berndt	YA 49 88 11 A	3
53	Diane	Eaton	BE 08 74 68 A	1

**DATA DICTIONARY (METADATA)**

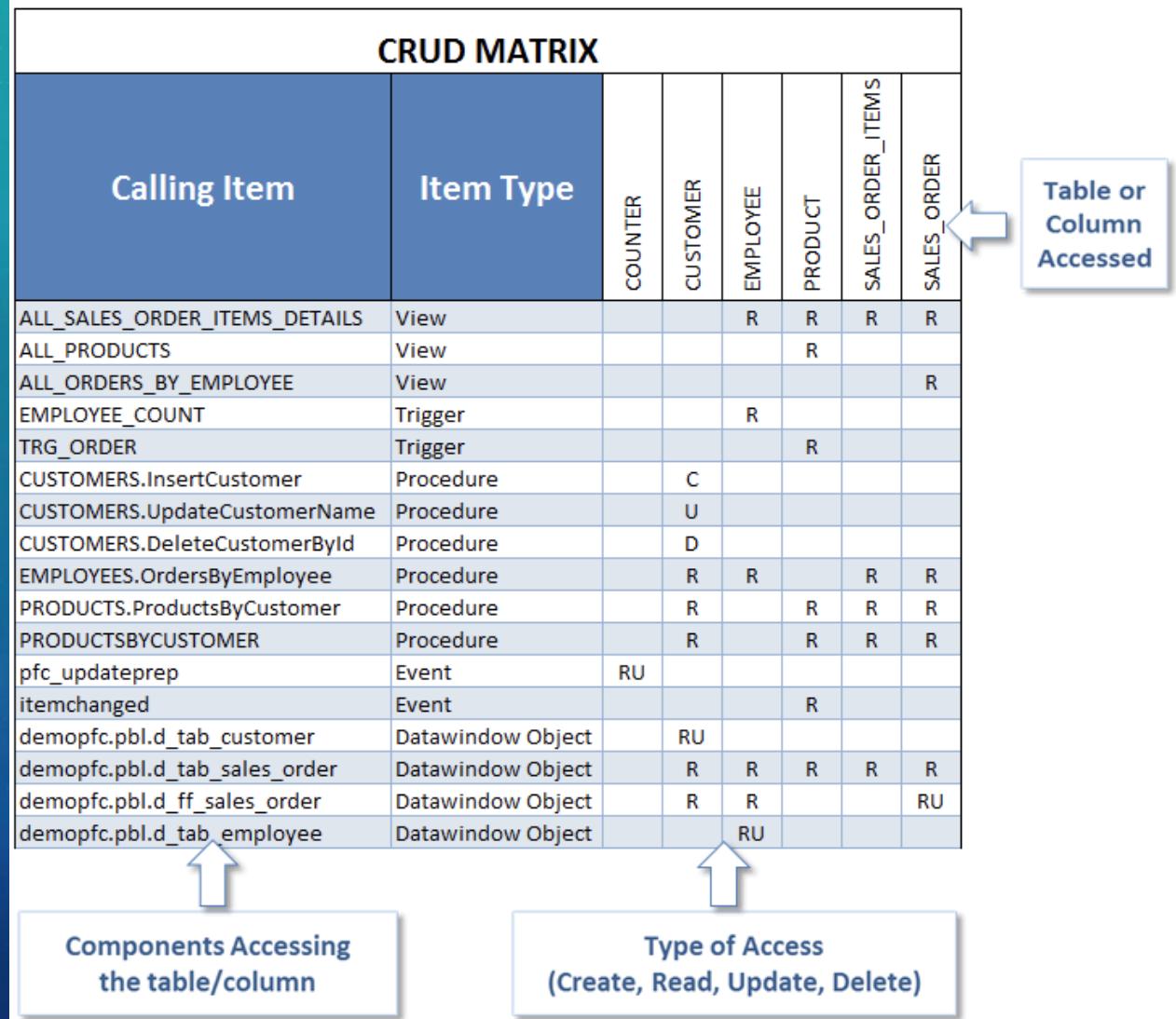
Column	Data Type	Description
employee_id	int	Primary key of a table
first_name	nvarchar(50)	Employee first name
last_name	nvarchar(50)	Employee last name
nin	nvarchar(15)	National Identification Number
position	nvarchar(50)	Current position title, e.g. Secretary
dept_id	int	Employee department. Ref: Departments
gender	char(1)	M = Male, F = Female, Null = unknown
employment_start_date	date	Start date of employment in organization.
employment_end_date	date	Employment end date.

