編號:H15H15-96AE003-0200803200-RA

RA（BA/SA）角色工作 -- 需求分析（ReqM, RD）

R00 陳建維 (Chris)

cwchen@hyweb.com.tw

目標(Purpose)

- 體認需求管理的重要性
- 瞭解各階段RA(BA/SA)角色的任務與做法
- 瞭解需求發展的流程與產出
- 明瞭公司QMS/CMMI需求管理的要求與做法
- 需求分析實務經驗分享

對象(Object)

- BA - 需求分析師 □ CIM
  - Business Analysis
  - Requirement Analysis
  - Information Architect
- SA – 系統分析師 □ PIM
  - System Analysis
- SD – 系統設計師 □ PSM
  - System Designer

常見問題

- 該從何開始？
- 怎樣才算需求階段做完？
- 需求該寫到多詳細？
- 有沒有漏列需求？
- 需求描述有沒有忽略重要資訊？

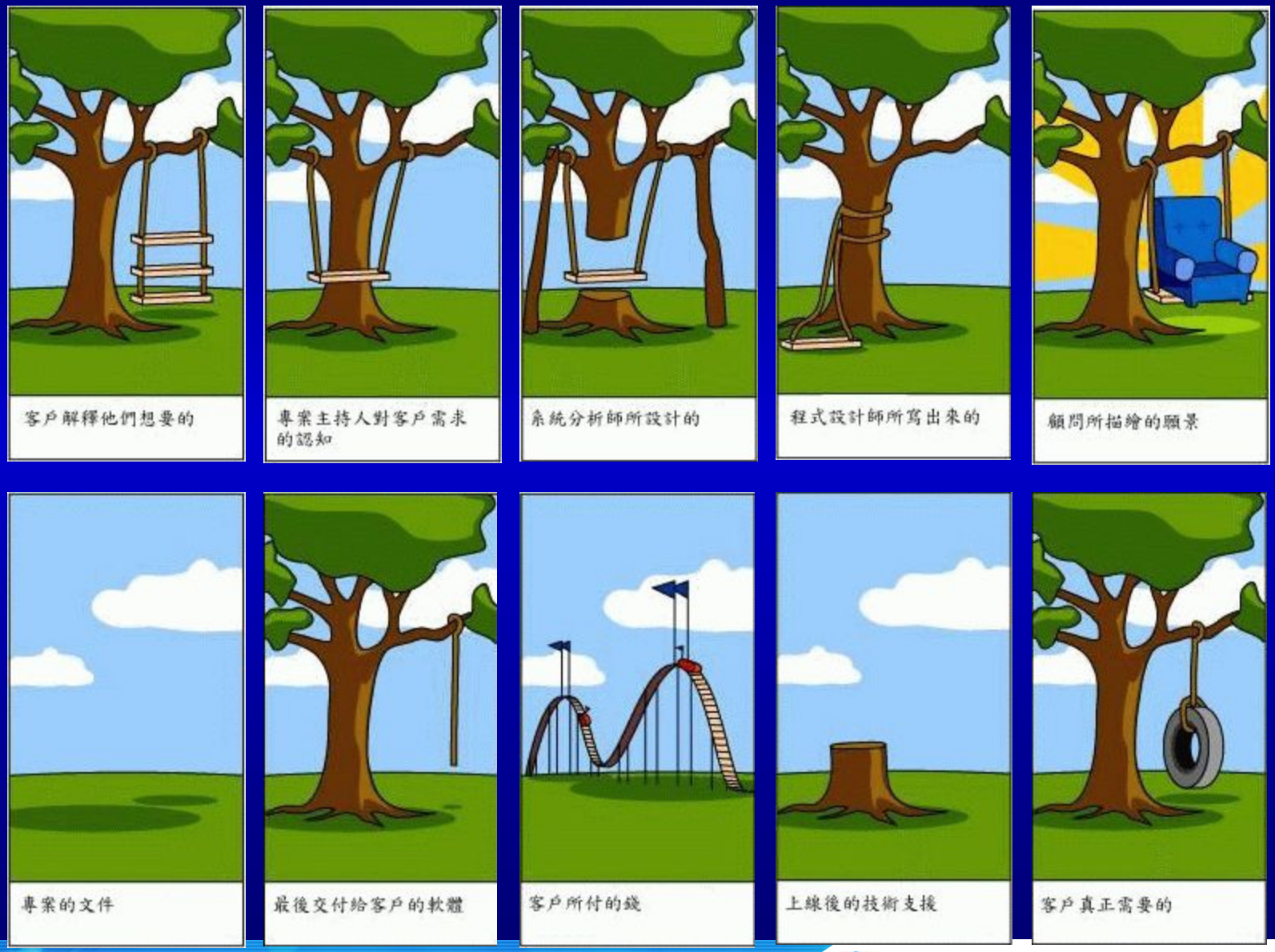
大綱(Agenda)

- 管好需求是確保專案成功的重要第一步
- BA/SA的任務與工作
- 需求發展
  - 計劃→誘導→分析→規格→驗證→承諾
- 需求管理
  - 維護需求追溯
  - 管控需求變更

需求規格 Requirements Specification

- WHY - 為何需求規格如此重要？
- WHO - 誰用到需求規格？誰負責？
- WHAT- 需求規格該是什麼樣子？
- HOW - 如何發展好需求規格？
- WHEN- 從接案到結案到維護
- WHERE-用在哪裡？存在哪裡？

對專案需求的不同認知



專案的焦油坑(需求型)

- 交貨時客戶才說:『這不是我要的』
- 需求發散，範圍愈搞愈大，好像永遠做不完
- 需求不明確，不知道要做到什麼程度才能結案
- 需求改來改去，改到後來系統架構一團亂
- 跟承辦人達成的共識，到長官處全被推翻
- 到底誰說的算？
- Unreadable (often unread) Specification

需求問題徵狀 checklist$_1$

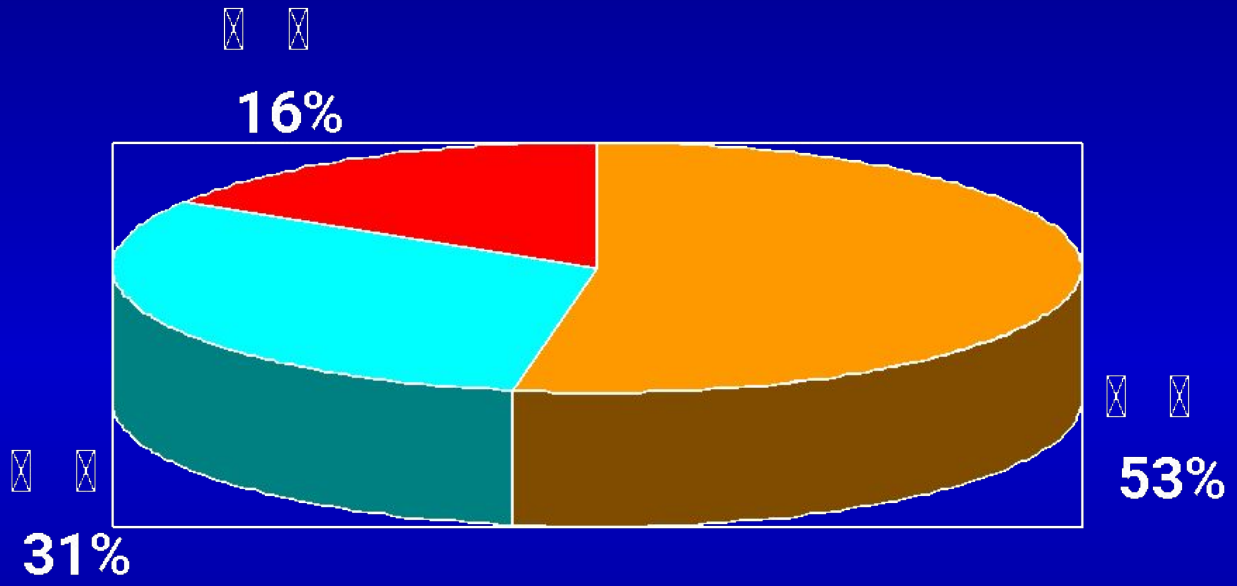
- The project's vision and scope are never clearly defined.
- Customers are too busy to spend time working with analysts or developers on the requirements.
- User surrogates, such as product managers, development managers, user managers, or marketers, claim to speak for the users, but they don't accurately represent user needs.
- Requirements exist in the heads of "the experts" in your organization and are never written down.
- Customers claim that all requirements are critical, so they don't prioritize them.
- Developers encounter ambiguities and missing information when coding, so they have to guess.

需求問題徵狀 checklist$_2$

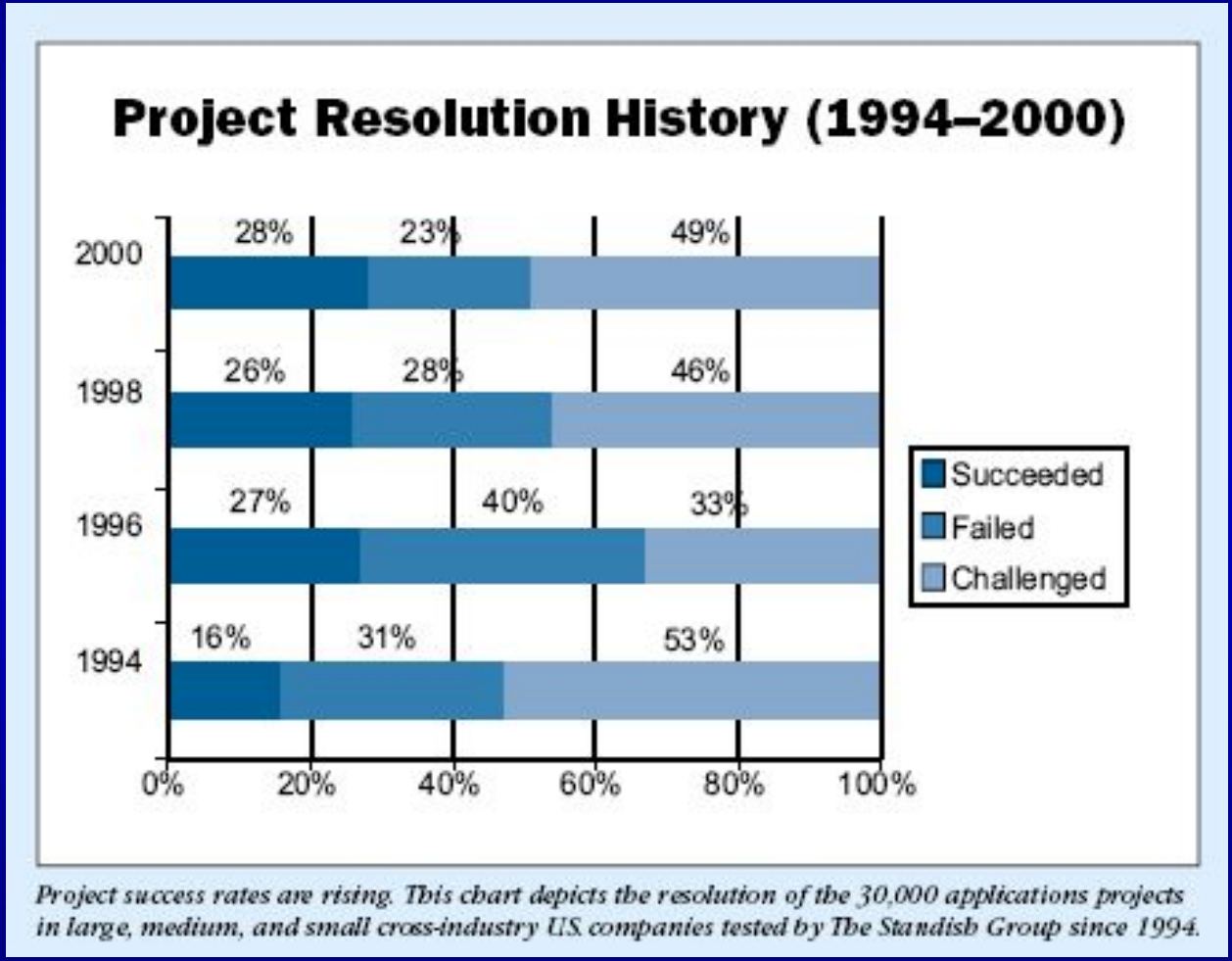
- Communications between developers and customers focus on user interface displays and not on what the users need to do with the software.
- Your customers sign off on the requirements and then change them continuously.
- The project scope increases when you accept requirements changes, but the schedule slips because no additional resources are provided and no functionality is removed.
- Requested requirements changes get lost, and you and your customers don't know the status of all change requests.
- Customers request certain functionality and developers build it, but no one ever uses it.
- The specification is satisfied, but the customer is not.

You Are Not Alone!

You Are Not Alone!



The CHAOS Report



Project success rates are rising. This chart depicts the resolution of the 30,000 applications projects in large, medium, and small cross-industry US companies tested by The Standish Group since 1994.

Source: The Standish Group 1994, 2000

痛苦專案的10大病灶

1. Lack of User Input	12.8%
2. Incomplete Requirements	12.3%
3. Changing Requirements	11.8%
4. Lack of Executive Support	7.5%
5. Technology Incompetence	7.0%
6. Lack of Resources	6.4%
7. Unrealistic Expectations	5.9%
8. Unclear Objectives	5.3%
9. Unrealistic Time Frames	4.3%
10. New Technology	3.7%
Others	23.0%

成功專案的10大特質(1994)

1. User Involvement	15.9%
2. Executive Management Support	13.9%
3. Clear Statement of Requirements	13.0%
4. Proper Planning	9.6%
5. Realistic Expectations	8.2%
6. Smaller Project Milestones	7.7%
7. Competence Staff	7.2%
8. Ownership	5.3%
9. Clear Vision & Objectives	2.9%
10. Hard-working, Focused Staff	2.4%
Others	13.9%

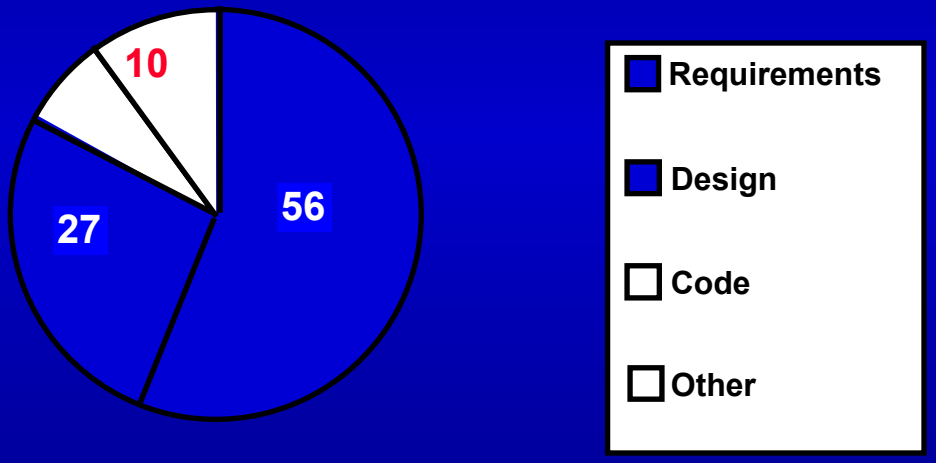
成功專案的 10大特質(2000)

1. Executive Management Support	18%
2. User Involvement	16%
3. Experienced Project Manager	14%
4. Clear Business Objectives	12%
5. Minimized Scope	10%
6. Standard Software Infrastructure	8%
7. Firm Basic Requirements	6%
8. Formal Methodology	6%
9. Reliable Estimates	5%
Others	5%

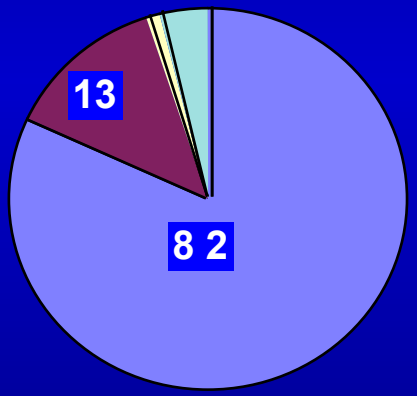
問題源由與克服成本

Software Bugs

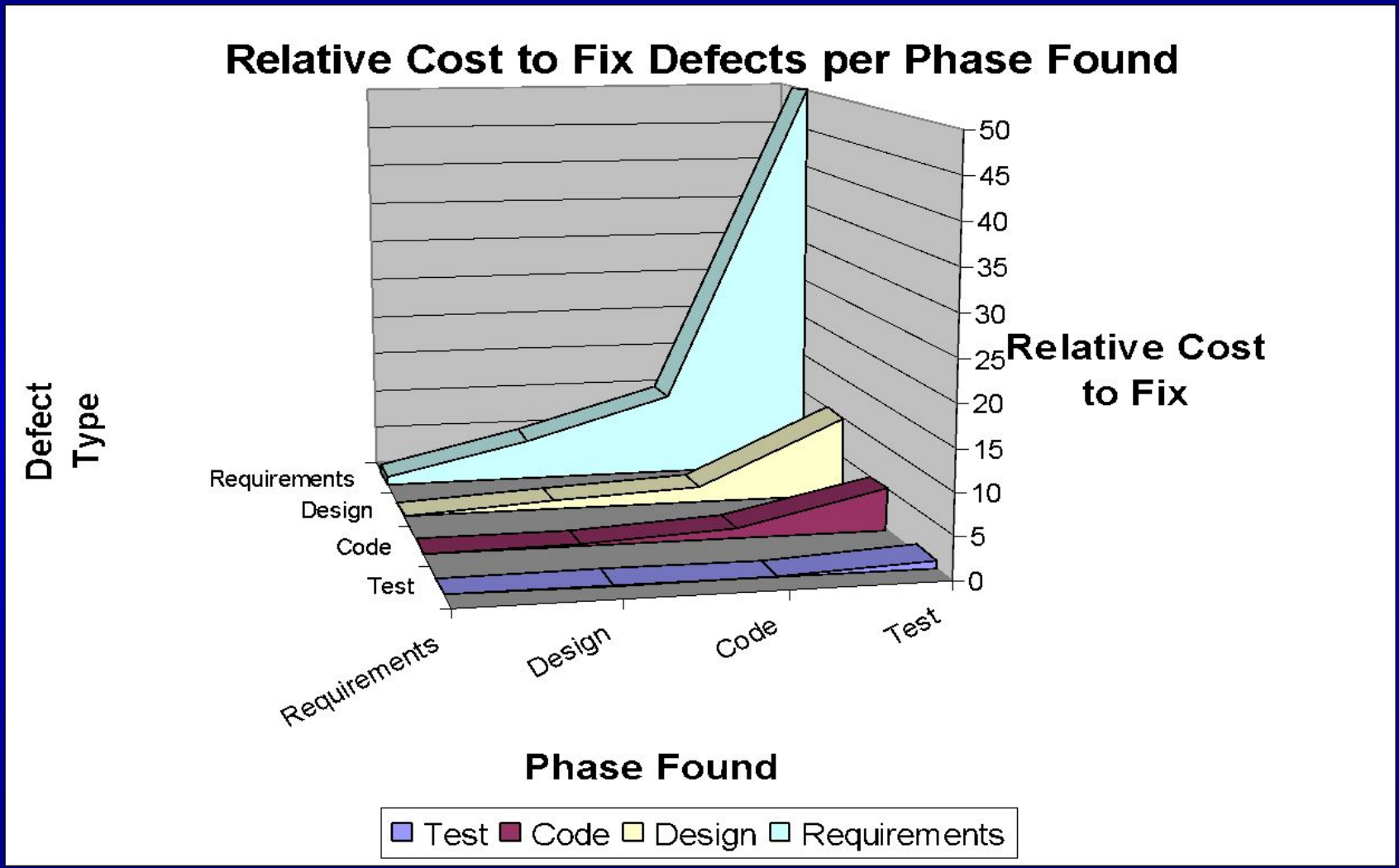
Percentage of Bugs by Source



Percentage of Cost of Fixing Bugs by Source



不同階段發現問題的克服成本



Requirement Engineering Goal

- to develop quality software
  —on time and on budget—
  that meets customers' real needs
- Our challenge is to understand *users'* problems in *their* culture and *their* language and to build systems that meet *their* needs