 Dataedo

# CRUD 矩阵

- CRUD 代表创建、读取、更新和删除。
- CRUD 矩阵将系统操作与数据实体关联起来，以显示每个重要数据实体的创建、读取、更新和删除的位置和方式。
- 相关性的类型，包括以下几种：
  - 数据实体和系统事件
  - 数据实体和用户任务或用例
  - 对象类和用例
- CRUD 不仅仅包含矩阵，但对于需求规范而言，矩阵是重点。
- 示例（数据实体和用户任务）：

CRUD MATRIX							
Calling Item	Item Type	COUNTER	CUSTOMER	EMPLOYEE	PRODUCT	SALES_ORDER_ITEMS	SALES_ORDER
ALL_SALES_ORDER_ITEMS_DETAILS	View			R	R	R	R
ALL_PRODUCTS	View				R		
ALL_ORDERS_BY_EMPLOYEE	View						R
EMPLOYEE_COUNT	Trigger			R			
TRG_ORDER	Trigger				R		
CUSTOMERS.InsertCustomer	Procedure		C				
CUSTOMERS.UpdateCustomerName	Procedure		U				
CUSTOMERS.DeleteCustomerById	Procedure		D				
EMPLOYEES.OrdersByEmployee	Procedure		R	R		R	R
PRODUCTS.ProductsByCustomer	Procedure		R		R	R	R
PRODUCTSBYCUSTOMER	Procedure		R		R	R	R
pfc_updateprep	Event	RU					
itemchanged	Event				R		
demopfc.pbl.d_tab_customer	Datawindow Object			RU			
demopfc.pbl.d_tab_sales_order	Datawindow Object		R	R	R	R	R
demopfc.pbl.d_ff_sales_order	Datawindow Object		R	R			RU
demopfc.pbl.d_tab_employee	Datawindow Object			RU			

A diagram illustrating the components of the CRU matrix. Two blue arrows point upwards from two white boxes to the 'Components Accessing the table/column' and 'Type of Access (Create, Read, Update, Delete)' columns in the matrix table.

Components Accessing the table/column

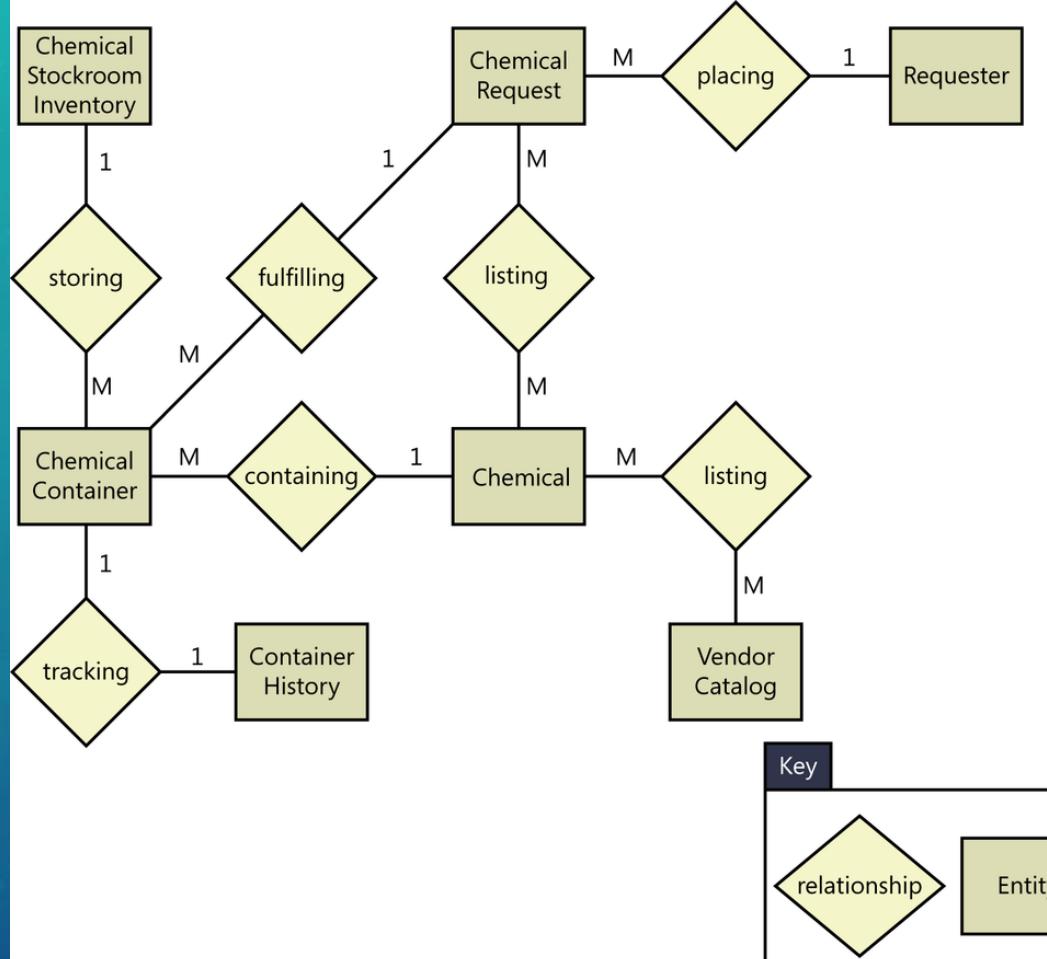
Type of Access (Create, Read, Update, Delete)

Table or Column Accessed

# 数据规范

## 案例研究——化学追踪系统

- “ER” 模型
- CRUD矩阵，例如：第一行表示：下单用例：创建订单，读取化学品、请求者和供应商目录的相关数据。
- CRUD 还规定了每个用例应该具备的方法，例如，第三行表示：化学品库存包含数据库中 C、U 和 D 类化数据的方法。如果数据无需手动操作，则可以定义一个名为“库存”的类。



Entity Use Case	Order	Chemical	Requester	Vendor Catalog
Place Order	C	R	R	R
Change Order	U, D		R	R
Manage Chemical Inventory		C, U, D		
Report on Orders	R	R	R	
Edit Requesters			C, U	

# 报告

## IT系统生成报告

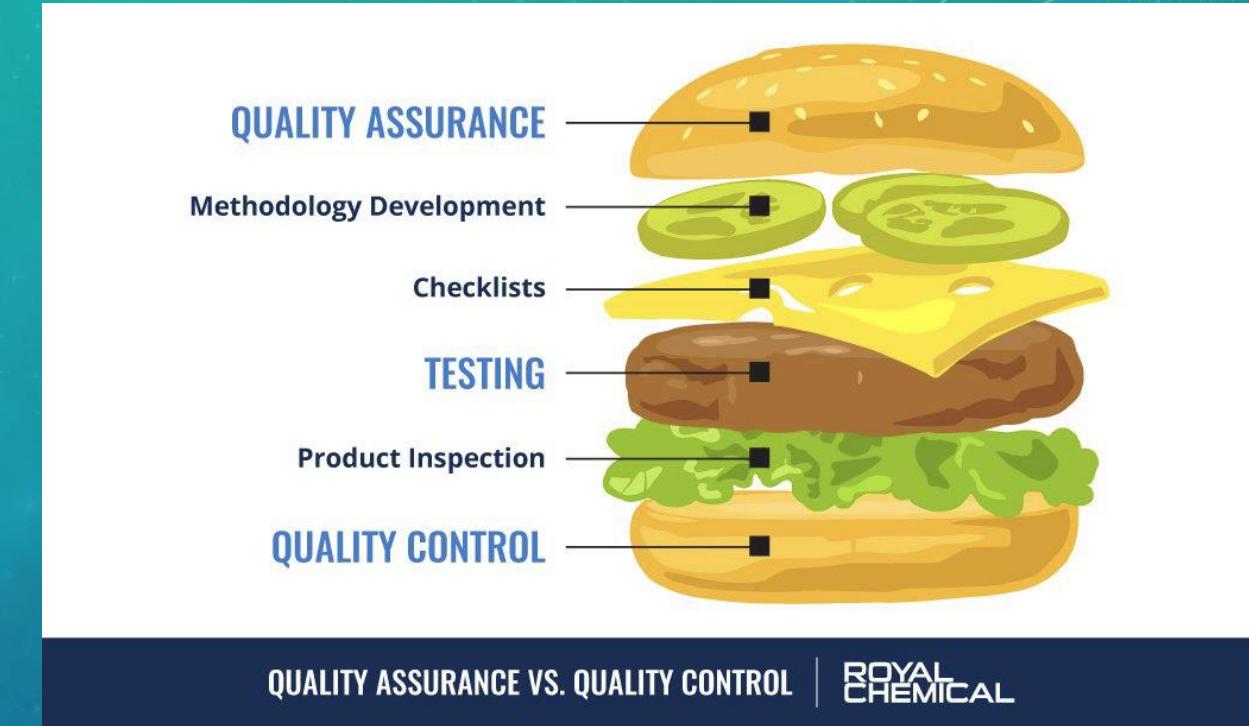
- 数据来源是什么？从数据存储库中提取数据的选择标准是什么？
- 用户可以选择哪些参数？
- 需要进行哪些计算或其他数据转换？
- 排序、分页和总计的标准是什么？
- 如果在尝试生成此报告时，查询未返回任何数据，系统应如何响应？
  