Build the Right Product

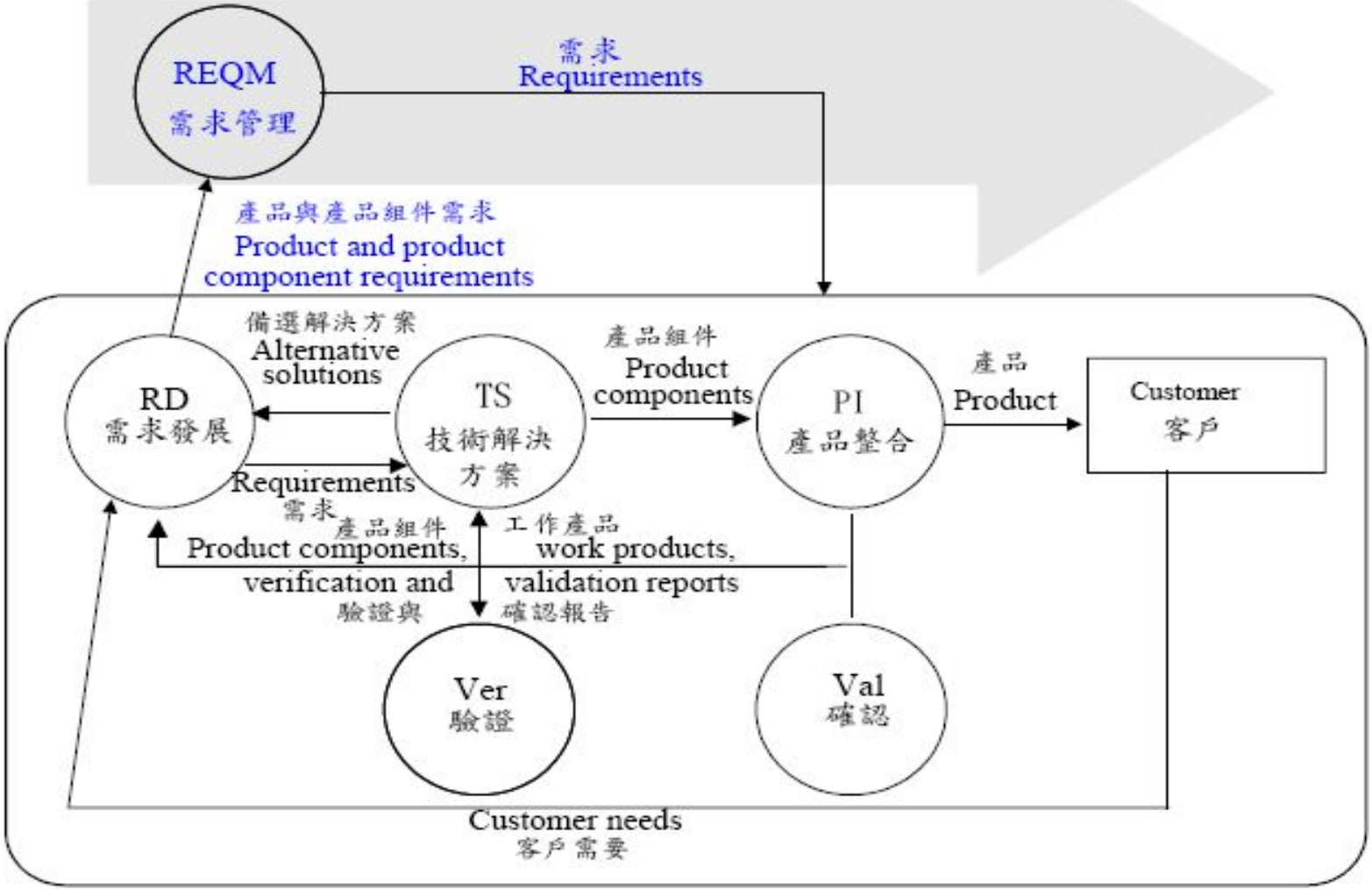
The only effective way is to correctly understand :

- What the product has to do
- How it will be used
- By Whom it will be used
- How it fits into the large picture of the organization

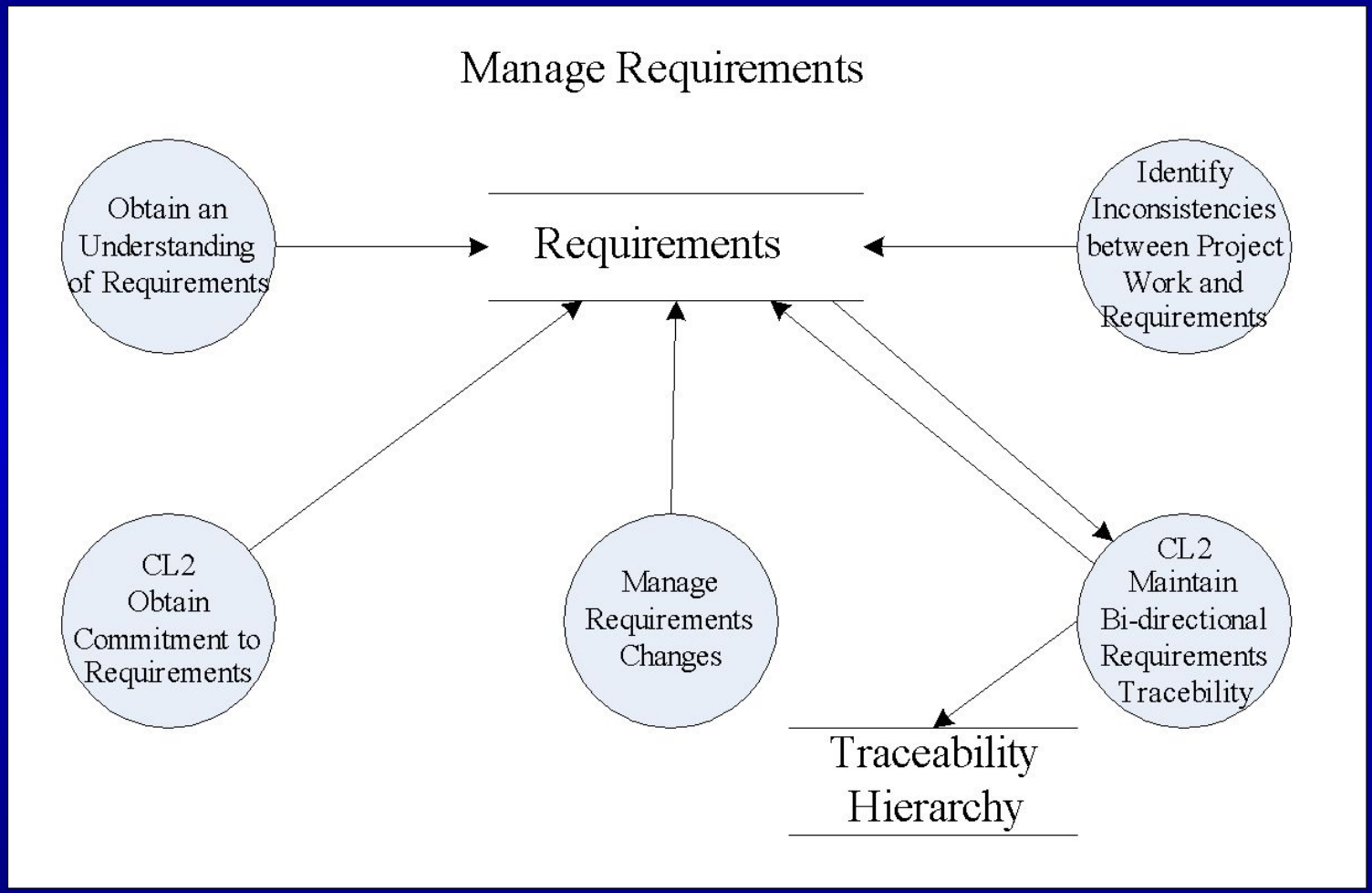
Requirements Specification -- What is it?

- *Everything* about WHAT the proposed new system will do,
  but
- *Nothing* about HOW it will be built or implemented.
- Expressed as an abstraction in technologically neutral way so as to intentionally avoid influencing the design of the solution.

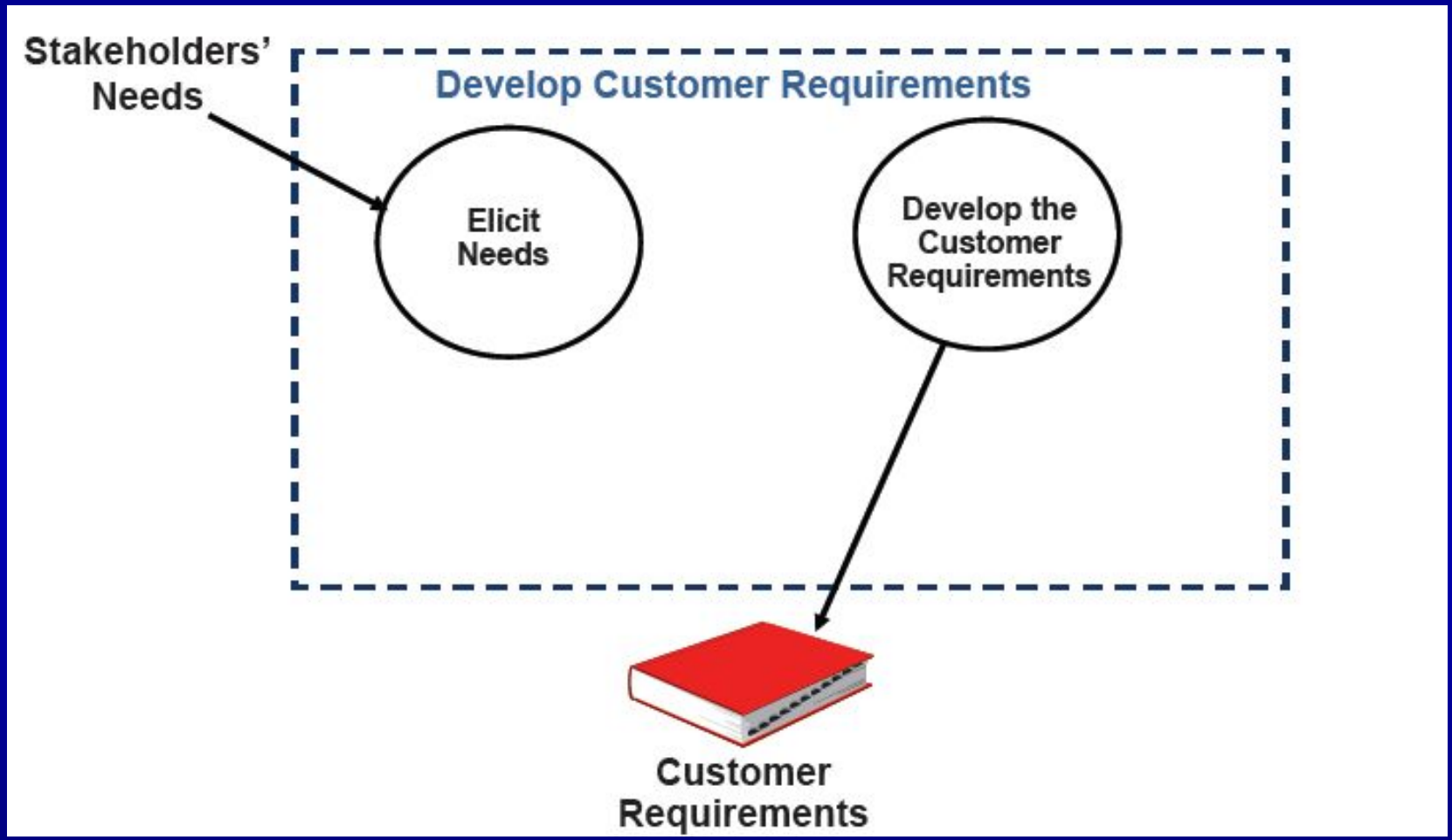
與其它PAs的關係



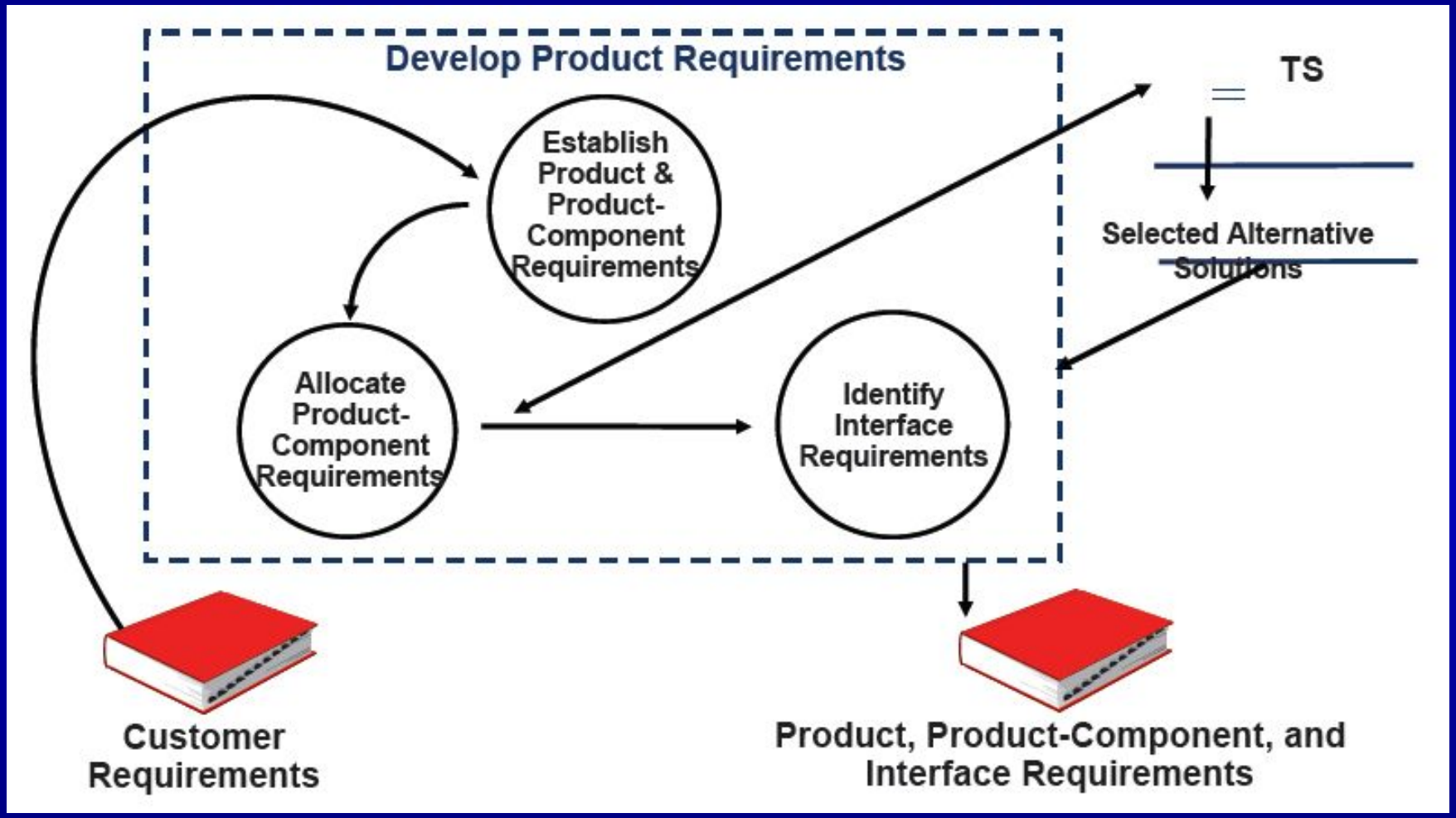
ReqM SGs & SPs



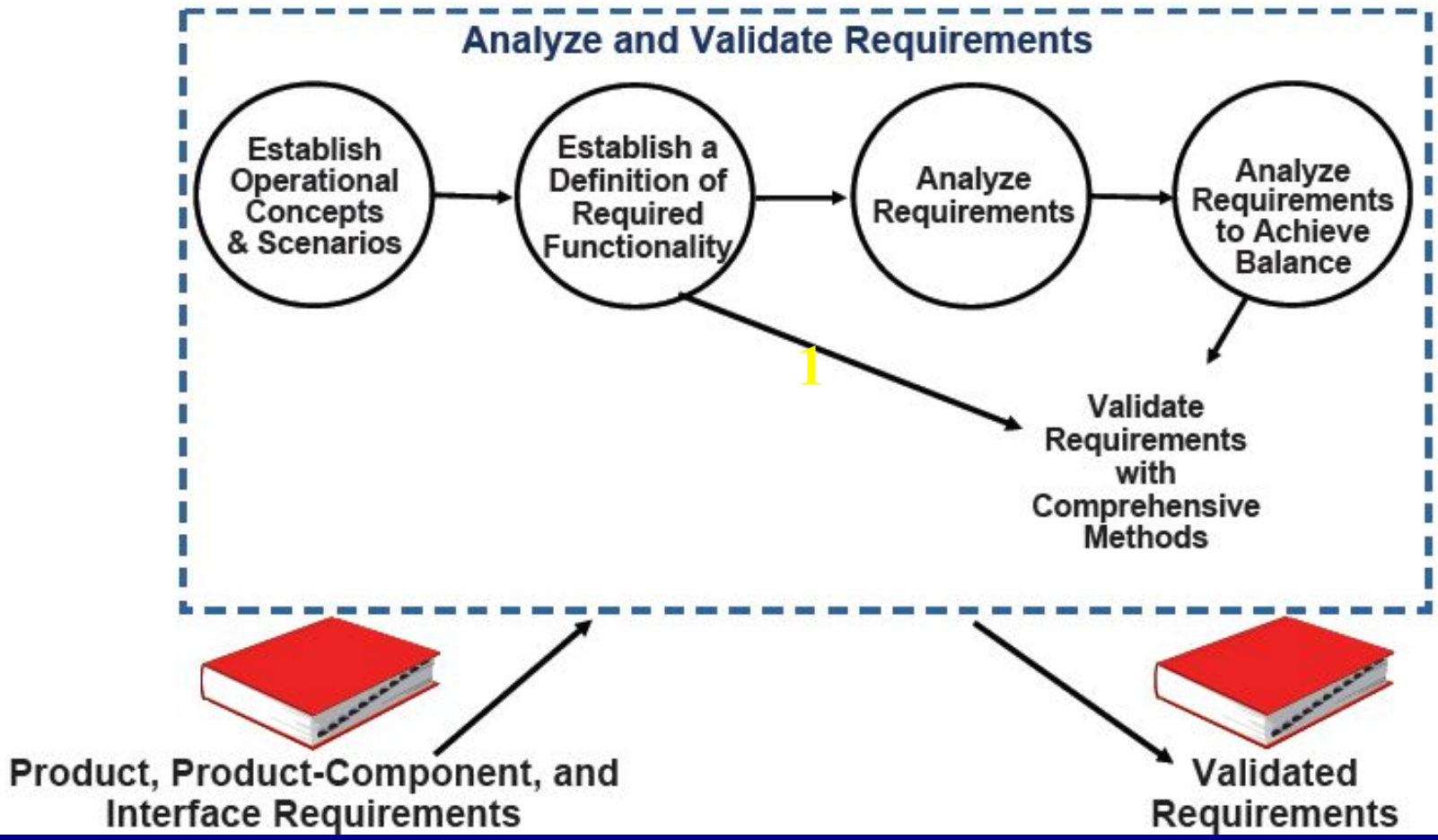
RD SG1



RD SG2



RD SG3

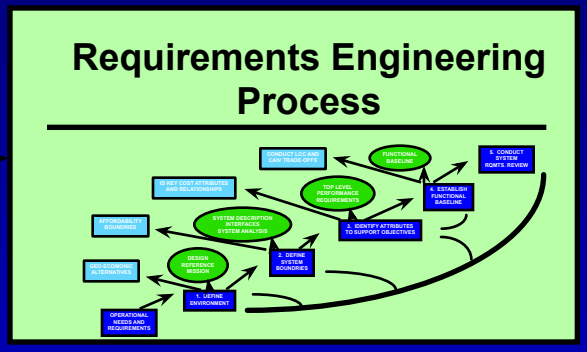


Requirements Engineering Process

Program Requirements

Rules, Orders, Regulations, Standards

Politics & Market Place



Inputs

- Customer / User Needs
  - Performance Requirements
  - System External Interfaces
  - Environmental Requirements
- External Influences
  - Federal Regulations
  - Navy Orders
  - Standards
  - Available Funding

Perform Systems Requirements Analysis and Functional Allocation

- Define customer needs
- Identify requirements
- Decompose requirements to appropriate working level
- Identify top-level functions
- Decompose functions to appropriate working level
- Integrate functional interfaces
- Define performance measures of effectiveness
- Identify Systematic Risks

Outputs

- Top Level System Requirements
- Requirements Traceability Matrix
- System Concept of Operations
- Functional Architecture
- System Functional Block Diagram
- System External Interfaces
- Life Cycle Cost Objectives
- Verification Methodology
- Programmatic Risks
- Technical Performance Measures
- Logistics
- Program Systems Engineering Master Schedule
- Preliminary Models / Simulations

QMS 需求管理流程簡介

- 政策
  - 需求與變更都須記錄、追溯、審查
- 目的
  - 在明確定義的專案範圍下，支持客戶的目標、需要、與期望
  - 有效地產出、分析、管理客戶需求、產品及產品組件需求，並與客戶及關係人等達成需求共識，以確保最終產品能符合客戶的需求
  - 隨時維護一份現行、正確的需求規格，以做為產品發展之依據
  - 建立需求追溯關聯，以提供變更控制時影響分析的資訊依據
  - 有效控制需求變更在對專案時程、成本與品質影響最少的狀況下，並界定這些需求與專案計畫和工作產品間之差異

需求分析師（BA）

- 負責制定需求管理計畫
- 需求瞭解與記錄，與使用者溝通協調
- 發展客戶需求(URS)
- 彙整並製作需求追溯表裡的URS部份
- 分析確認需求
- 依需要提出需求變更申請單
- 變更影響評估

系統分析師（SA）

- 承諾需求
- 建立與維護需求規格(SRS)與需求追溯關聯
- 發展功能性架構，配置System Requirements
- 分析確認需求
- 依需要提出需求變更申請單
- 變更影響評估

RA角色與職責

階段	工作	產出
售前準備	1.參與設計解決方案與決策分析(H13J01-決策分析與解決方案指引) 2.參與專案可行性分析	1. H15J01-決策分析與解決方案記錄表-DAR 2. H15A07-會議紀錄(可行性分析) 3. H14T01-RFP 4. H14T02-建議書 5. H14T05-產品企畫功能規格書 6. H15K03-風險檢核表
專案起始	1.參與專案起始會議 2.制定需求管理計畫(H12M01-需求管理流程)	1. H15A07-會議紀錄(起始會議) 2. H14K01-專案管理計畫書裡的3.3.1需求管理計畫
專案啟動	1.審查專案管理計畫書 2.參與專案啟動會議	1. H15I03-審查意見表(PMP) 2. H15A07-會議紀錄(啟動會議)
專案監控	1.參與專案進度審查會議(至少兩週一次)，專案進度報告 2.撰寫工作週報	1. H15L01-專案執行進度報告 2. H15A09-工作週報

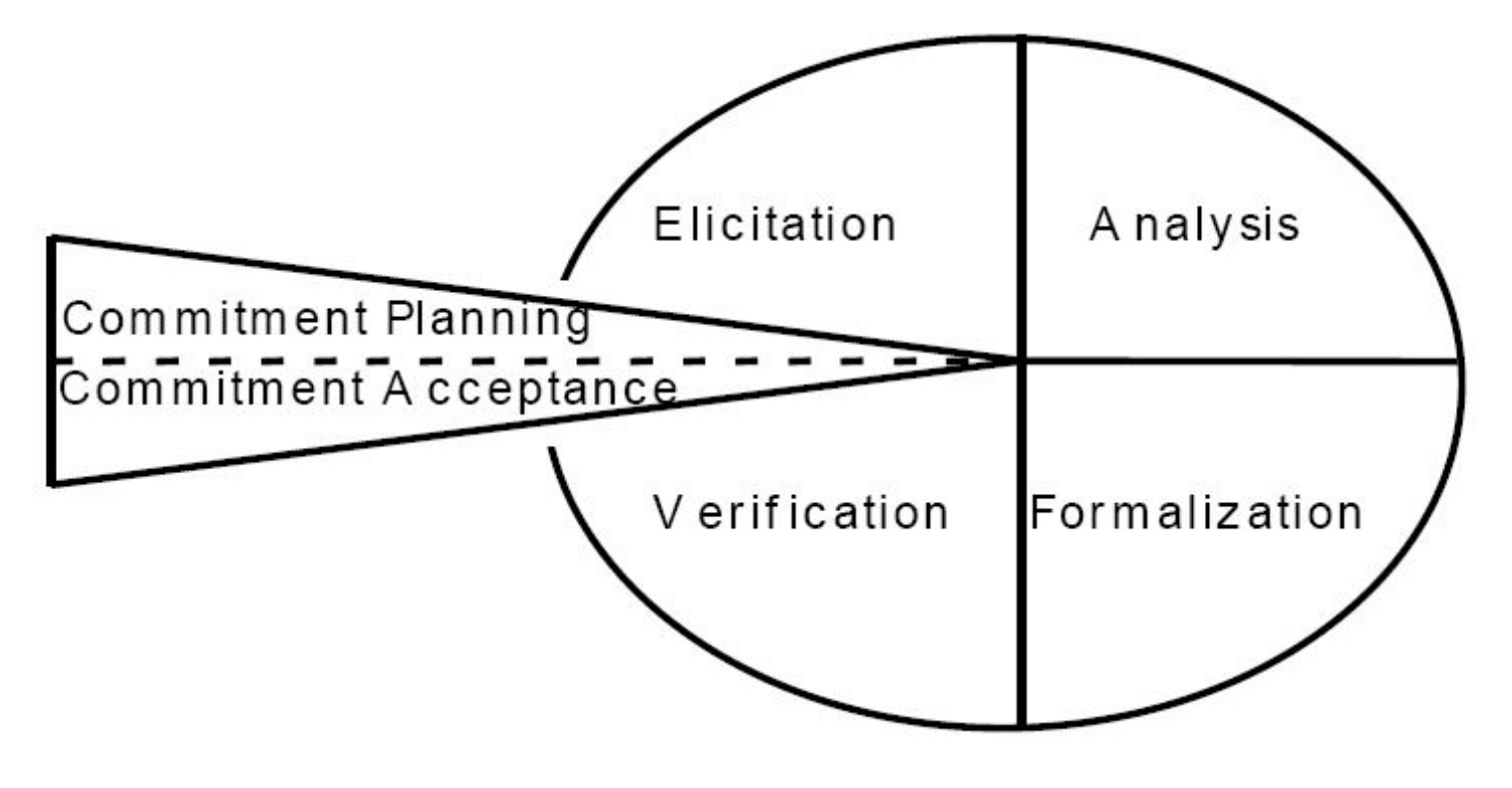
RA角色與職責

階段	工作	產出
需求(BA)	1.發展客戶需求 (H13M11-需求規格發展指引)	1. H14T01-RFP 2. H14T02-建議書 3. H14T05-產品企畫功能規格書 4. H15A07-會議紀錄(需求訪談)
需求(BA)	2.彙整客戶需求 (H13M11-需求規格發展指引)	1. H14M51-需求追溯表 URS-SRS工作表的URS條列
需求(SA)	1.發展軟體需求規格書(SRS) 2.維護需求追溯表(RTM) 3.分析需求規格 4.審查軟體需求規格書(SRS/RTM) 5.確認軟體需求規格書(SRS) 6.申請發佈需求基線	1. H14M21-軟體需求規格書 2. H14M51-需求追溯表 URS-SRS工作表的SRS對應 SRS-SDD-TC工作表的SRS條列 3. H15K03-風險檢核表 4. H15I01-審查記錄(SRS/RTM) H13I02-審查檢查清單 5. H15A07-會議紀錄(需求規格) 6. H15B11-基線發佈申請單

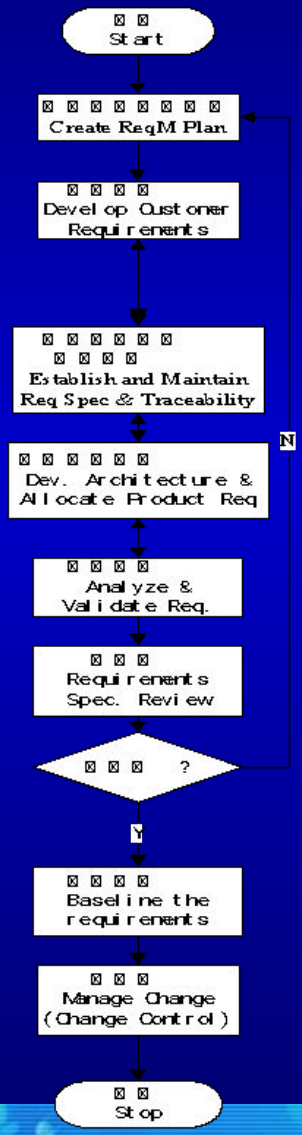
RA角色與職責

階段	工作	產出
設計	1.審查系統設計規格書	1. H15I03-審查意見表(SDD)
測試	1.審查軟體測試計畫書 2.審查軟體測試描述書 3.參與測試回饋會議	1. H15I03-審查意見表(STP) 2. H15I03-審查意見表(STD) 3. H15A07-會議紀錄(測試回饋會議)
驗收	1.協助客戶進行驗收測試 2.準備驗收項目 3.參與專案結案會議	1. 2. 3. H15A07-會議紀錄(結案會議)
需求變更	1.依需要提出需求變更申請單 2.變更影響評估 (H13B51-變更管制指引)	1. H15B51-變更申請單

需求發展Spiral



H12M01-需求管理流程



Person in Charge	Template/Form	Operation Procedure/ Working Instruction
(BA) (PM)	ReqM Plan Template	ReqM Process
(BA) (Customer Req. Representative)	Req. Interview MOM Customer Use Cases/Operation Scenario Req. Spec. Checklist (Domain/General)	Requirements Elicitation Guideline
(BA) (SA)	SRSDS System Requirement Spec. (SRS) Req. Traceability Matrix	SRSDS Tutorial SRS Guideline
(SA)	SRSDS	SRSDS Tutorial
(BA) (SA)	SRSDS	SRSDS Tutorial
(PM) (BA) (SA) (Prj. Member) (Test Staff) (Customer Req. Representative) (Dept. Head)	SRS Checklist	Review Guideline
(RA) (SA) (CMR) (QAR)	SRS (Baselined) Requirements Audit. Checklist	CM Process PPQA Process
(PM) (RA) (SA) (CCB)	Change Request Form Impact Analysis Change Track Log	Change Control Guideline

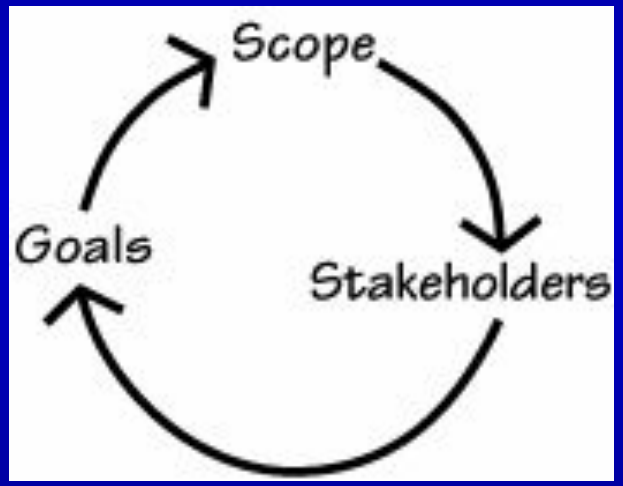
需求管理計畫（in PMP 3.3.1）

- 定義目標(Vision)和範圍(Scope)
- 界定不同類別角色的使用者
  - 界定外接系統
- 建立需求審查準則
  - 需求提供準則(Providence Criteria)
  - 需求接受準則(Acceptance Criteria)
- 安排需求發展活動及時程
- 載明需求變更程序
  - 參照H14B01-建構管理計畫書: 第6節.變更管制

需求發展計畫（RDP）

- Define Product Vision & Project Scope
- Identify Classes (Roles) of Users
- Identify External Systems
- Establish Requirements Review Criteria
  - Providence Criteria
  - Acceptance Criteria
- Plan the Resource and Schedule
- Constraints
- Terminology (Project Glossary)
- Relevant Facts and Assumptions
- The Risk

Trinity of SGS



- We are talking about the work people do.
- To build the right product, you have to
  - understand the work,
  - the people who do or influence the work,
  - and the end they are trying to achieve.

Pattern: SharedClearVision

- Problem:
  - The lack of a clear vision about a system can lead to indecision and contrary opinions among the stakeholders and can quickly paralyze the project.
- Solution:
  - Prepare a statement of purpose for the system that clearly describes the objectives of the system and supports the mission of the organization. Freely distribute it to everyone involved with the project.

Define the Goal

- Business Requirements
  - Background
  - Business Opportunity
  - Business Objectives & Success Criteria
  - Customer or Market Needs
  - Business Risks
- Vision of the Solution
  - Vision Statement
  - Major Features
  - Assumptions and Dependencies

Goal Analysis

- Purpose
- Advantage
- Measurement
- Reasonable
- Feasible
- Achievable

Pattern: VisibleBoundary

- Problem:
  - The scope of a system will grow in an uncontrollable manner if you do not know its boundaries.
- Solution:
  - Establish a visible boundary between the system and its environment by enumerating both the people and equipment that interact with the system.

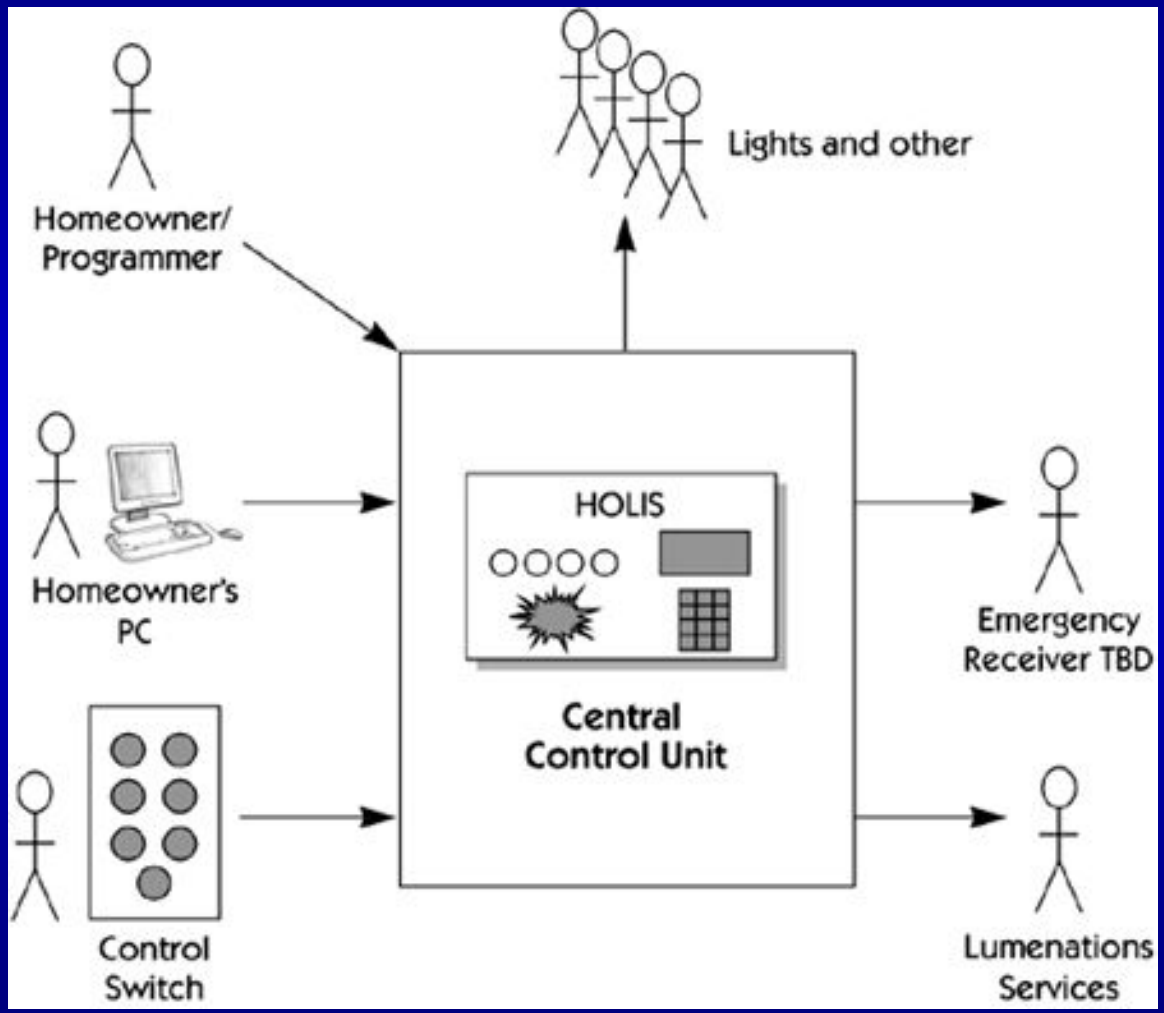
Define the Scope

- Scope and Limitations
  - Scope of the Initial Release
  - Scope of Subsequent Releases
  - Limitations and Exclusions
- Business Context
  - Stakeholder Profiles
  - Project Priorities
    - Features, Quality, Schedule, Cost, Staff
  - Operating Environment

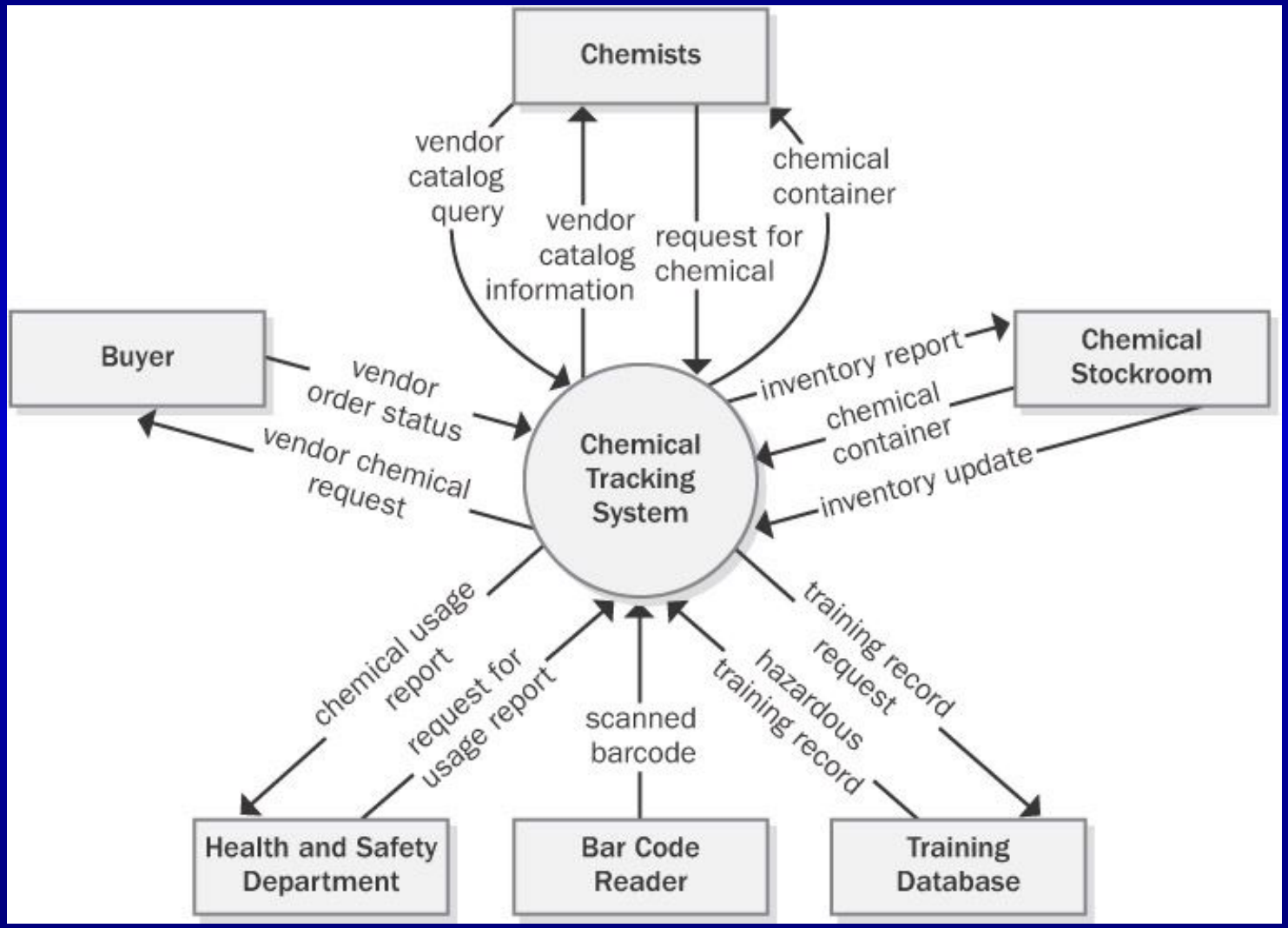
Context Diagram

- Define a boundaries of the system to be developed
- Identify external interfaces
  - Items consider to be outside the scope
  - Connections between the system and the outside world
- Identify stakeholders

Context Diagram example



Context Diagram example



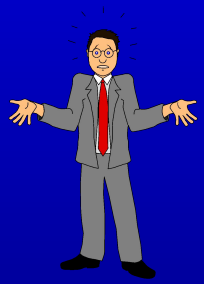
Identify Stakeholders

Customers

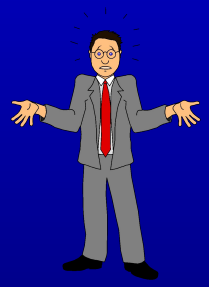
Users

Project Manager

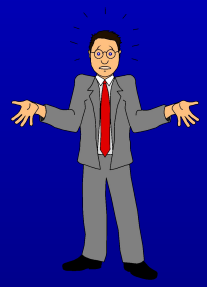
RA (BA/SA)



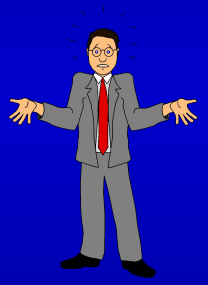
Sales, Marketing,
Field Support,
Help Desk



Doc Writers,
Legal Staff,
Manufacturing

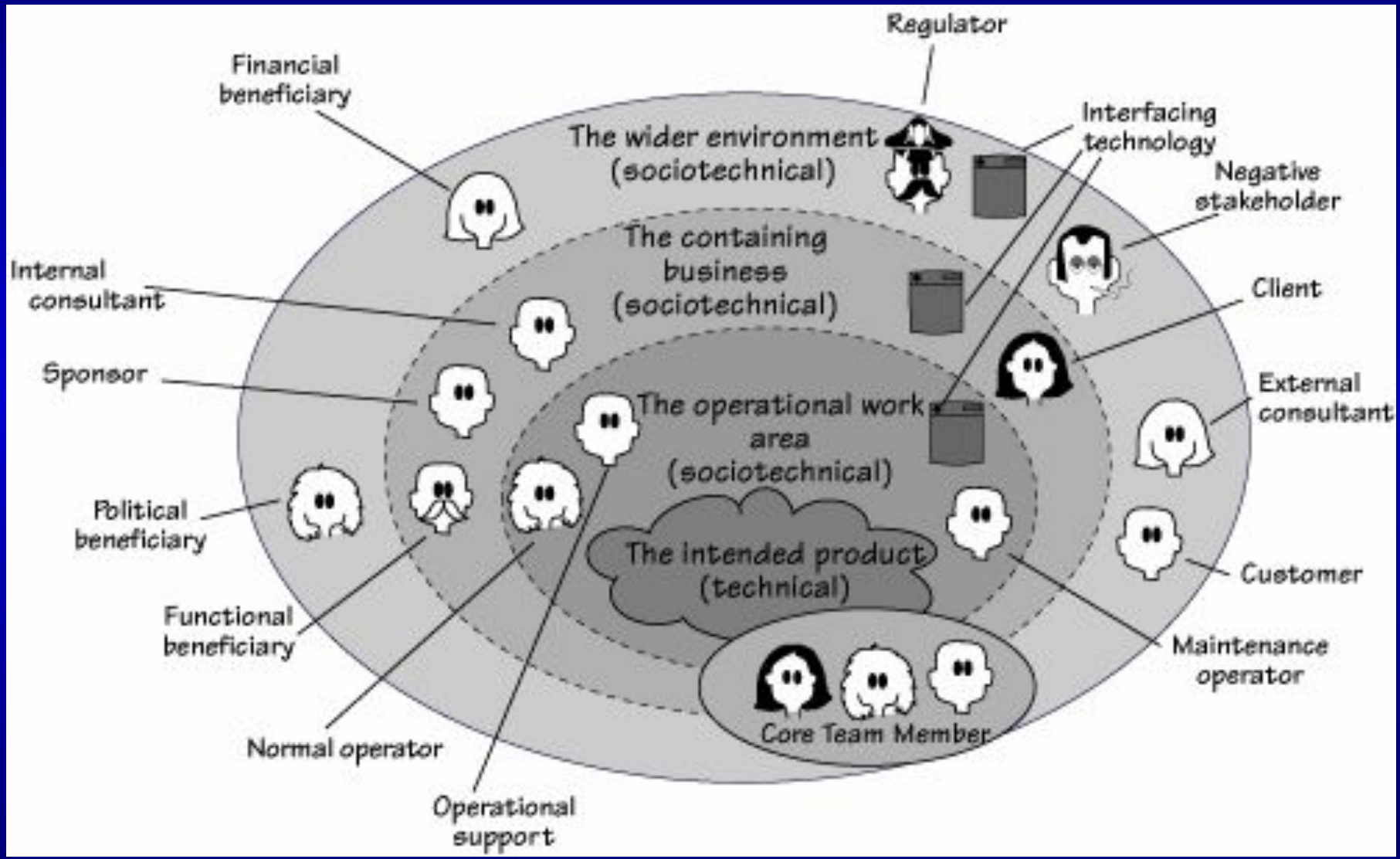


Developers



Testers

Stakeholder Map



The Client (客戶)

- Pays for the development of the product
- Has final say on acceptance of the product thus must be satisfied with the product as delivered
- particularly, the usability, adaptability, and productization requirements

The Customer (顧客)

- The customers buy the product
- You have to understand your customers well enough to specify a product they will buy and find useful and pleasurable
- What appeals to your customers. What requirements will they pay for? What will they find useful? What window-dressing features are attractive, and what is downright trivial?