- 是否应向用户提供报告的基础数据，以便用户进行临时报告？
- 这份报告能否作为模板，用于撰写一系列类似的报告？

# 第三部分：质量（第14章）

# 质量属性

## 讨论非功能性需求的原因

- 除了功能之外，用户对产品的实际表现也抱有一些期望，尽管这些期望往往是未曾言明的。
- 优秀的产品体现了各种相互竞争的质量特性之间的最佳平衡——因此，我们需要识别并优先考虑这些质量属性。（想想.NET IDE。它让我们开发图形用户界面非常方便，但IDE本身也变得过于复杂。虽然Java允许你在任何文本编辑器上进行编程，但与.NET相比，实现图形用户界面并不容易——这是我10年前的经验。）



## 分类

- 外部质量因素是用户可以看到的，因此对用户而言最为重要。
- 内部质量因素是指开发人员可以看到的东西，因此对开发和维护人员来说更为重要。
- ( (更多内容请见下一页)

# 质量属性

两类：

• 用户  
兴趣

• 开发者  
兴趣

外部质量	简述
可用性 安装性 正直 互操作性	系统服务在需要时和地点的可用性；应用程序的正确安装、卸载和重新安装的难易程度。 系统防止数据不准确和丢失的程度；系统与其他系统或组件互连和交换数据的便捷程度
表现 可靠性 鲁棒性 安全 安全 可用性	系统对用户输入或其他事件的响应速度和可预测性如何；系统在发生故障前能运行多久；系统故障发生前的运行时间。 系统对意外运行状况的响应能力如何？系统防止人身伤害或财产损失的能力如何？ 该系统对防止未经授权访问应用程序及其数据的能力如何？人们学习、记忆和使用该系统是否容易？
内部质量	简述
效率 可修改性 可移植性 可重复使用性 可扩展性 可验证性	系统利用计算机资源的效率如何 系统的维护、变更、增强和重构的难易程度；系统在其他操作系统环境下运行的难易程度；组件在其他系统中的可用程度。 系统扩展处理更多用户、交易、服务器等的便捷程度如何？开发人员和测试人员能否轻松确认软件已正确实现？

# 外部质量属性

**可用性**= $MTBF/(MTBF+MTTR)$ ，其中 MTBF 为平均故障间隔时间，MTTR 为平均修复时间。可用性与可靠性密切相关，并受可维护性子类别（可修改性）的显著影响。可用性可以表示为：

- 系统正常运行时间（XX 至 YY），系统停机时间（AA 至 BB）。
- 维护前的警告或警报

提高系统的**安装性**减少安装操作所需的时间、成本、用户干扰、错误频率和技能水平。衡量系统可安装性的一个指标是系统平均安装时间。例如：

- 未经培训的用户平均只需 10 分钟即可成功完成应用程序的首次安装。
- 安装应用程序的升级版本时，用户配置文件中的所有自定义设置都将被保留，并在需要时转换为新版本的数据格式。
- 安装程序会在开始安装过程之前验证下载文件的正确性。
- 在服务器上安装此软件需要管理员权限。
- 安装成功后，安装程序将删除与该应用程序关联的所有临时文件、备份文件、过时文件和不需要的文件。

# 外部质量属性

数据**正直**还涉及数据的准确性、完整性和正确格式。

- 需求示例：
  - 文件备份完成后，系统应将备份副本与原始文件进行核对，并报告任何差异。
  - 该系统应防止未经授权的数据添加、删除或修改。
  - 化学追踪系统应确认导入第三方结构绘制工具的编码化学结构代表有效的化学结构。
  - 系统应每日确认应用程序可执行文件未被添加未经授权的代码而修改。