The Users (用戶) Get to Know Them

- Who will ultimately operate your product
- Identifying the users is the first step in understanding the work they do
- Usability requirements and Functional requirements
- eg., AAA accessibility

Users Classes List

- User Class Name
- User Role / Population
- Major Value or Benefit from the Product
- Major Features and Usage Frequency
- Subject/Technical matter experience
- Priorities Assigned to Users
- User Participation
  - Describe the contribution that you expect these users to provide—for example, business knowledge, interface prototyping, or usability requirements

Pattern: ClearCastOfCharacters

- Problem:
  - If we analyze only the uses of the system, and ignore the roles they play with respect to the system, then we can miss important system behavior or introduce redundant behavior.
- Solution:
  - Identify the actors the system must interact with and the role each plays with respect to the system. Clearly describe each.

Pattern: ParticipatingAudience

- Problem:
  - You cannot satisfy stakeholders' need without their input and feedback.
- Solution:
  - Actively involve your customers and internal stakeholders in the use case development process when possible.

Pattern: BalancedTeam

- Problem:
  - Using teams of similar, like-minded individuals to develop use cases can result in a set of limited, narrowly ranged use cases that do not satisfy everyone's needs.
- Solution:
  - Staff team with people from different specialties to champion the interests of the stakeholders in the development process. Make sure the team contains both developers and end uses.

Pattern: SmallWritingTeam

- Problem:
  - Using too many people to write a use case is inefficient, and the compromises made to align the many different points of view may result in a less than satisfactory system.
- Solution:
  - Restrict the number of people refining any one work product to just two or three people. Use a TwoTierReview process to include more people.

Mandated Constraints

- Solution Constraints
- Implementation Environment
- Partner or Collaborative Applications
- Off-the-Shelf Software
- Anticipated Workplace Environment
- Schedule Constraints
- Budget Constraints

Project Glossary

- Definitions of All Terms, Including Acronyms, Used in the Project
- Data Dictionary

發展客戶需求

參考 H13M11-需求發展指引

- 需求誘導 (Requirement Elicitation)
  - 文件研讀
  - 需求討論會議
  - 需求訪談
  - 現場觀察
  - 雛型
- 用戶需求規格 (URS)
  - RFP，建議書，訪談紀錄...
  - 彙總在H14M51-需求追溯表的URS-SRS表裡條列URS

Requirements Elicitation

- Documents Study
  - **RFP/Proposal, System requirements specifications**
  - **Documents describe current or competing products**
  - **Problem reports, enhancement requests, customer complains for current systems**
  - **Marketing/competitive surveys and user questionnaires**
- Elicitation Workshop
- Interview & Discussions w/ potential users
- Observe Users at Work
- Scenario analysis of user tasks
- Events and responses
- Prototyping
- Web Collaboration (eg., Wiki / Blog)

Elicitation Workshop

- Establish ground rules
- Stay in scope
- Use parking lots to capture items for later consideration
- Timebox discussions
- Keep the team small and include the right participants
- Keep everyone engaged

Interview Template

Part I: Establishing the Customer or User Profile

Name:
Company:
Industry:
Job title:
(The above information can typically be entered in advance.)
What are your key responsibilities?
What outputs do you produce?
For whom?
How is success measured?
Which problems interfere with your success?
What, if any, trends make your job easier or more difficult?

Part II: Assessing the Problem

For which *[application type]* problems do you lack good solutions?
What are they? *(Hint: Keep asking, "Anything else?")*
For each problem, ask the following questions.

- Why does this problem exist?
- How do you solve it now?
- How would you like to solve it?

Part III: Understanding the User Environment

Who are the users?
What is their educational background?
What is their computer background?
Are users experienced with this type of application?
Which platforms are in use?
What are your plans for future platforms?
Are additional applications in use that are relevant to this application? If so, let's talk about them a bit.
What are your expectations for usability of the product?
What are your expectations for training time?
What kinds of user help (for example, hard copy and online documentation) do you need?

Part VII: Assessing the Opportunity

Who in your organization needs this application?
How many of these types of users would use the application?
How would you value a successful solution?

Part VIII: Assessing the Reliability, Performance, and Support Needs

What are your expectations for reliability?
What are your expectations for performance?
Will you support the product, or will others support it?
Do you have special needs for support?
What about maintenance and service access?
What are the security requirements?
What are the installation and configuration requirements?
Are there special licensing requirements?
How will the software be distributed?
Are there labeling and packaging requirements?

Part IV: Recap for Understanding

You have told me:
(List customer-described problems in your own words.)

-
-
-

Does this adequately represent the problems you are having with your existing solution?
What, if any, other problems are you experiencing?

Part V: The Analyst's Inputs on the Customer's Problem

(Validate or invalidate assumptions.)
(If not yet addressed) Which, if any, problems are associated with: *(List any needs or additional problems you think should concern the customer or user.)*

-
-
-

For each suggested problem, ask the following questions.

- Is this a real problem?
- What are the reasons for this problem?
- How do you currently solve the problem?
- How would you like to solve the problem?
- How would you rank solving these problems in comparison to others you've mentioned?

Part VI: Assessing Your Solution *(if applicable)*

(Summarize the key capabilities of your proposed solution.)
What if you could:

-
-

How would you rank the importance of these?

Part IX: Other Requirements

Are there any legal, regulatory, or environmental requirements or other standards that must be supported?
Can you think of any other requirements we should know about?

Part X: Wrap-up

Are there any other questions I should be asking you?
If I need to ask follow-up questions, may I give you a call? Would you be willing to participate in a requirements review?

Part XI: The Analyst's Summary

After the interview, and while the data is still fresh in your mind, summarize the three highest-priority needs or problems identified by this user/customer.

1.
2.
3.

Context-Free Process Questions

- Who is the client?
- What is a highly successful solution really worth to this client?
- What is the real reason for wanting to solve this problem?
- How much time do we have for this project?
- What is your trade-off between time and value?
- Where else can the solution to this design problem be obtained?
- Can we copy something that already exists?

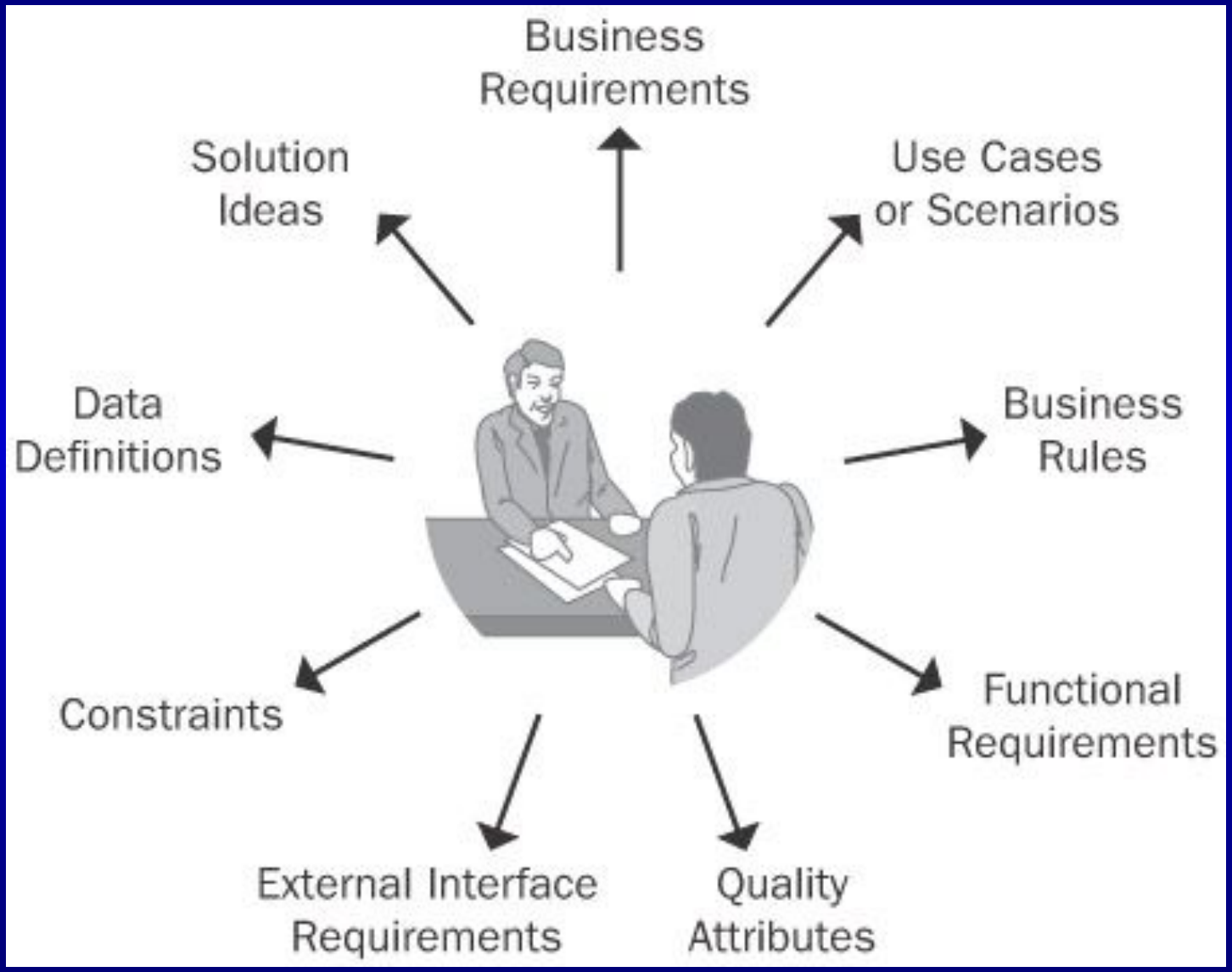
Context-Free Product Questions

- What problems does this product solve?
- What problems could this product create?
- What Environment is this product likely to encounter?
- What kind of precision is required or desired in the product?

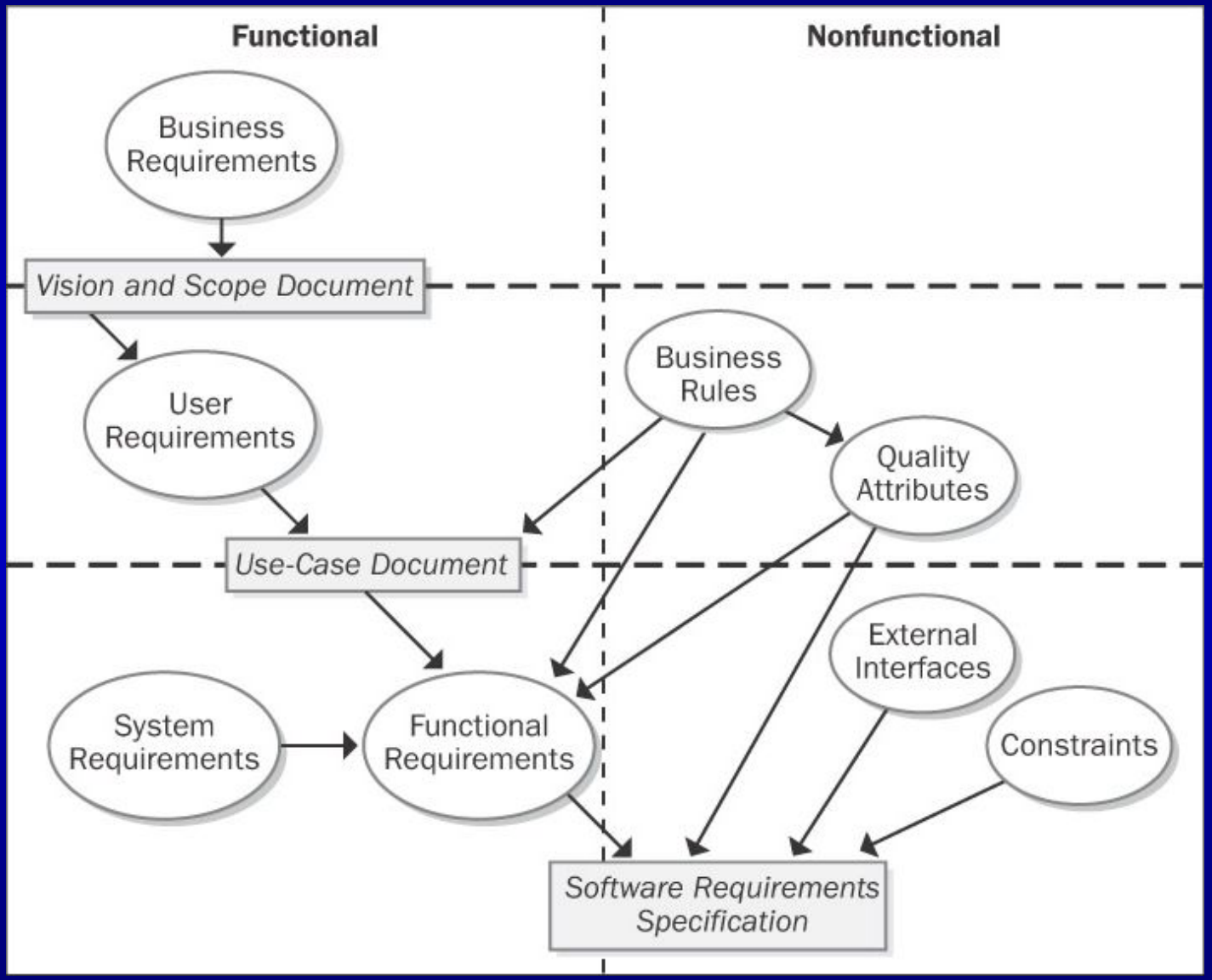
Context-Free Meta-Questions

- Am I asking you too many questions?
- Do my questions seems relevant?
- Are you the right person to answer these questions?
- Are your answers official?
- Asked at the end of every interaction:
  - Is there anything else I should be asking you?
  - Is there anything you would like to ask me?
  - May I return or call you with more questions later, in case I don't cover everything this time?

Classifying Customer Input



不同性質的需求資訊



URS vs. SRS

- URS (User Requirements Specification)
  BA (Business Analyst負責)
  - RFP, Contract, Proposal
  - System Requirements Spec, Interface Spec
  - 使用者提供的文件
  - 需求訪談記錄, 相關附件
  - Business Use Cases
  - Business Rules
- SRS (Software Requirements Specification)
  SA (System Analyst負責)
  - Itemized 需求管控單元

發展功能架構配置 產品需求

H14M21-軟體需求規格書 (SRS)

- 發展功能架構配置產品需求
- 分割模組配置產品組件需求(分割準則)
- 效能需求、設計限制
- 界定產品與外部的介面，以及產品內部的介面

IEEE 830 SRS standard

1. **Introduction**
   - 1.1 Purpose
   - 1.2 Document Conventions
   - 1.3 Intended Audience and Reading Suggestions
   - 1.4 Project Scope
   - 1.5 References
2. **Overall Description**
   - 2.1 Product Perspective
   - 2.2 Product Features
   - 2.3 User Classes and Characteristics
   - 2.4 Operating Environment
   - 2.5 Design and Implementation Constraints
   - 2.6 User Documentation
   - 2.7 Assumptions and Dependencies
3. **System Features**
   - 3.x System Feature X
     - 3.x.1 Description and Priority
     - 3.x.2 Stimulus/Response Sequences
     - 3.x.3 Functional Requirements
4. **External Interface Requirements**
   - 4.1 User Interfaces
   - 4.2 Hardware Interfaces
   - 4.3 Software Interfaces
   - 4.4 Communications Interfaces
5. **Other Nonfunctional Requirements**
   - 5.1 Performance Requirements
   - 5.2 Safety Requirements
   - 5.3 Security Requirements
   - 5.4 Software Quality Attributes
6. **Other Requirements**

Appendix A: Glossary

Appendix B: Analysis Models

Appendix C: Issues List

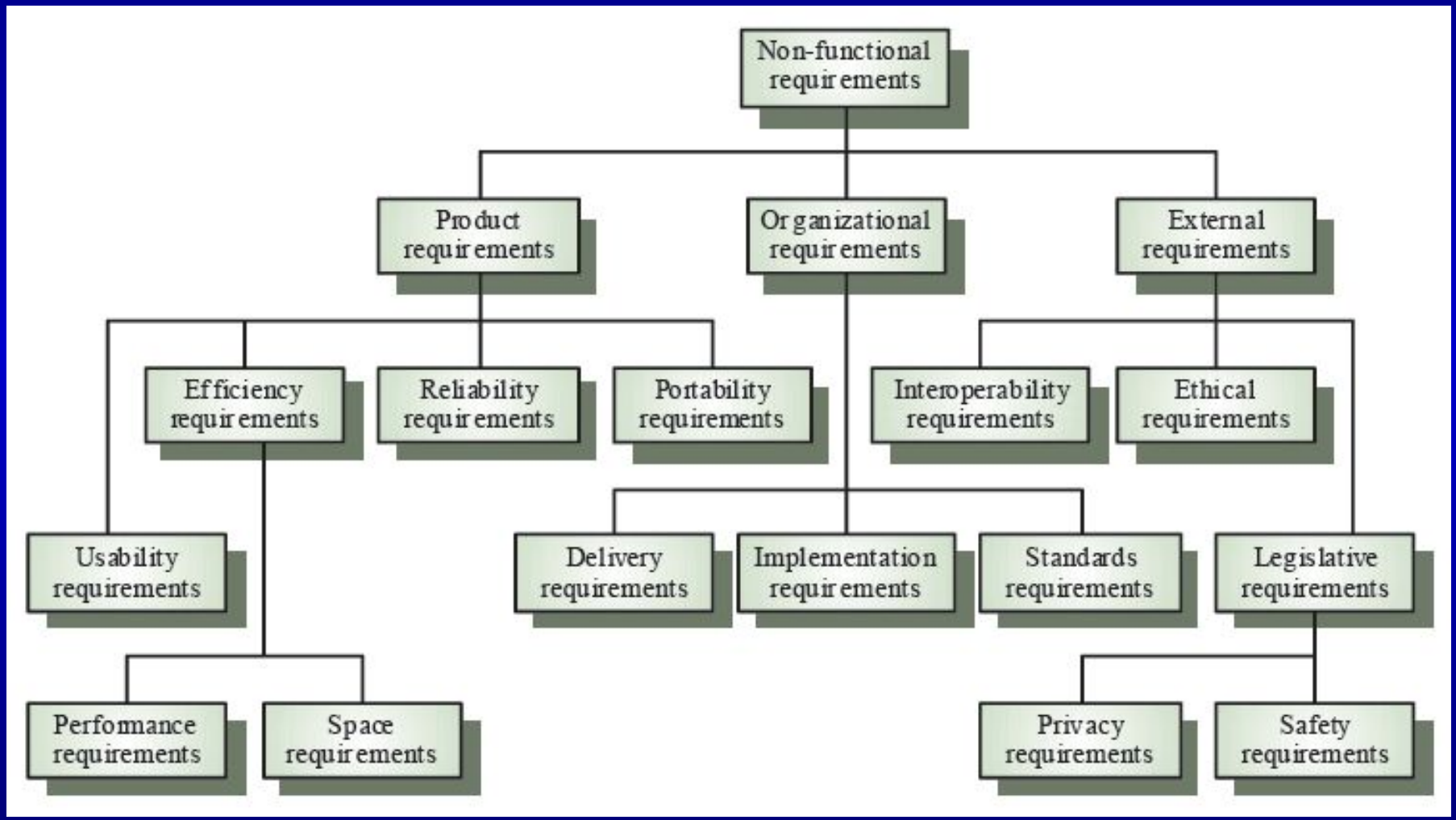
Functional vs. Nonfunctional

- If you can model a requirement by writing a narrative description of the interaction between actor and system, then it is a functional requirement
- Use cases can be used to document any functional requirement

NFR: Nonfunctional Requirements

- **Requirements that specify how well your product does what it does.**
- **May be more critical than functional requirements. If these are not met, the system is useless.**
  - Ask the users about response time needs, and ask them what annoys them about the way the system works now. The answers will give you valuable leads for nonfunctional requirements.
  - If there is a legacy system, it can be another good source for nonfunctional requirements. Ask users what they liked and disliked about it.