**互操作性**表明系统与其他系统协同工作的便捷程度，例如与其他软件系统交换数据和服务以及与外部硬件设备集成。

- 例子：
  - 基于网络的医疗保健信息系统应该能够从物联网导入传感数据，从互联网搜索信息，并使用来自云端的预测模型。

# 外部质量属性

## 表现

性能维度	例子
响应时间	显示网页所需的秒数
吞吐量	每秒处理的信用卡交易量
数据容量	数据库最多可以保存 9GB 的信息（真的吗？有用户问过数据库的大小吗？）。（谷歌邮箱声称可以保存无限数量的邮件。）
动态容量	社交媒体网站的最大并发用户数
实时系统中的可预测性	预测汽车行驶的道路状况
延迟（我认为这是捏造的。延迟与网络有关。用户只能看到响应时间。）	音乐录制和制作软件中的时间延迟
降级模式或过载状态下的行为	自然灾害导致紧急电话系统呼叫量激增。（我的MacBook Pro开启录制功能后，无法流畅运行MS Teams会议。）

# 外部质量属性

软件规范和测量方法**拉可靠性**包括操作正确完成的百分比、系统平均故障间隔时间 (MTBF) 以及给定时间段内可接受的最大故障概率。例如：

- 每1000次运行中出现1次失败
- 读卡器组件的平均故障间隔时间应至少为 90 天 (**卡尔在90天内使用读卡器多少小时？**)
- 故障率为每1000小时3次

**鲁棒性**是指系统在面对无效输入、连接的软件或硬件组件缺陷、外部攻击或意外运行情况时，仍能正常运行的程度。文本编辑器就是一个很好的例子，例如……

- 如果文本编辑器在用户保存文件之前发生故障，则当同一用户下次启动该应用程序时，它应恢复故障发生前至多一分钟内正在编辑的文件内容。 (**微软Office**)

**安全**-如果软件系统控制硬件，则可能造成人身伤害和财产损失。

# 外部质量属性

**安全**这是互联网软件和云计算领域的一个主要问题。在收集安全需求时需要考虑的一些因素。

- 用户授权或权限级别（普通用户、访客用户、管理员）和用户访问控制（角色和权限矩阵可能是一个有用的工具）
- 用户身份识别和验证（密码构造规则、密码更改频率、安全问题、忘记登录名或密码的处理程序、生物识别、访问失败后锁定帐户、无法识别的计算机）
- 数据隐私（谁可以创建、查看、更改、复制、打印和删除哪些信息）
- 蓄意销毁、篡改或窃取数据
- 抵御病毒、蠕虫、特洛伊木马、间谍软件、rootkit 和其他恶意软件
- 防火墙和其他网络安全问题
- 安全数据的加密
- 构建操作和访问尝试的审计跟踪记录

# 外部质量属性

可能的改进方法**可用性**（用户友好性、易用性和人机工程学）

学习的难易程度	易用性
详细提示	键盘快捷键
巫师	丰富且可自定义的菜单和工具栏
可见菜单选项	访问同一功能的多种方法
有意义、通俗易懂的信息	条目自动完成
帮助屏幕和工具提示	错误自动纠正
与其他熟悉的系统相似	宏录制和脚本功能
显示的选项和小部件数量有限。	能够将先前交易的信息延续到下一个交易中
	自动填写表单字段
	命令行界面

下图是否让你想起了什么？人们曾拿它开玩笑。  
关于 MS Windows 多年前如何启动该系统。  
现在，就连电动汽车也采用了相同的技术。  
象征。



# 内部质量属性 (???)

**效率**效率与性能这一外部质量属性密切相关。效率衡量的是系统对处理器容量、磁盘空间、内存或通信带宽的利用程度。效率（以及由此决定的性能）是系统架构的关键驱动因素，它影响着设计者如何选择在系统组件之间分配计算和功能。

- 例如：
  - 在计划的峰值负载条件下，应用程序可用的处理器容量和内存至少应有 30% 未被使用。
  - 当使用负荷超过计划最大容量的 80% 时，系统应向操作员发出警告信息。

测量方法**可修改性**包括添加新功能或修复问题所需的平均时间，以及修复成功率。它与维护密切相关。

- 例如：
  - 经过认证的维修技术人员应该能够在 10 分钟内更换扫描仪模块。
  - 如果更换的墨盒没有插入正确的插槽，打印机将显示错误信息。

# 内部质量属性

**可移植性**随着应用程序需要在多种环境下运行，例如 Windows、Mac 和 Linux；iOS 和 Android；以及 PC、平板电脑和手机，可移植性变得越来越重要。一些从业者将产品的国际化和本地化能力归入可移植性的范畴。

- 例如：
- 将 iOS 版本的应用程序修改为可在 Android 设备上运行，所需更改的源代码不得超过 10%。
- 用户应能够将浏览器书签在 Firefox、Internet Explorer、Opera、Chrome 和 Safari 之间相互导入导出。