Nonfunctional Requirements



Important to Users

- Availability
- Efficiency
- Flexibility
- Integrity
- Interoperability
- Reliability
- Robustness
- Usability

Important to Developers

- Maintainability
- Portability
- Reusability
- Testability

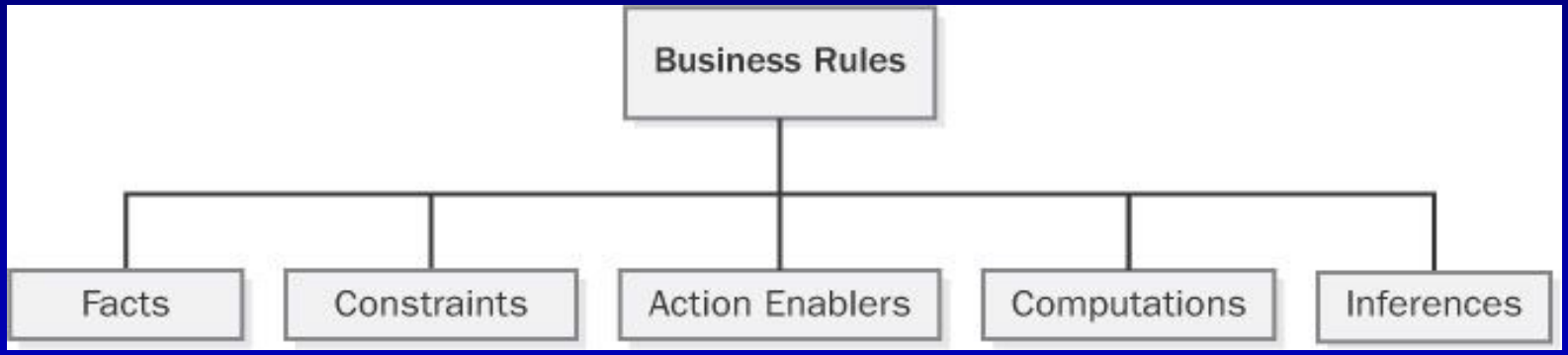
NFR Trade-offs

	Availability	Efficiency	Flexibility	Integrity	Interoperability	Maintainability	Portability	Reliability	Reusability	Robustness	Testability	Usability
Availability							+		+			
Efficiency			–		–	–	–	–		–	–	–
Flexibility		–		–		+	+	+			+	
Integrity		–			–				–		–	–
Interoperability		–	+	–			+					
Maintainability	+	–	+					+			+	
Portability		–	+		+	–			+		+	–
Reliability	+	–	+			+				+	+	+
Reusability		–	+	–	+	+	+	–			+	
Robustness	+	–						+				+
Testability	+	–	+			+		+				+
Usability		–								+	–	

非功能需求度量方式

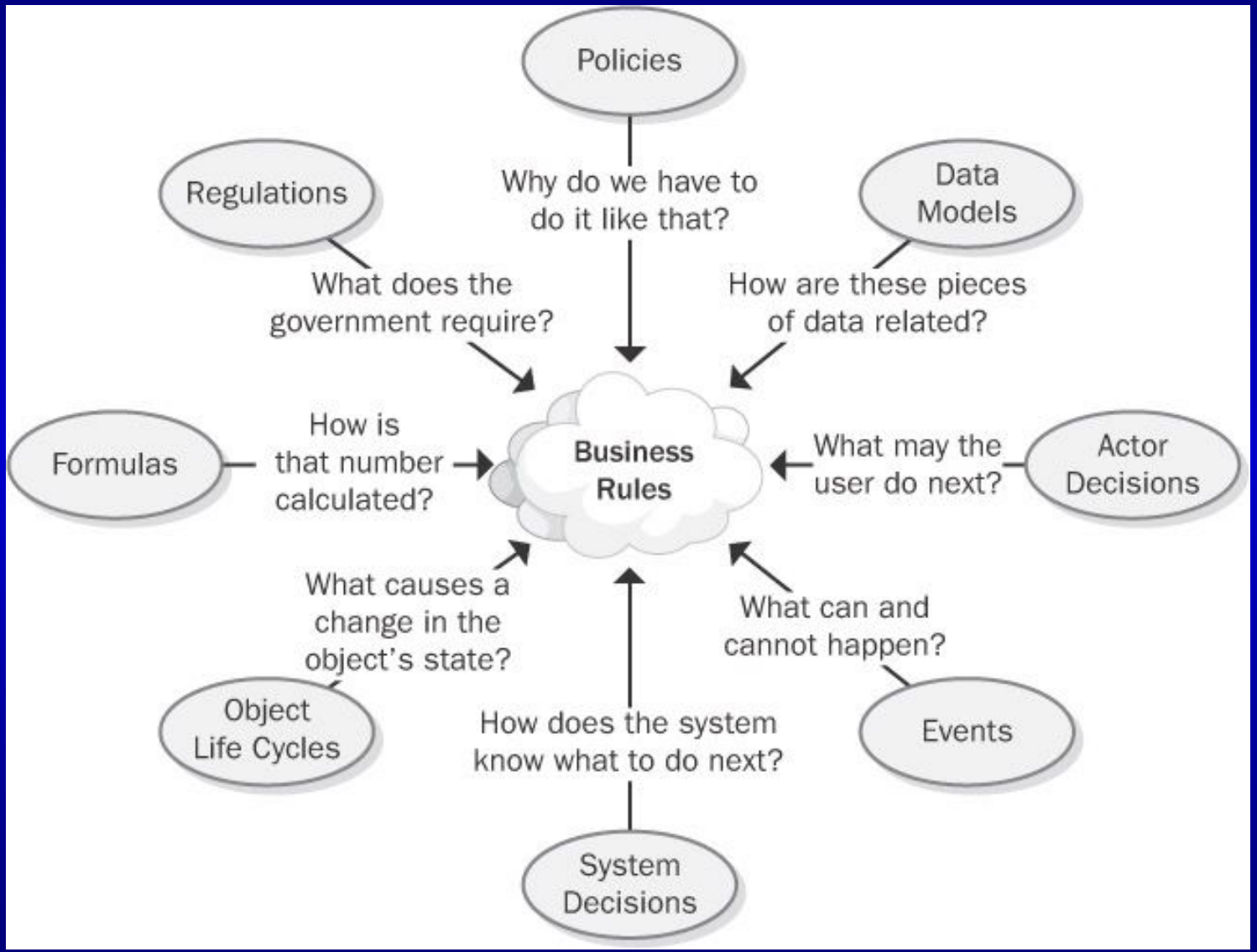
Speed	Processed transactions/second User/Event response time Screen refresh time
Size	K Bytes
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

Business Rules □ Volatility



- Fact: simply true statements
- Constraint: restrict the actions may perform
- Action Enabler: triggers some activity
- Inference: establishes some new knowledge
- Computation: performed using specific mathematical formulas or algorithms

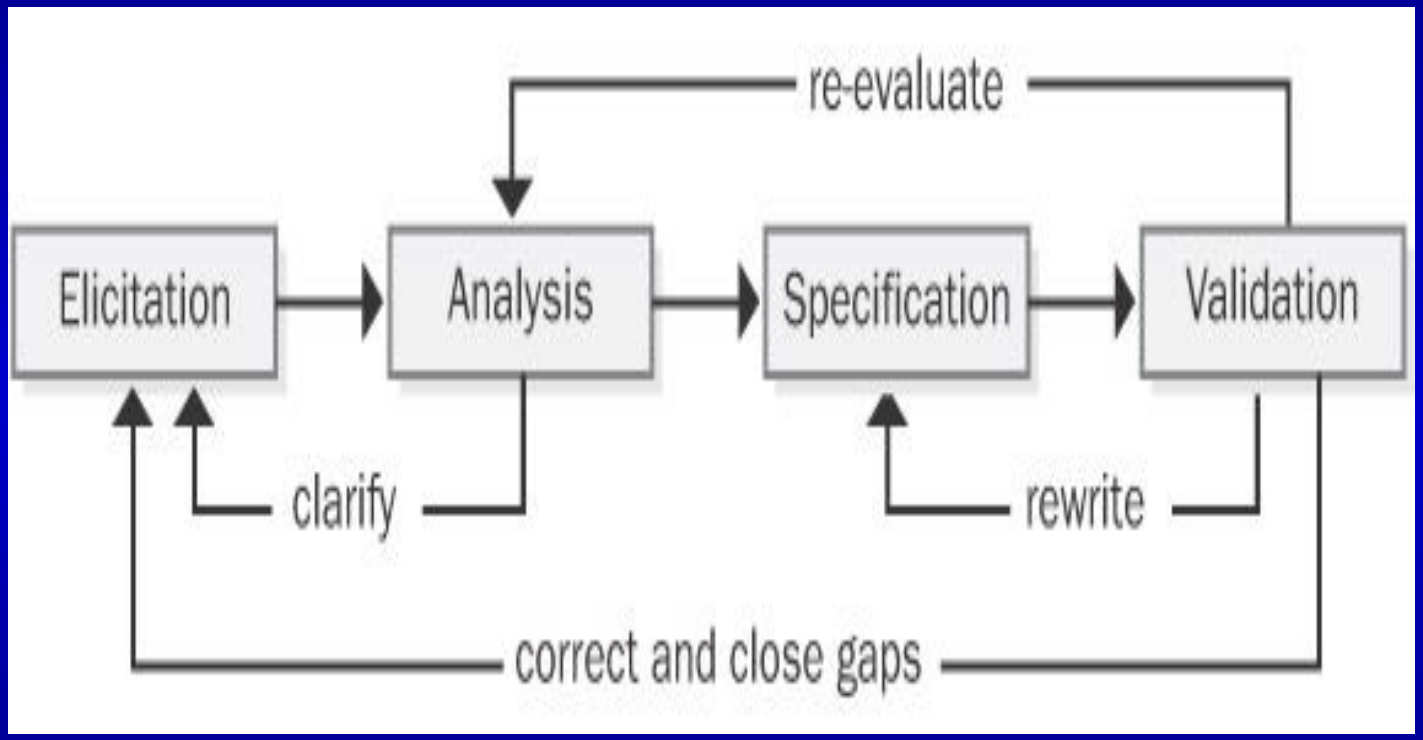
Ask for Business Rules



Data Dictionary

- A shared repository that defines
  - Name (naming space)
  - Meaning
  - Data type
  - Length
  - Format
  - Necessary precision
  - Allowed range or list of values
- Can be:
  - Primitive data elements
  - Composition
  - Iteration
  - Selection

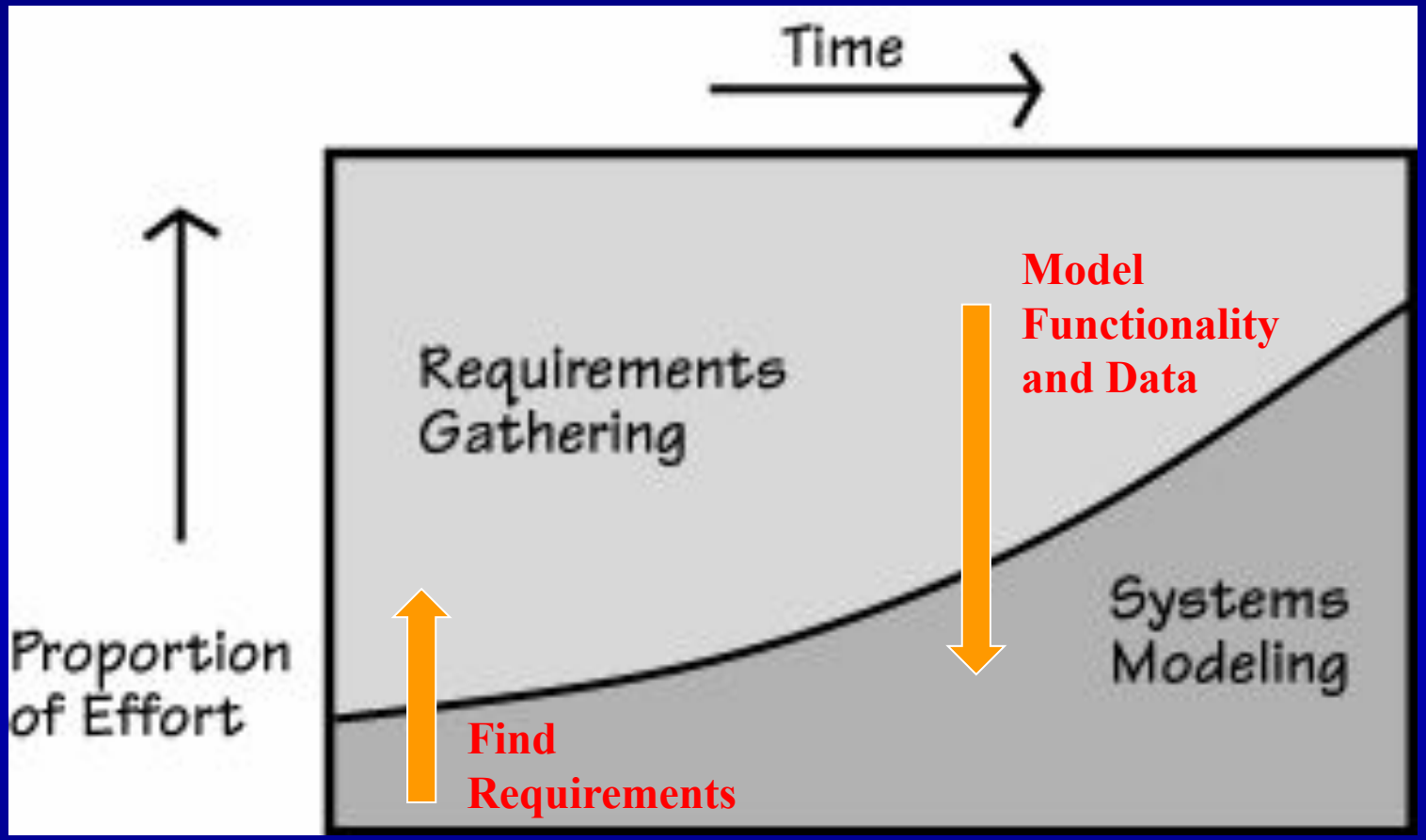
RD Process



分析確認需求

- 建立操作概念及劇本
- 建立功能定義
- 分析需求以確保必要與充分
- 分析需求以取得平衡 (處理衝突)
- 採用廣泛的方法確認需求

Requirements Gathering & System Modeling



Both produce artifacts used to understand and specify requirements.

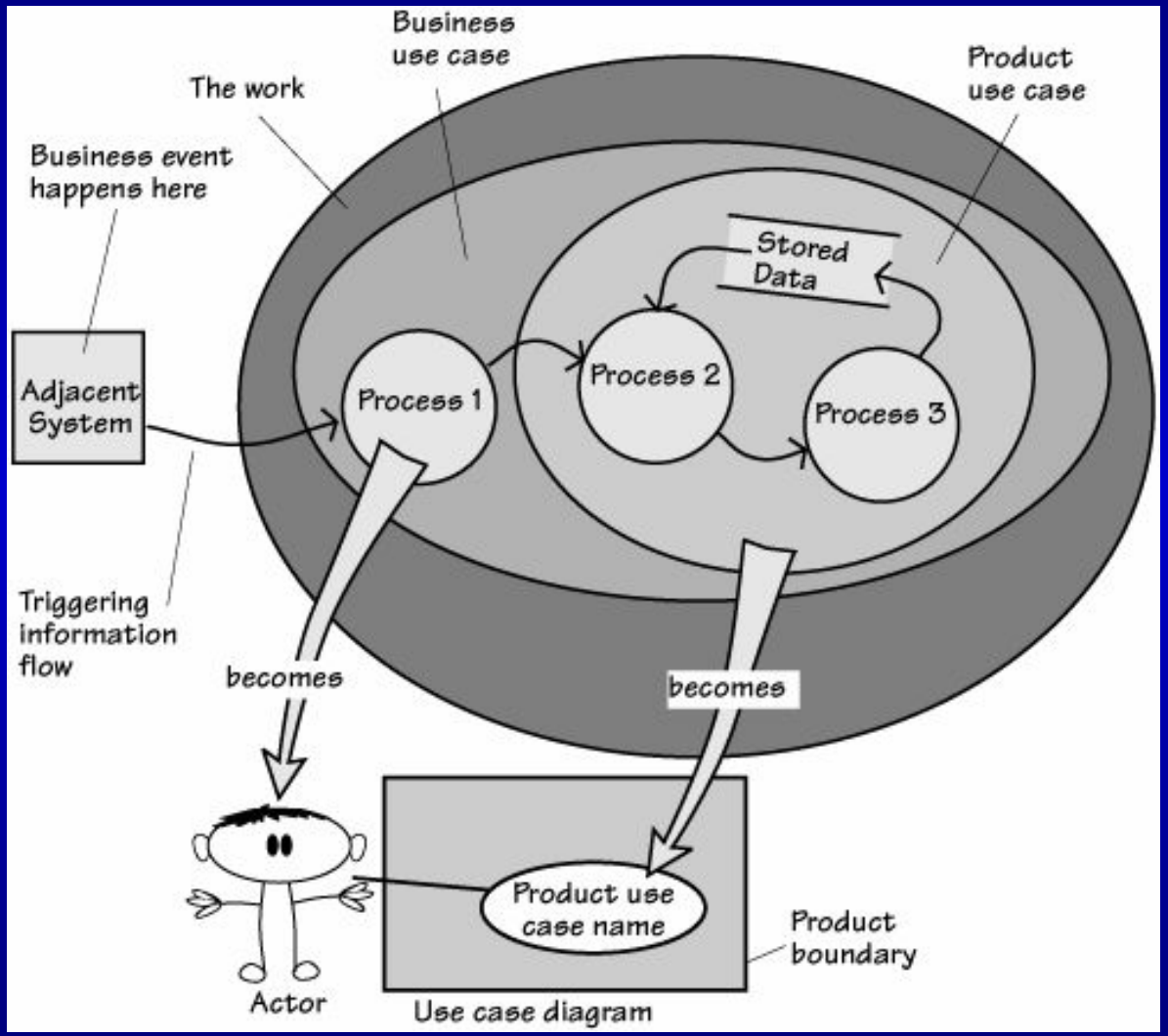
In the beginning

- The only Models been built:
  - Context Diagram
  - Exploratory Data Model
  - Stakeholders Map
- The RAs busy discovering:
  - Business Goals
  - Stakeholders
  - The Work (Business Domain)
  - Desired Outcome

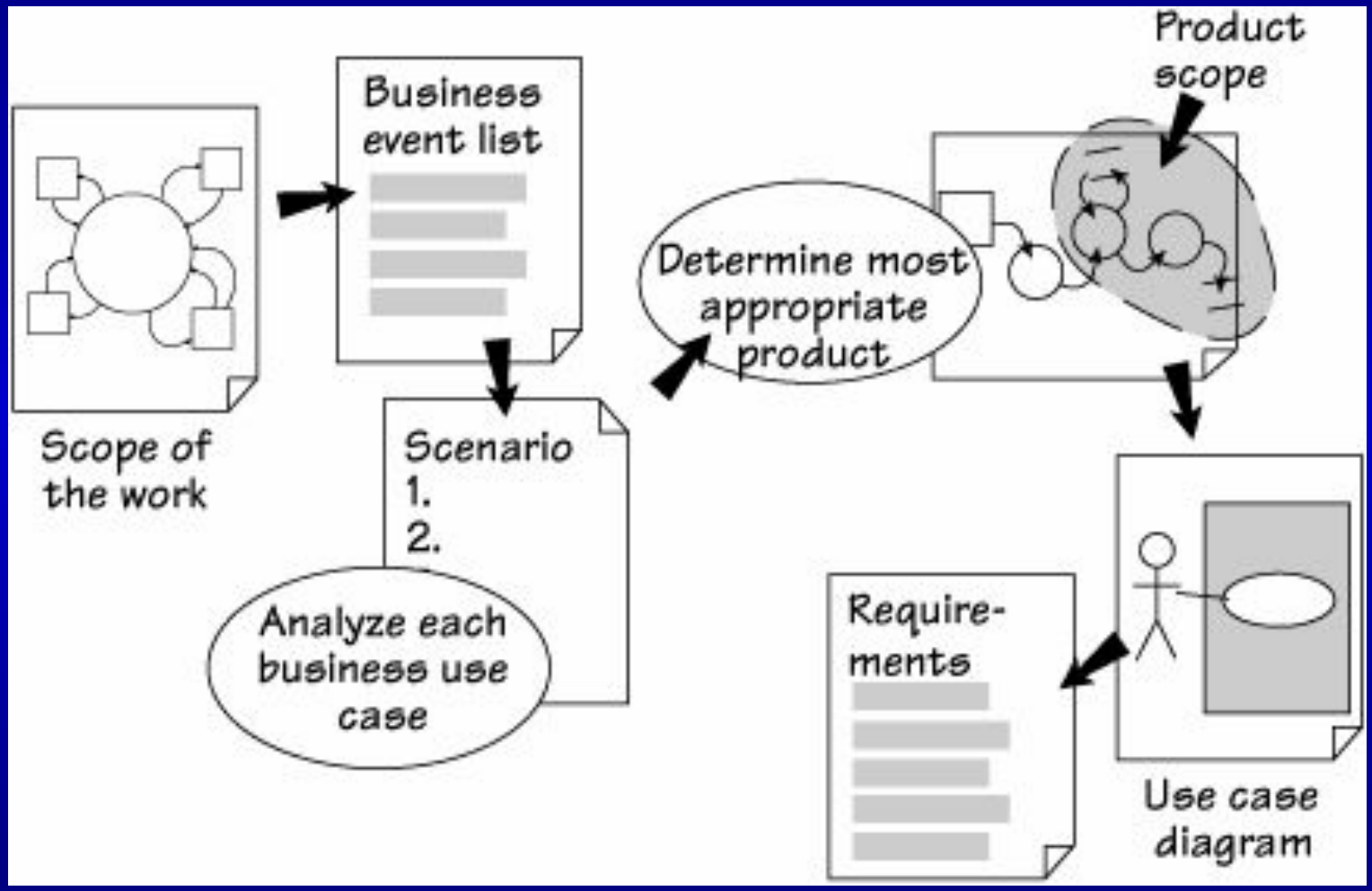
Models

- Helpful to specify the product, at least its functionality.
  - As the Model is Complete, and
  - Supported by a Data Dictionary, and
  - Process Descriptions or Scenarios.
- A suitable alternative to the textual specification.
- UseCase, DFD, ERD, 流程，操作手冊，安裝/維護/支援概念，雛型...