**可重复使用的**软件必须是模块化的、文档齐全的、独立于特定应用程序和操作系统环境的，并且具有一定的通用功能。

**可扩展性**解决应用程序在不影响性能或正确性的前提下，如何扩展以容纳更多用户、数据、服务器、地理位置、交易、网络流量、搜索和其他服务的问题。可扩展性既涉及硬件也涉及软件。

为……设计软件**可验证性**这意味着可以轻松地将软件置于所需的预测试状态，提供必要的测试数据，并观察测试结果。

# 识别与排名

## 排名的必要性

- 并非所有软件系统都需要满足所有内部和外部质量属性要求。
- 仅考虑内部空间。它是一个六维空间。某些维度之间可能存在冲突。很难保持平衡。
- 排名示例：
  - 嵌入式系统：性能、效率、可靠性、鲁棒性、安全性、可靠性、易用性
  - 互联网和企业应用：可用性、完整性、互操作性、性能、可扩展性、安全性、易用性
  - 桌面和移动系统：性能、安全性、易用性

# 识别与排名

## 过程

- 步骤 1：从广泛的分类开始——从一系列丰富的质量属性开始考虑，以降低忽略重要质量维度的可能性。
- 步骤 2：缩减清单——让各利益相关方参与进来，通过评估哪些属性可能对项目很重要来制定质量目标。
- 步骤 3：确定属性优先级——确定相关属性的优先级，并为未来的需求收集讨论设定重点。

Attribute	Score	Availability	Integrity	Performance	Reliability	Robustness	Security	Usability	Verifiability
Availability	2	^	^	^	<	^	^	<	
Integrity	6		<	<	<	^	<	<	
Performance	4			<	<	^	^	<	
Reliability	2				<	^	^	^	<
Robustness	1					^	^	<	
Security	7						<	<	
Usability	5							<	
Verifiability	1								

In、Pe、Re、Se、Us、Av。Ro、Ve  
Se、In、Pe、Re、Us、Av、Ro、Ve  
Se、In、Pe、Us、Re、Av、Ro、Ve  
.....

# 识别与排名

- 第四步：明确具体期望（赋予属性值）——需要问的问题包括：
  - 对于典型的专利申请，在收到查询后，合理的或可接受的检索响应时间是多久？
  - 用户认为典型查询的响应时间多久是不可接受的？
  - 预计平均有多少用户同时在线？
  - 您预计最多可以有多少个同时在线用户？
  - 一天中、一周中、一个月中或一年中的哪些时段的使用量远高于平时？
- 第五步：明确结构良好的质量要求——运用SMART原则（具体、可衡量、可实现、相关、有时限）。

# 质量属性权衡

这是为了平衡质量属性要求。

- 例子：
  - + 表示增加相应行中的属性通常会对相应列中的属性产生积极影响。
  - - 这意味着增加该行中的属性通常会对列中的属性产生不利影响。

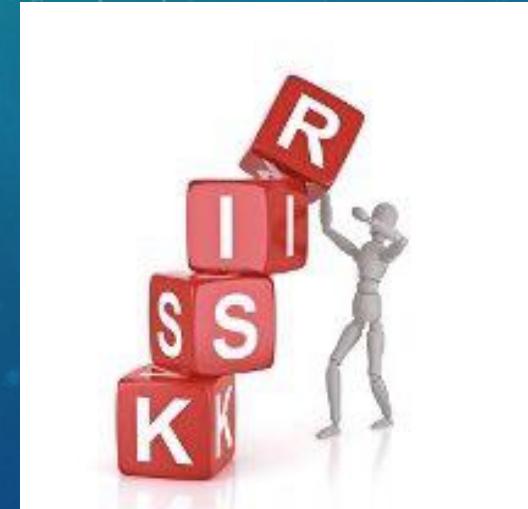
	Availability	Efficiency	Installability	Integrity	Interoperability	Modifiability	Performance	Portability	Reliability	Reusability	Robustness	Safety	Scalability	Security	Usability	Verifiability
Availability																
Efficiency	+					-	-	+	-			-	+		-	
Installability	+		+						+				+			
Integrity		-		+						-		+		+	-	-
Interoperability	+	-	-	-	+		-	+	+		+	-		-		
Modifiability	+	-			+		-	+	+			+				+
Performance		+			-	-	+		-			-	-		-	-
Portability		-			+	-	-		+				-	-	+	
Reliability	+	-	+		+	-			+			+	+		+	+
Reusability		-		-	+	+	-	+		+			-		+	
Robustness	+	-	+	+	+		-		+		+	+	+	+	+	
Safety		-		+	+		-			+		+		+	-	-
Scalability	+	+		+			+	+	+		+			+		
Security	+			+	+		-	-	+		+	+		+	-	-
Usability		-	+				-	-	+		+	+			+	-
Verifiability	+		+	+		+			+	+	+	+		+	+	