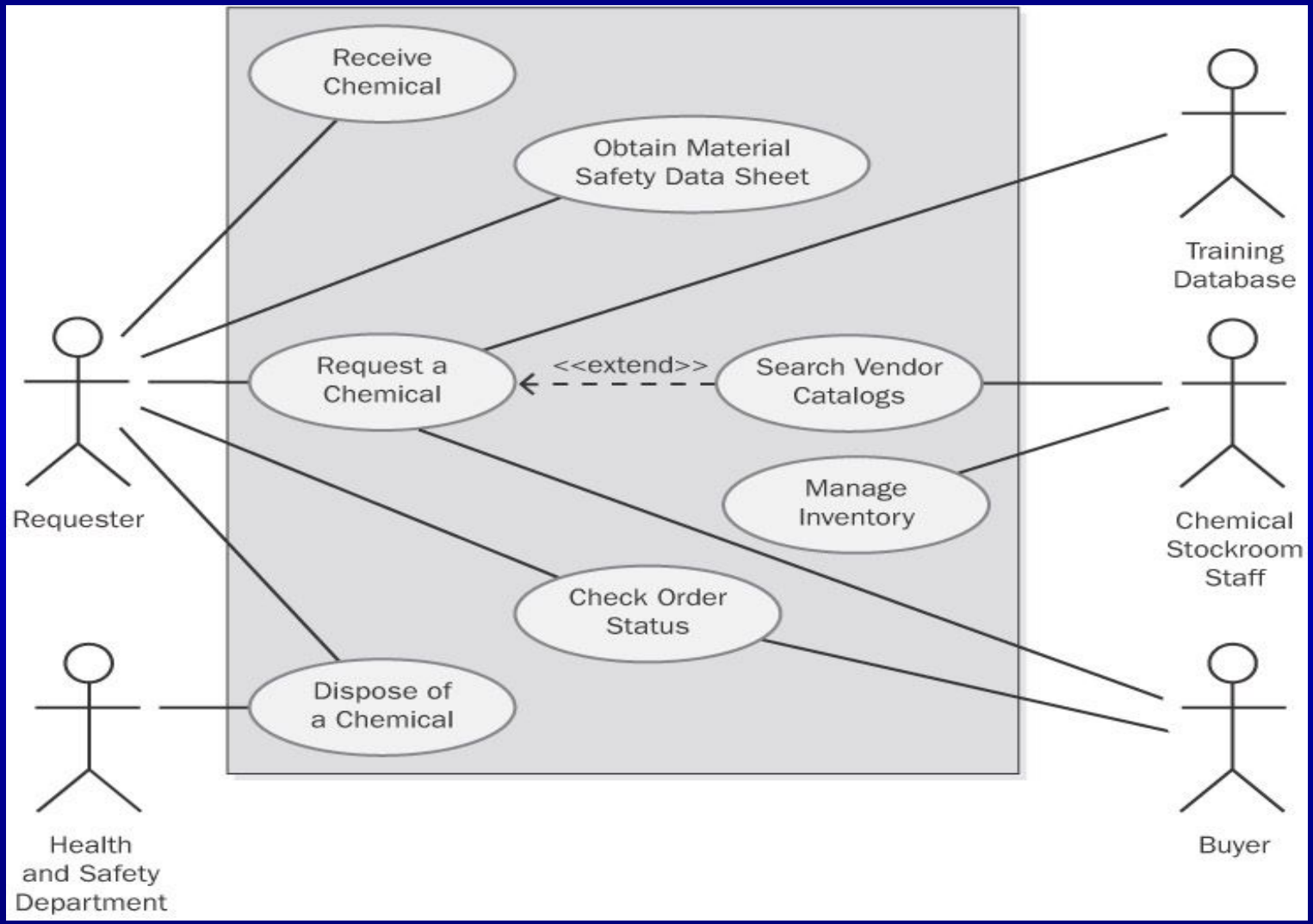
Business vs. Product Usecase



Business & Product Usecase



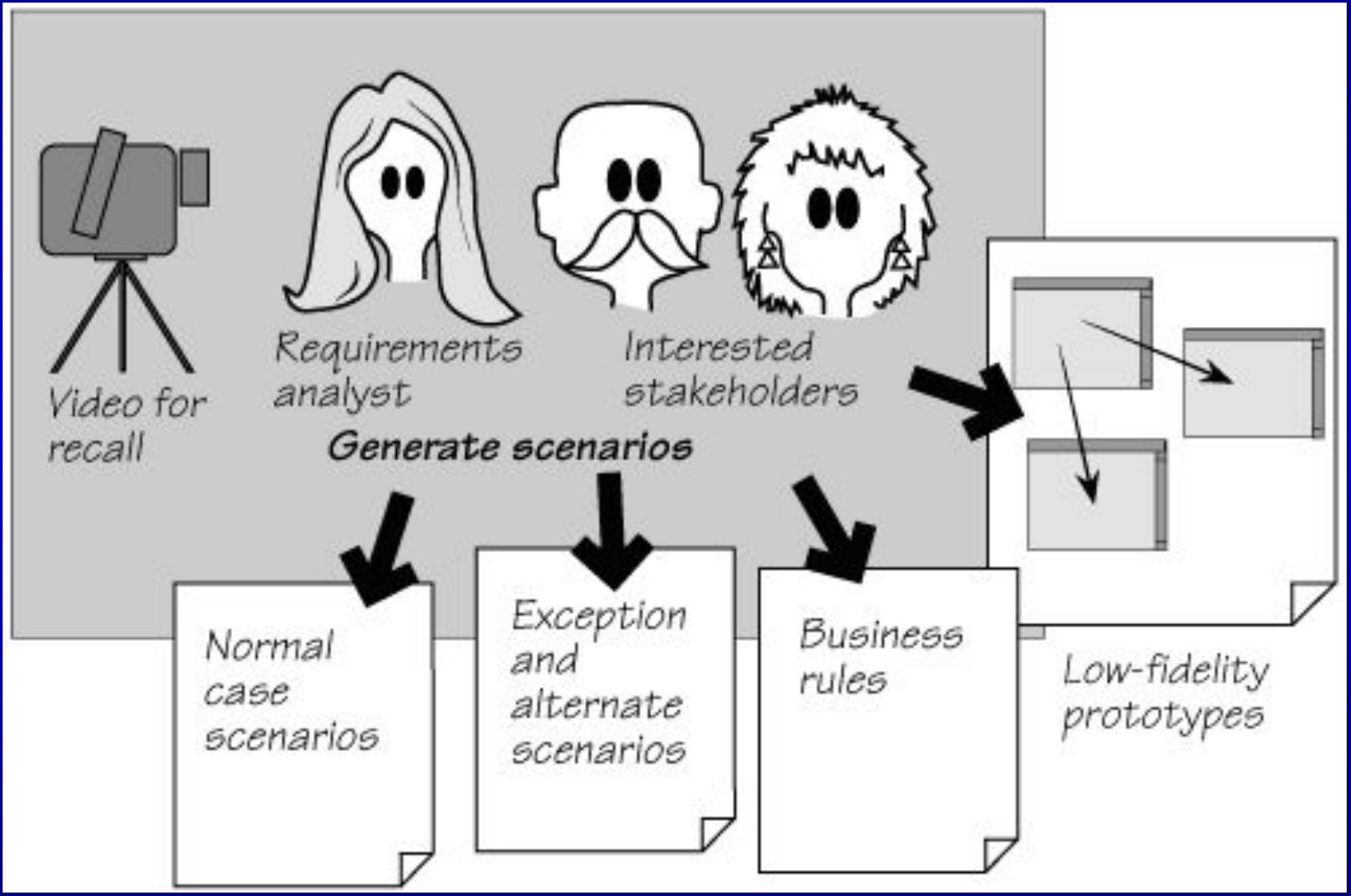
Use Cases Diagram



Pattern: BreadthBeforeDepth

- Problem:
  - You will not make timely progress or create coherent use case sets if you waste energy writing detailed use cases sequentially.
- Solution:
  - Conserve your energy by developing an overview of your use cases first, then progressively add detail, working across a group of related use cases.

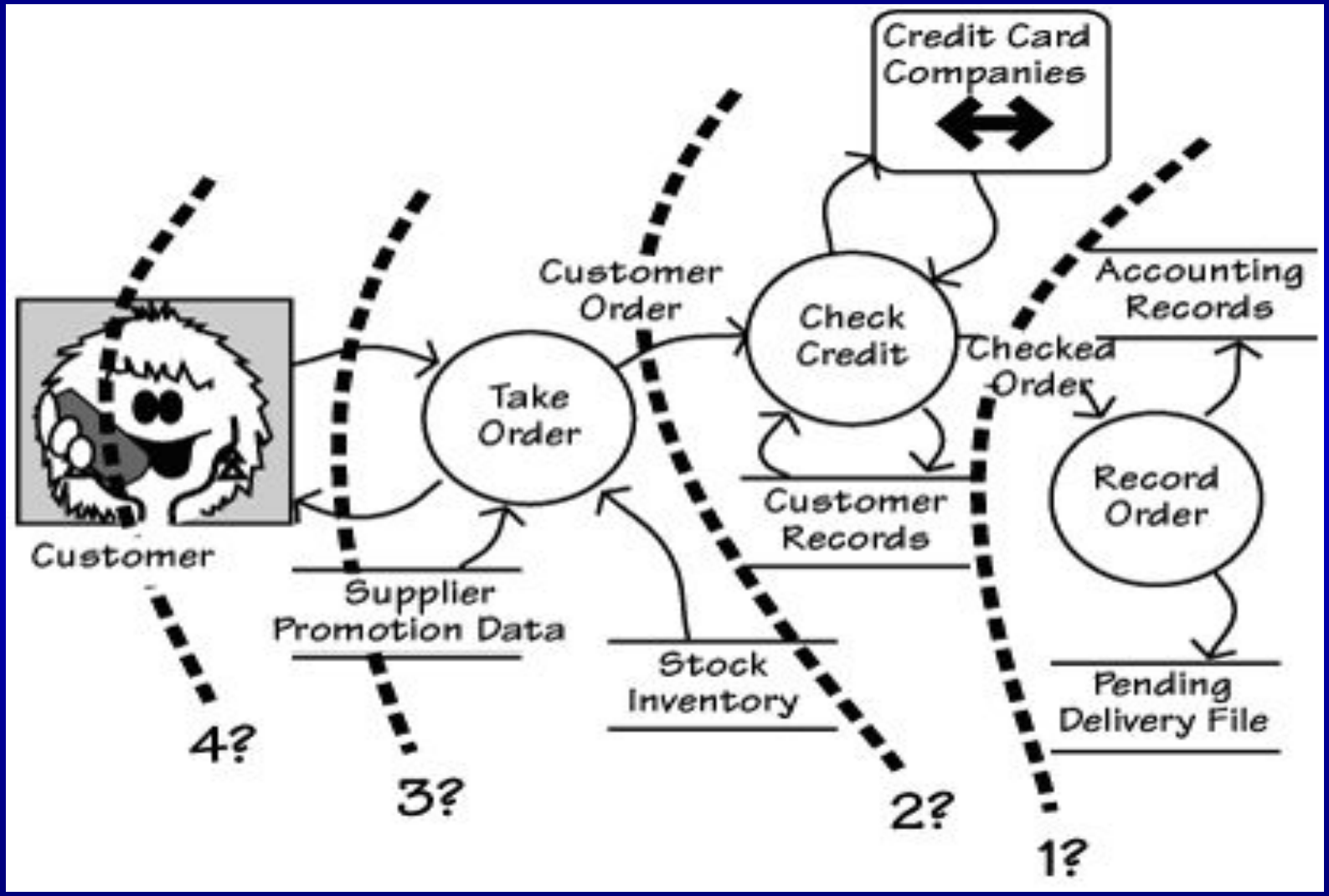
BreadthBeforeDepth



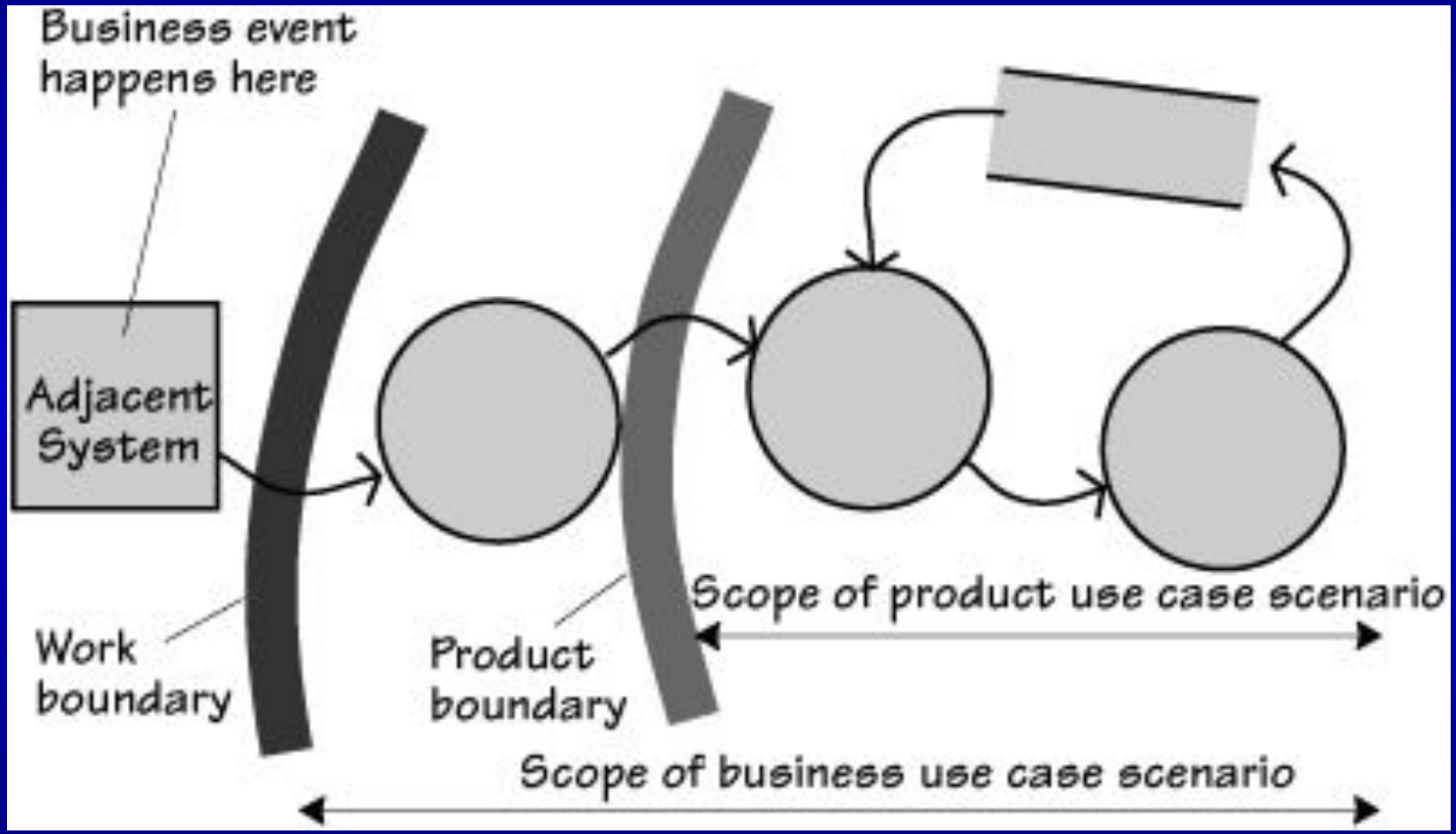
Pattern: SpiralDevelopment

- Problem:
  - Developing use cases in a single pass is difficult and can make it expensive to incorporate new information into them. Even worse, it can delay the discovery of risk factors.
- Solution:
  - Develop use cases in an iterative, breadth-first manner, with iteration progressively increasing the precision and accuracy of the use case set.

Layered Scope



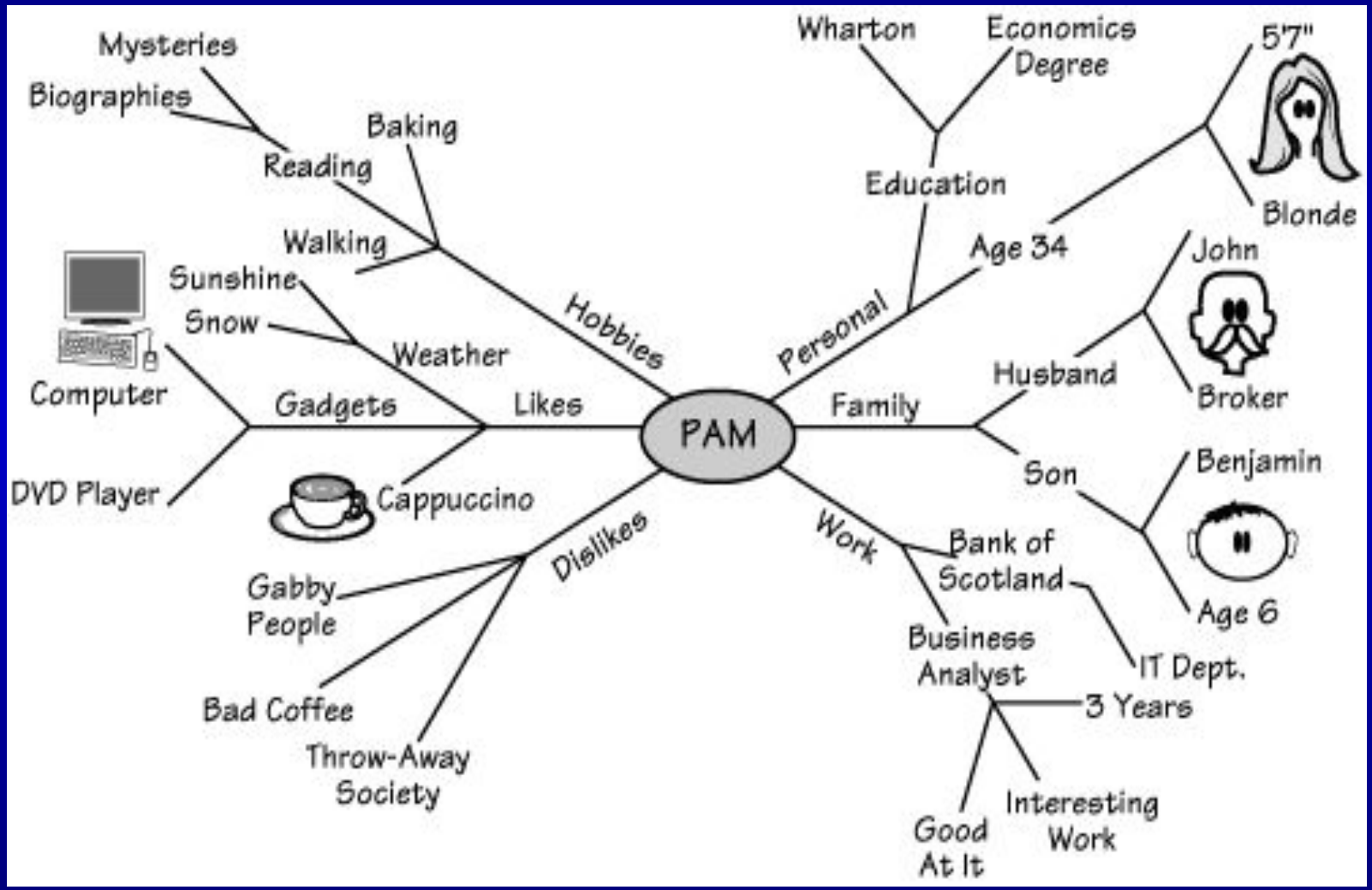
Product Scope



Pattern: MultipleForms

- Problem:
  - Different projects need different degrees of formality in their work, and various people have different preferences for the template. Requiring everyone to use the same use case template is counterproductive.
- Solution:
  - Select the format based on the risks associated with the project and the preferences of the people involved.

MindMap



Pattern: WritersLicense

- Problem:
  - Excessive emphasis on style issues unnecessarily impedes the effort of writing use cases.
- Solution:
  - Small differences in writing style are inevitable. Once a use case passes the tests for QuittingTime, the writer can claim "writer's license" on small stylistic differences.

Pattern: TwoTierReview

- Problem:
  - Many people may need to review the use cases. This is an expensive, time-consuming proposition.
- Solution:
  - Hold two types of reviews: 1st by a smaller, internal team, possibly repeated many times; 2nd by the complete group, perhaps just once.

Pattern: QuittingTime

- Problem:
  - Developing a use case model beyond the needs of the stakeholders and developers wastes resources and delays the project.
- Solution:
  - Stop developing use cases once they are complete and satisfactorily meet audience needs.

No Over-Specified

- MIND – Manual Impact Nail Device
  - The MIND shall have an impact face between 15 and 20 mm in diameter and must be made from dropforged tungsten steel.
  - The MIND shall also have an integrated NRU—Nail Recovery Unit—consisting of twin 30mm carbon steel prongs, set apart in a V formation with a maximum gap spacing of 5 mm.
  - Lastly, the MIND should have a human-machine interface of oak or equivalent hardwood…

Requirements Pitfalls

- Don’t add unnecessary requirements
  - Writing requirement is a difficult task
  - Increase complexity
  - Invest the time in improving/validating the necessary requirements
- Never assume the existence of external systems
  - Must be explicitly required by the client
- The main goal is increase profits…
  - Avoid complex and/or expensive solutions

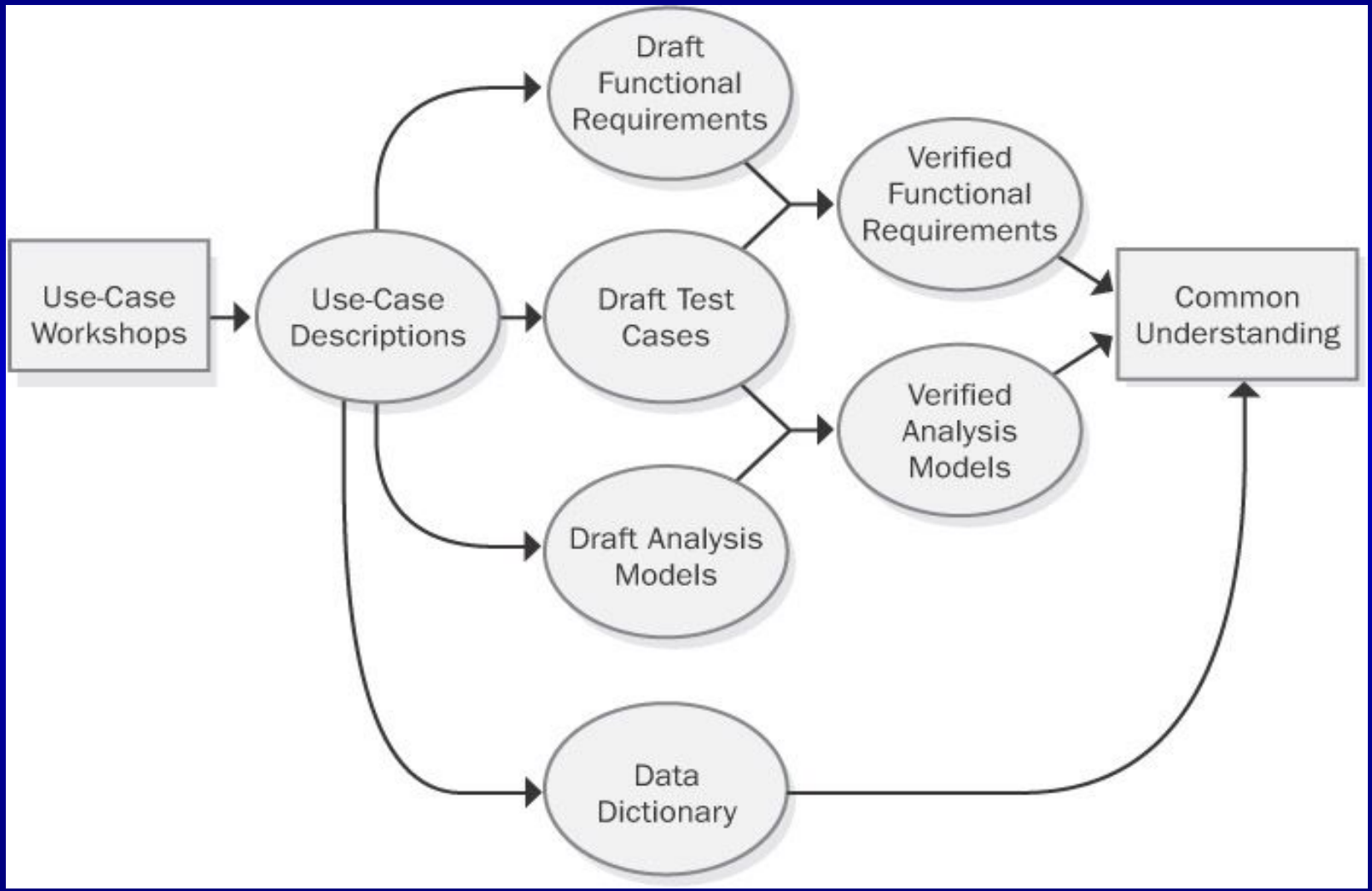
Pattern: UserValuedTransactions

- Problem:
  - A system is deficient if it cannot deliver services that are valuable to its uses and it does not support the goals and objectives specified by the system vision.
- Solution:
  - Identify the valuable services that the system delivers to the actors to satisfy their business purposes.

Pattern: EverUnfoldingStory

- Problem:
  - The number of steps needed to describe the behavior of the system exceeds both the memory and the interest of various types of readers.
- Solution:
  - Organize the use case set as a hierarchical story that can be either unfolded to get more detail or folded up to hide detail and show more context.

Use-case Elicitation Approach



Use Case Description

Use Case ID	**UC-1**	Use Case Name	**Request a Chemical**
Created By	Tim	Last Updated By	Janice
Date Created	12/4/02	Date Last Updated	12/27/02
Actors	Requester		
Description	The Requester specifies the desired chemical to request by entering its name or chemical ID number or by importing its structure from a chemical drawing tool. The system satisfies the request either by offering the Requester a new or used container of the chemical from the chemical stockroom or by letting the Requester create a request to order from an outside vendor.		
Preconditons	1. User's identity has been authenticated. 2. User is authorized to request chemicals. 3. Chemical inventory database is online.		
Postconditions	1. Request is stored in the Chemical Tracking System. 2. Request was e-mailed to the chemical stockroom or to a Buyer.		
Normal Course	**1.0 Request a Chemical from the Chemical Stockroom** 1. Requester specifies the desired chemical. 2. System verifies that the chemical is valid. 3. System lists containers of the desired chemical that are in the chemical stockroom. 4. Requester has the option to View Container History for any container. 5. Requester selects a specific container or asks to place a vendor order (alternative course 1.1). 6. Requester enters other information to complete the request. 7. System stores request and e-mails it to chemical stockroom.		
Alternative Courses	**1.1 Request a Chemical from a Vendor (branch after step 5)** 1. Requester searches vendor catalogs for a chemical. 2. System displays a list of vendors with available container sizes, grades, and prices. 3. Requester selects a vendor, container size, grade, and number of containers. 4. Requester enters other information to complete the request. 5. System stores request and e-mails it to Buyer.		

Use Case Description

Exceptions	**1.0.E.1 Chemical is not valid (at step 2)** 1. System displays message: "That chemical does not exist." 2. System asks Requester whether he wishes to request another chemical or to exit. 3a. Requester asks to request another chemical. 4a. System starts Normal Course over. 3b. Requester asks to exit. 4b. System terminates use case. **1.0.E.2 Chemical is not commercially available (at step 5)** 1. System displays message: "No vendors for that chemical." 2. System asks Requester whether he wishes to request another chemical or to exit. 3a. Requester asks to request another chemical. 4a. System starts Normal Course over. 3b. Requester asks to exit. 4b. System terminates use case.
Includes	**UC-22** View Container History
Priority	High
Frequency of Use	Approximately five times per week by each chemist, 100 times per week by each member of chemical stockroom staff.
Business Rules	**BR-28** Only staff who are authorized by their laboratory managers may request chemicals.
Special Requirements	1. The system must be able to import a chemical structure in the standard encoded form from any of the supported chemical drawing packages.
Assumptions	1. Imported chemical structures are assumed to be valid.
Notes and Issues	1. Tim will find out whether management approval is needed to request a chemical on the Level 1 hazard list. Due date 1/4/03.

Pattern: CompleteSingleGoal

- Problem:
  - Improper goals will leave the writes uncertain about where one use case ends and another begins.
- Solution:
  - Write each use case to address one complete and well-defined goal. That goal may be at any level in the EverUnfoldingSotry.

Pattern: VerbPhraseName

- Problem:
  - Meaningless generic names will not set reader expectations or provide a convenient reference point.
- Solution:
  - Name the use case with an active verb phrase that represents the goal of the primary actor.

Pattern: ScenarioPlusFragments

- Problem:
  - Readers must be able to follow the path easily through the specific scenario or story that they are interested in; otherwise, they are likely to become frustrated or miss important information.
- Solution:
  - Write the success story line as a simple scenario without any consideration for possible failures. Below it, place story fragments that show what alternatives may occur.

Pattern: ExhaustiveAlternatives

- Problem:
  - A use case may have many alternatives. Missing some alternatives means the developers will misunderstand the system's behavior, and the system will be deficient.
- Solution:
  - Capture all alternatives and failures that must be handled in the use case.

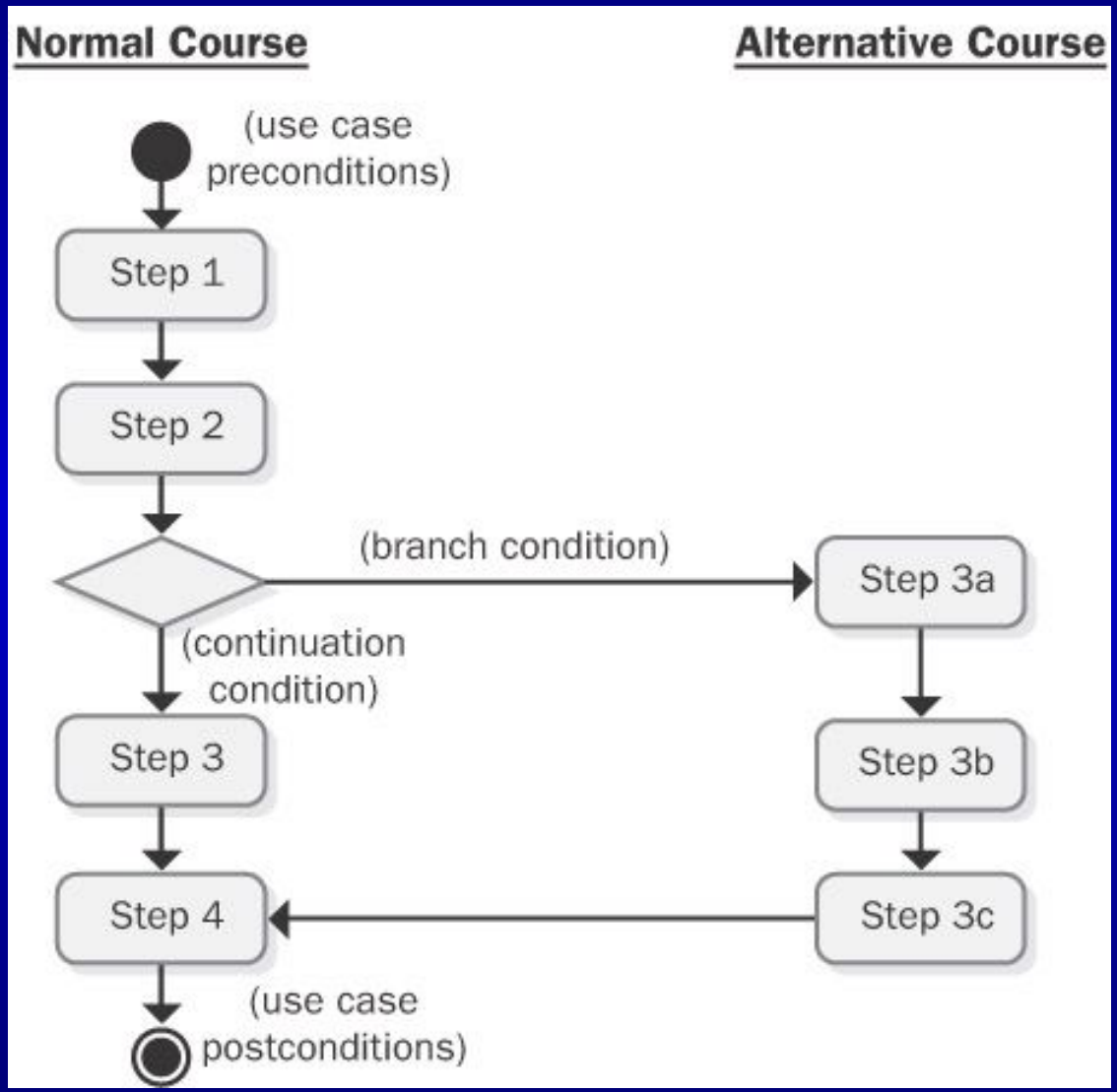
Pattern: Adornments

- Problem:
  - The inclusion of nonfunctional requirements in a use case can quickly clutter and obscure the use case.
- Solution:
  - Create additional fields in the use case template that are outside the scenario text to hold the supplementary information that is useful to associate with the use case.

Pattern: PreciseAndReadable

- Problem:
  - Use cases that are too complicated for non-technical readers, or too imprecise for developers, are deficient and likely to result in poorly built, inadequate systems.
- Solution:
  - Write the use case to be readable enough so that the stakeholders bother to read and evaluate it, and precise enough so that the developers understand what they are building.

Normal/Alternative Course



Pattern: DetectableConditions

- Problem:
  - Writers always wrestle with how many and which conditions to include.
- Solution:
  - Include only detectable conditions. Merge conditions that have the same net effect on the system.

Pattern: LeveledSteps

- Problem:
  - Excessively large or excessively small use case steps obscure the goal and make the use case difficult to read and comprehend.
- Solution:
  - Keep scenarios to three to nine steps. Ideally, the steps are all at similar levels, and at a level of abstraction just below the use case goal.

Pattern: ActorIntentAccomplished

- Problem:
  - Both readers and developers get confused about a system's behavior if it is not clear which actor has responsibility for performing a step, and what the actor is trying to accomplish in that step.
- Solution:
  - Write each step to show clearly which actor is performing the action, and what the actor gets accomplished.

Pattern: ForwardProgress

- Problem:
  - Writes have to decide how much behavior to put into any one step. They can easily write too much detail, making the use case long and tiring to read.
- Solution:
  - Eliminate or merge steps that do not advance the actor. Simplify passages that distract the reader from this progress.

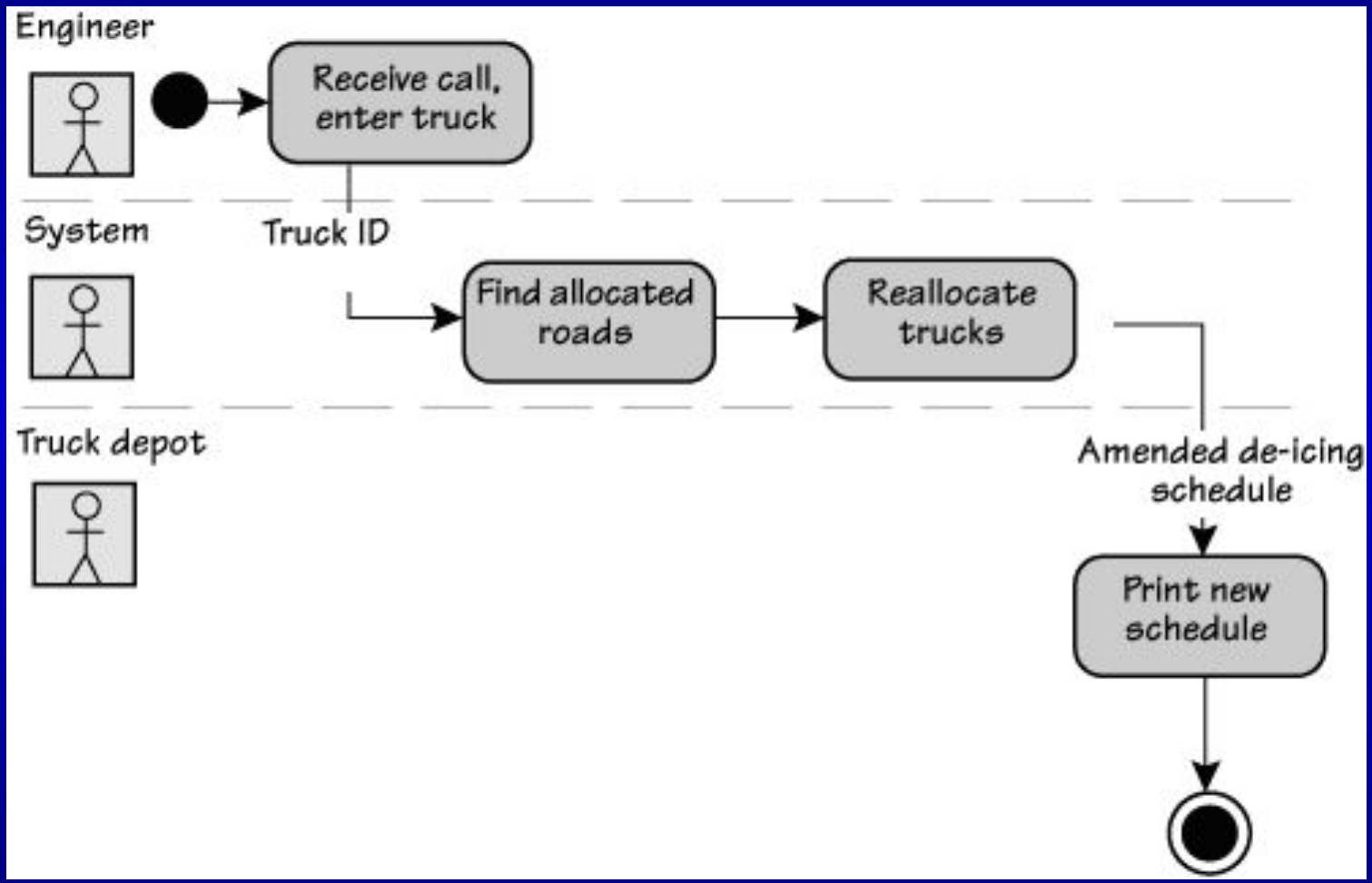
Pattern: TechnologyNeutral

- Problem:
  - Including technology constraints and implementation details in a use case description increases the complexity and obscures the goal of the use case.
- Solution:
  - Write each use case in a technology-neutral manner.

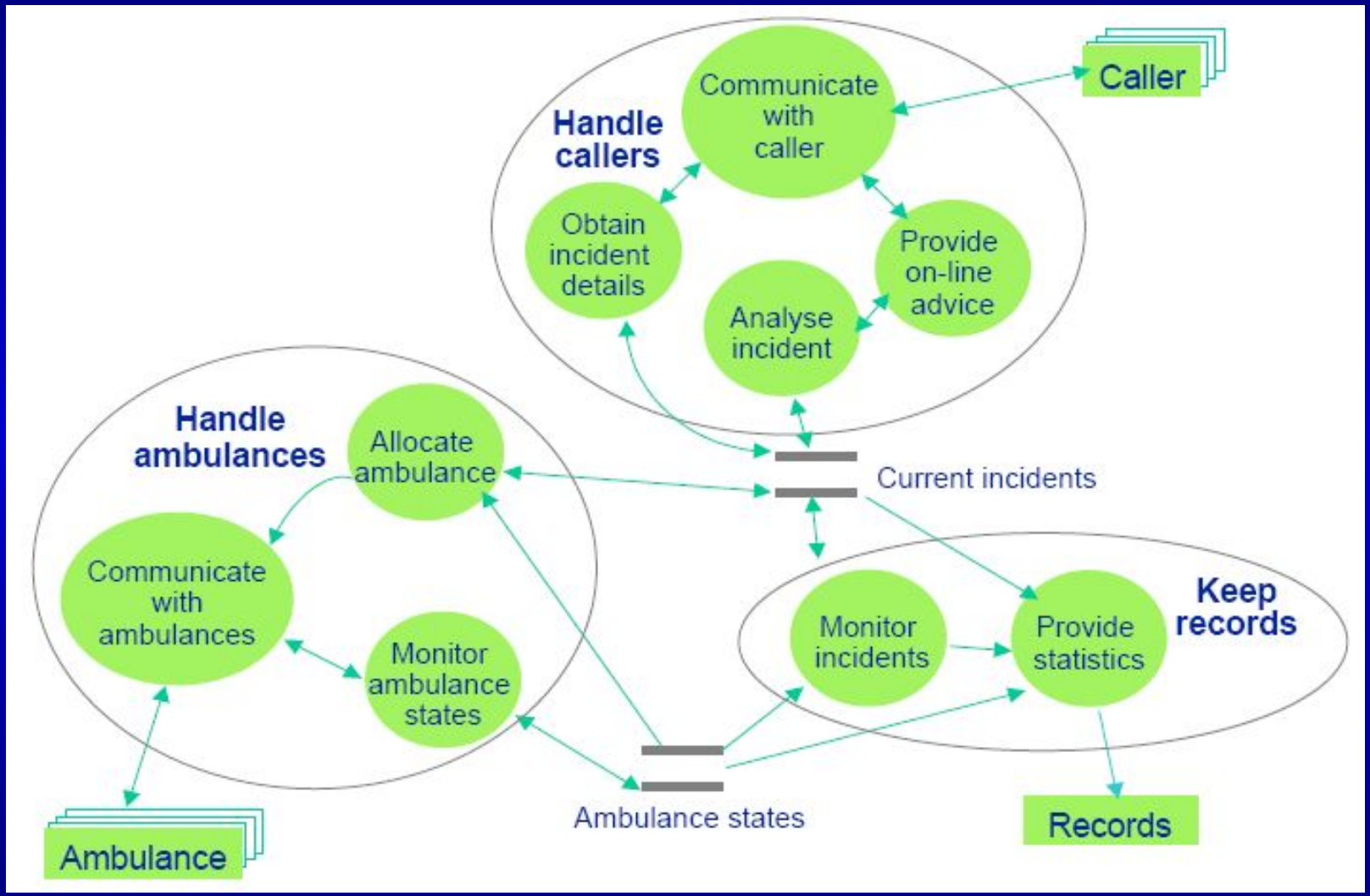
Alternatives to Use Cases

- Data Flow Model
  - Swim Lanes
  - Data Flow Diagram
  - Structured Analysis and Design
  - Functional Decomposition