# 第四部分：风险移民（第15章）

# 风险与原型设计

## 风险？什么风险？

- 由于不确定性，需求可能不明确。
  - 模糊性
  - 歧义
  - 不完整
  - 不准确
  - 变化
- 这些不确定性可能会使预期的软件面临风险，从而导致系统出现问题。不满足用户真实需求
- 除了用户需求之外，还有很多其他问题，例如 GDPR、知识产权、道德问题（例如“功能失调”的软件的后果）等等。
- 原型可以识别并降低风险。它们展示了预期软件的运行方式、无法正常运行的部分以及由此产生的后果。



# 风险与原型设计

## 原型类型或用途

- 范围：模型（面向用户）和概念验证（面向开发者）
- 弗罗姆，无论你将来是否会使用它：一次性的和进化的
- 形式：纸质版和电子版

## 小样：专注于界面的一部分。

- 为了探索目标系统的某些特定行为，并最终完善需求，我们制作了模型。该模型有助于用户判断基于原型设计的系统是否能够让他们以合理的方式完成工作。
- 模型可以展示用户可用的功能选项、用户界面的外观和感觉（颜色、布局、图形、控件）以及导航结构。
- 模型虽然看起来像是“界面”的一部分，但它实际上并没有执行任何有用的工作，也就是说，它只显示外观，而不显示交互。

# 风险与原型设计

一个**概念验证**从用户界面到所有技术服务层，实现应用程序功能的一部分。

- 概念验证原型机的工作原理与真实系统相同。
- 当您不确定拟议的架构方案是否可行且合理，或者想要优化算法、评估拟议的数据库模式、确认云解决方案的合理性或测试关键时序时，请开发概念验证。

**丢弃**原型仅用于探索。

- 为了回答问题、消除不确定性并提高需求质量
- 由于这些代码最终会被丢弃，系统质量（内部和外部）都会被忽略。因此，务必确保不要让来自一次性原型系统的低质量代码迁移到生产系统中。

**进化**原型为产品逐步构建提供了坚实的架构基础，随着需求的逐渐明确，产品可以逐步完善。就软件系统质量而言，它具有很高的品质。

# 风险与原型设计

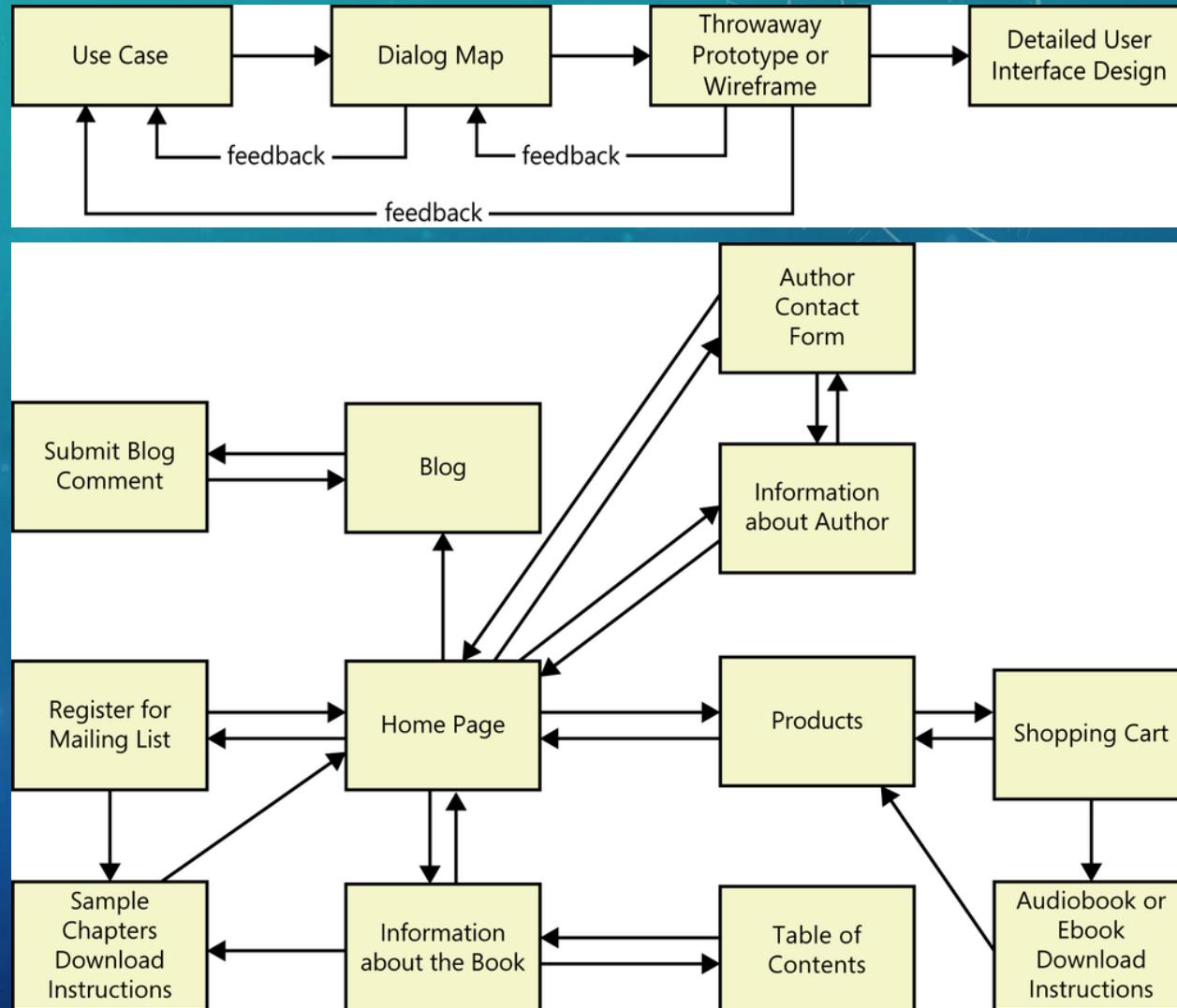
	丢弃	进化
小样	<p>明确并完善用户和功能需求。</p> <p>找出缺失的功能。</p> <p>探索用户界面设计方案。</p>	<p>实现核心用户需求。</p> <p>根据优先级实现其他用户需求。</p> <p>实施并完善网站。</p> <p>调整系统以适应快速变化的业务需求。</p>
证明概念	<p>展示技术可行性。</p> <p>评估绩效。</p> <p>学习相关知识以改进施工估算。</p>	<p>实现并扩展核心多层功能和通信层。</p> <p>实现并优化核心算法。</p> <p>测试并优化性能。</p>

# 风险与原型设计

## 例子：

将《沙中珍珠》一书数字化，以便在线阅读。此外，还将通过博客、电子邮件等方式发布。

用户类 (演员)	用例
游客	获取本书信息 获取作者信息 阅读样章  阅读博客 联系作者
顾客	订购产品 下载电子产品 请求问题帮助
管理员 托尔	管理产品列表；向客户退款；管理电子邮件列表



# 风险与原型设计

## Pearls from Sand

[background image of pearl  
on a sandy beach]

Home About the Book About the Author Blog Submit a Pearl Buy the Book Contact

### Products



#### Signed Paperback

description blah blah blah

Add to Cart



#### Paperback from Amazon, Kindle

description blah blah blah

Link

Link



#### PDF ebook, audiobooks, etc.

description blah blah blah

Add to Cart

View Cart

Sample Pearl of Wisdom

## Pearls from Sand

Home About the Book About the Author Blog Submit a Pearl Buy the Book Other Books Contact

### Products



Signed Paperback – \$14.00

This paperback copy of *Pearls from Sand* has been signed by the author. \$2.00 will be added for shipping by US Mail.  
See Sample Chapters

Add to Cart



Paperback, Kindle, Nook

Order paperback or eBook copies of *Pearls from Sand* from these on-line retailers:



See Sample Chapters



eBook in PDF Format – \$5.00

This PDF file contains the complete text of the paperback book. You'll receive an e-mail with downloading instructions after placing your order.  
See Sample Chapters

Add to Cart