Swim Lanes



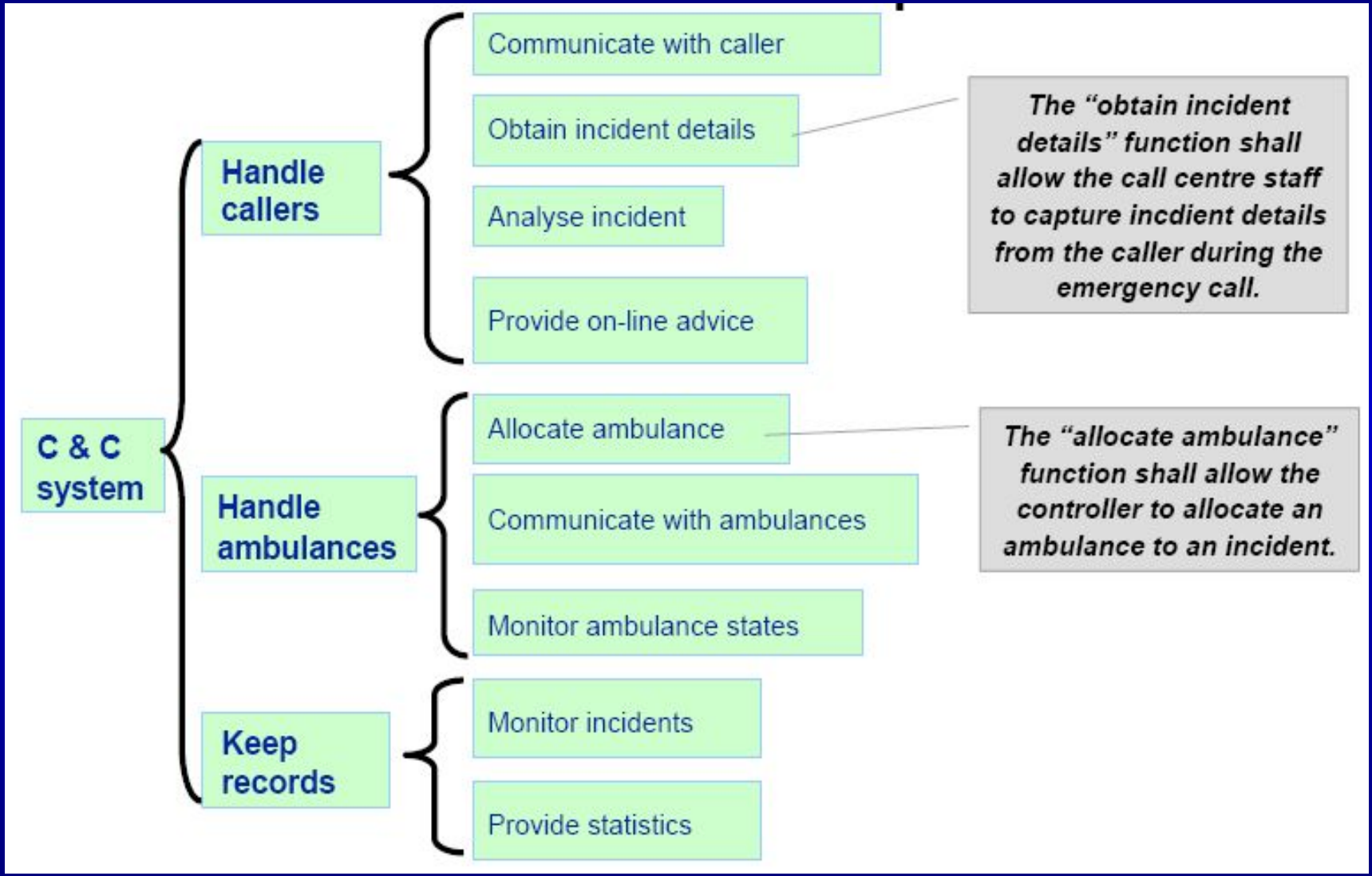
DFD (Data Flow Diagram)



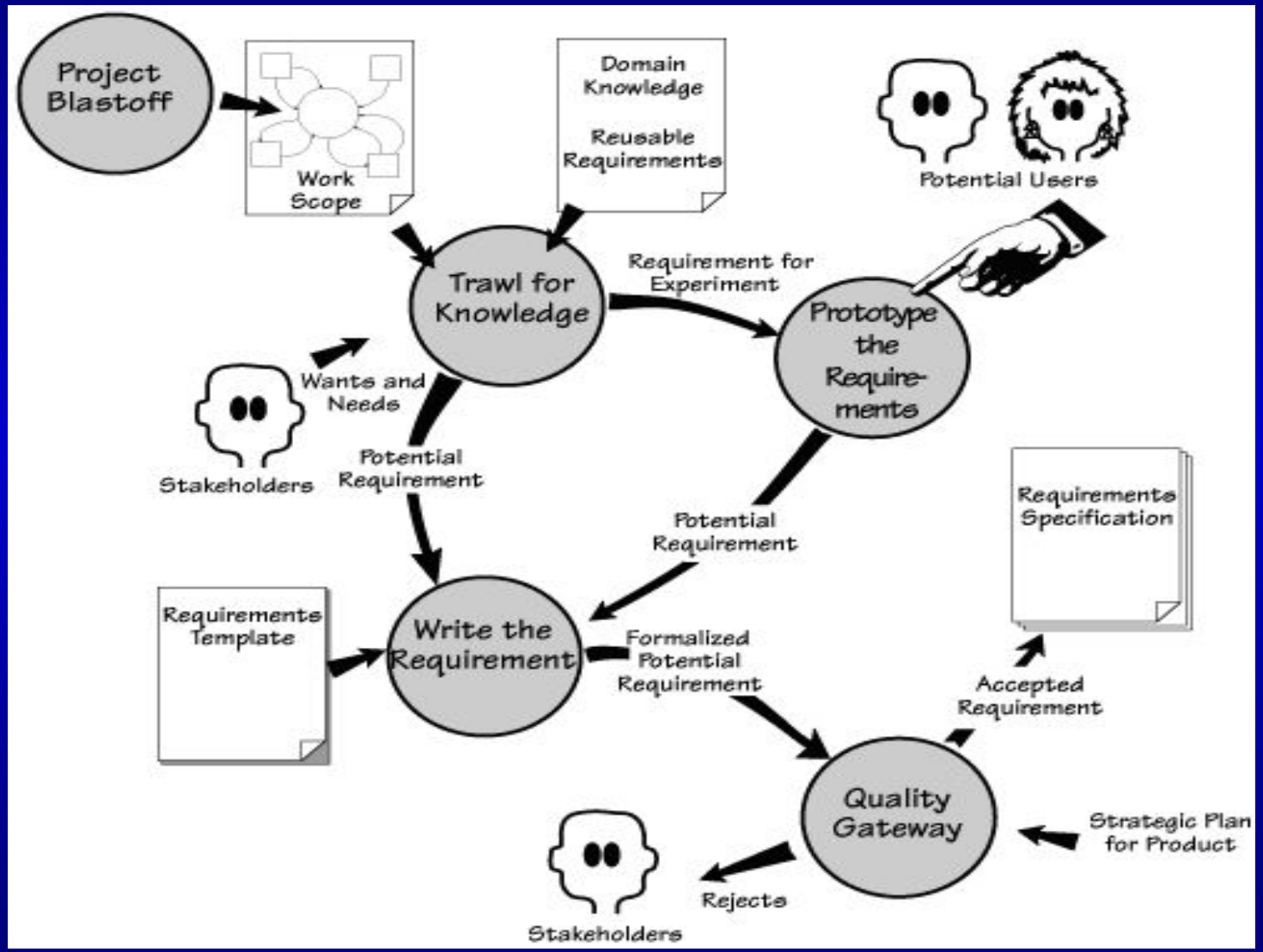
Structured Analysis & Design

- Establish the ***type*** of information flow
  - Transform Flow
  - Transaction Flow
- Indicate the flow ***boundaries***
- Map the DFD into ***program structure***
  - Transform: Incoming, Transform, Outgoing
  - Transaction: Reception, Dispatcher
- Define the control hierarchy by ***factoring***
- Refine the resultant structure using ***design measures and heuristics***

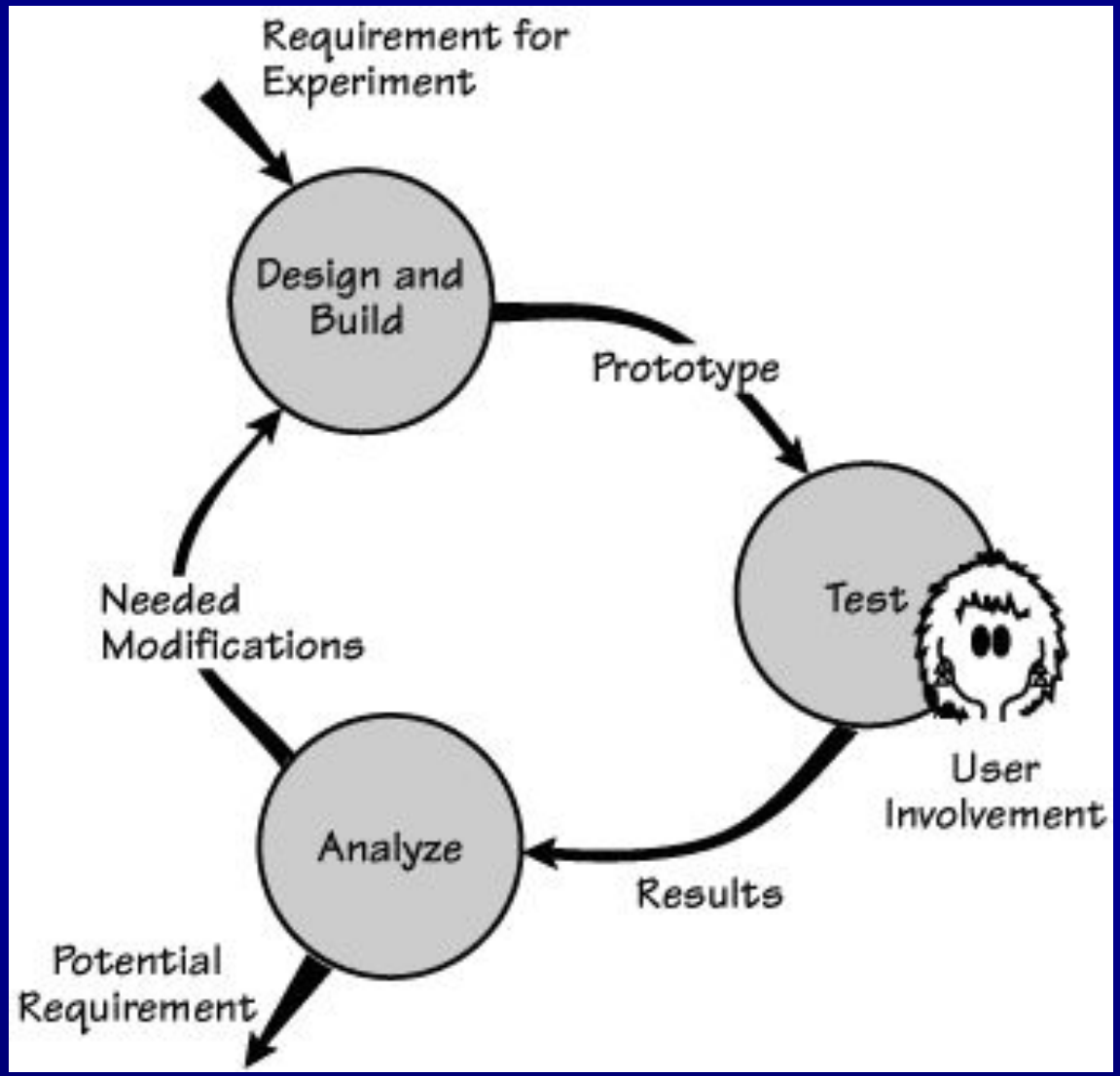
Functional Decomposition



Prototyping



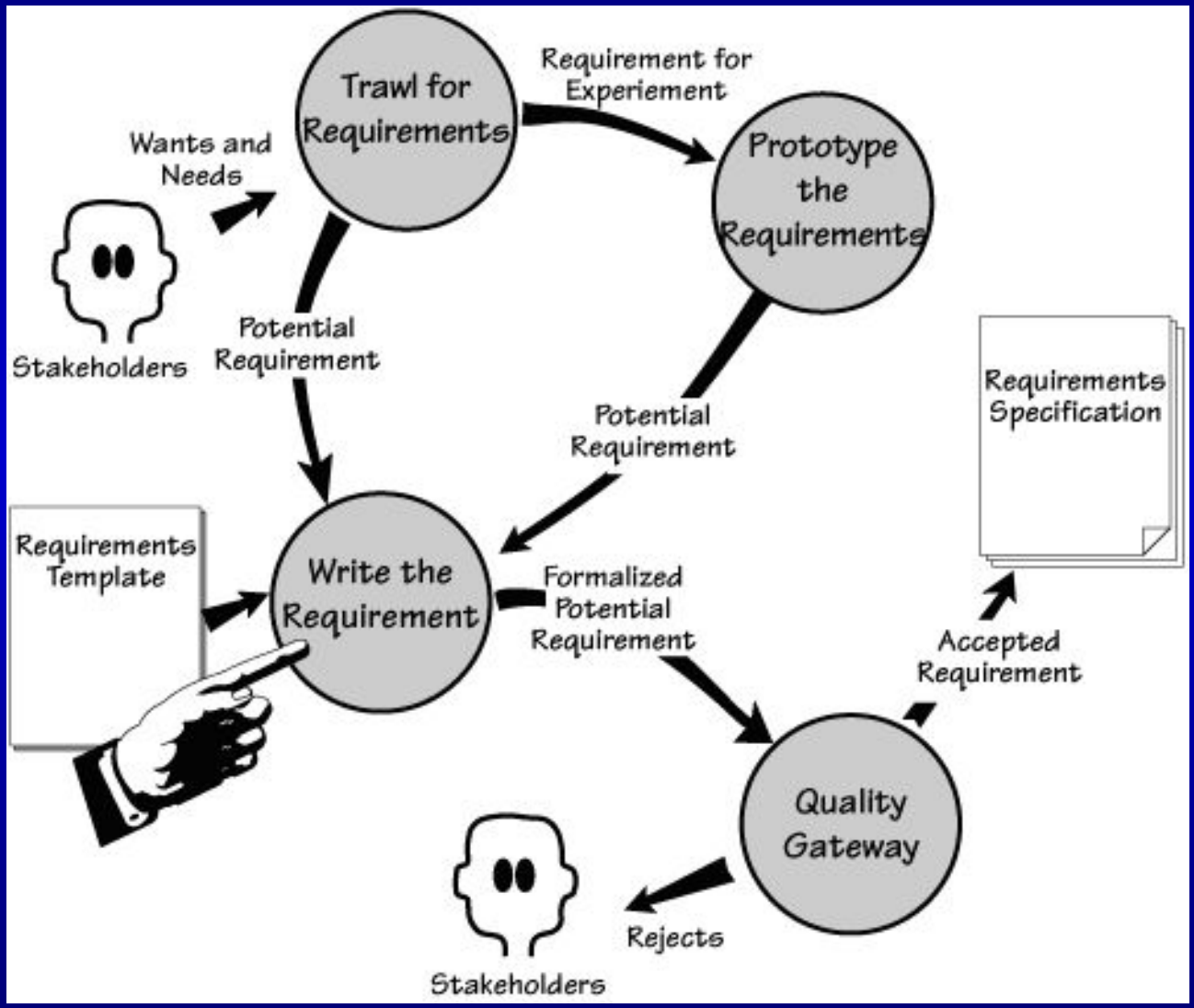
Prototyping – Reduce Risks



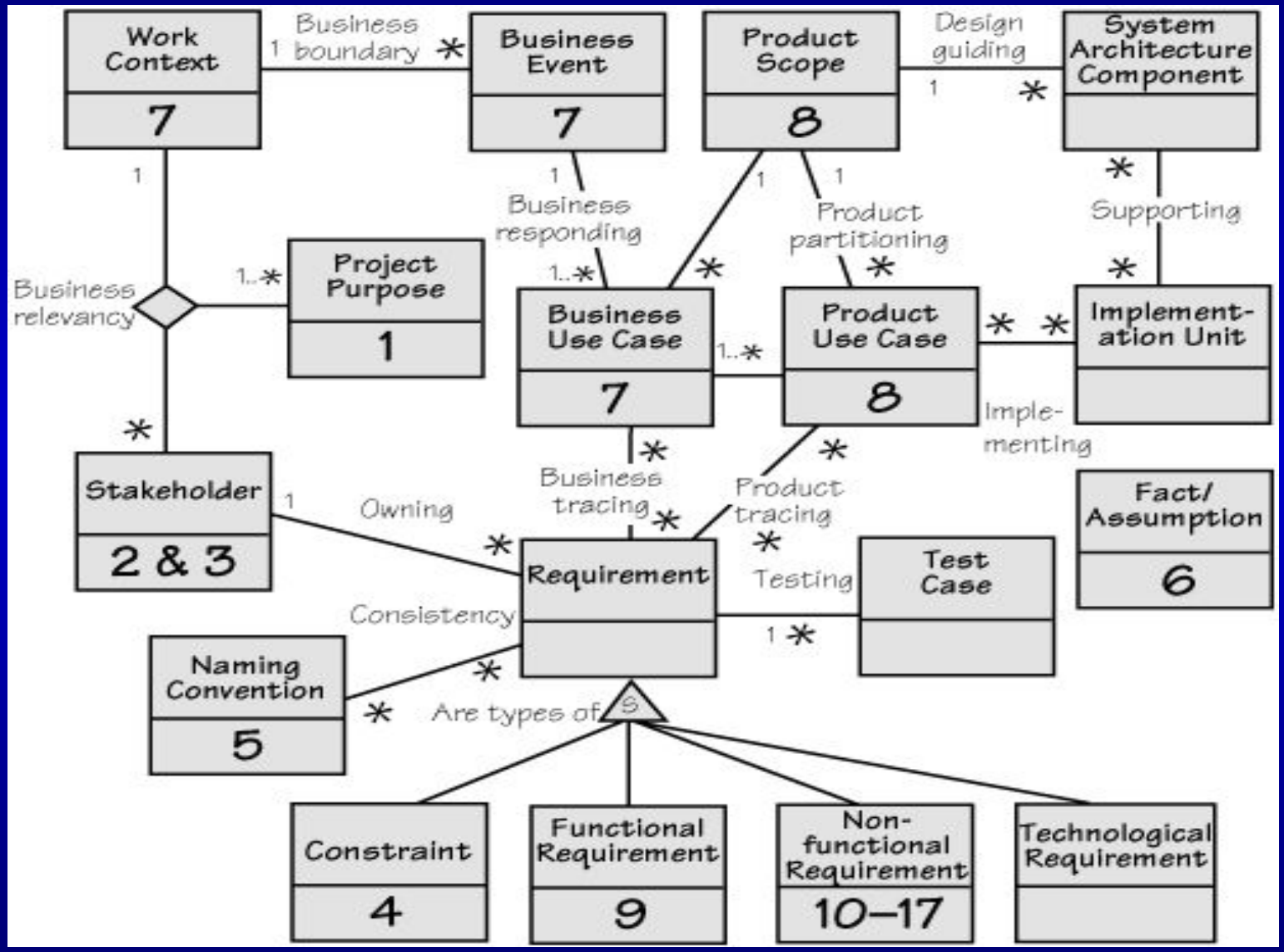
Prototyping: What & Why

- Clarify and complete the requirements
- Explore design alternatives
- Grow into the ultimate product
- Allocate Requirements

Specify Requirements



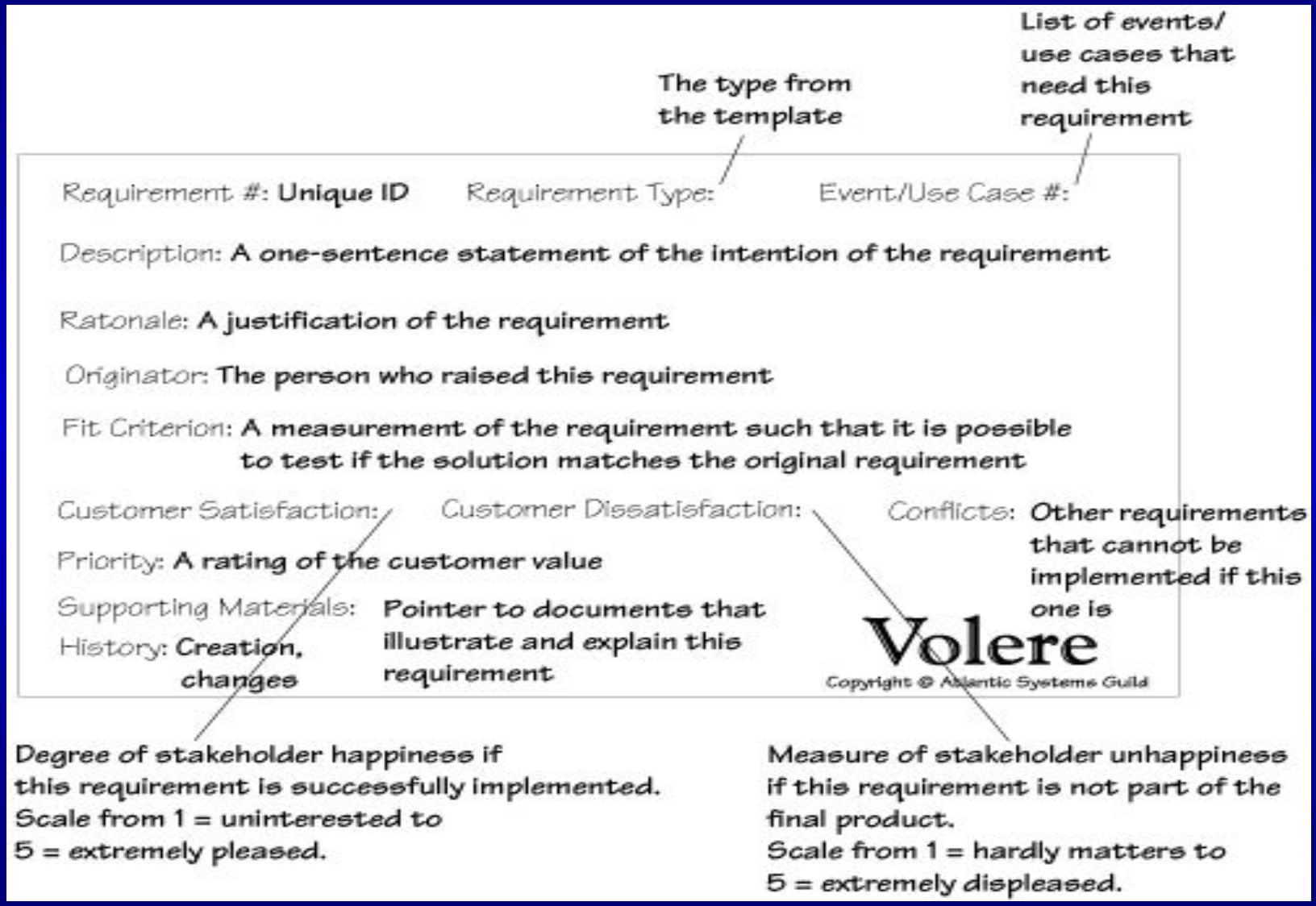
Requirements Knowledge Model



Requirement Specification

- Functional Requirements allocated to Product Use Cases
- Level of Detail or Granularity
  - The product shall…
- Exceptions and Alternatives
  - If … , the product shall
- Technological Requirements (內部only)
- Requirements, Not Solutions

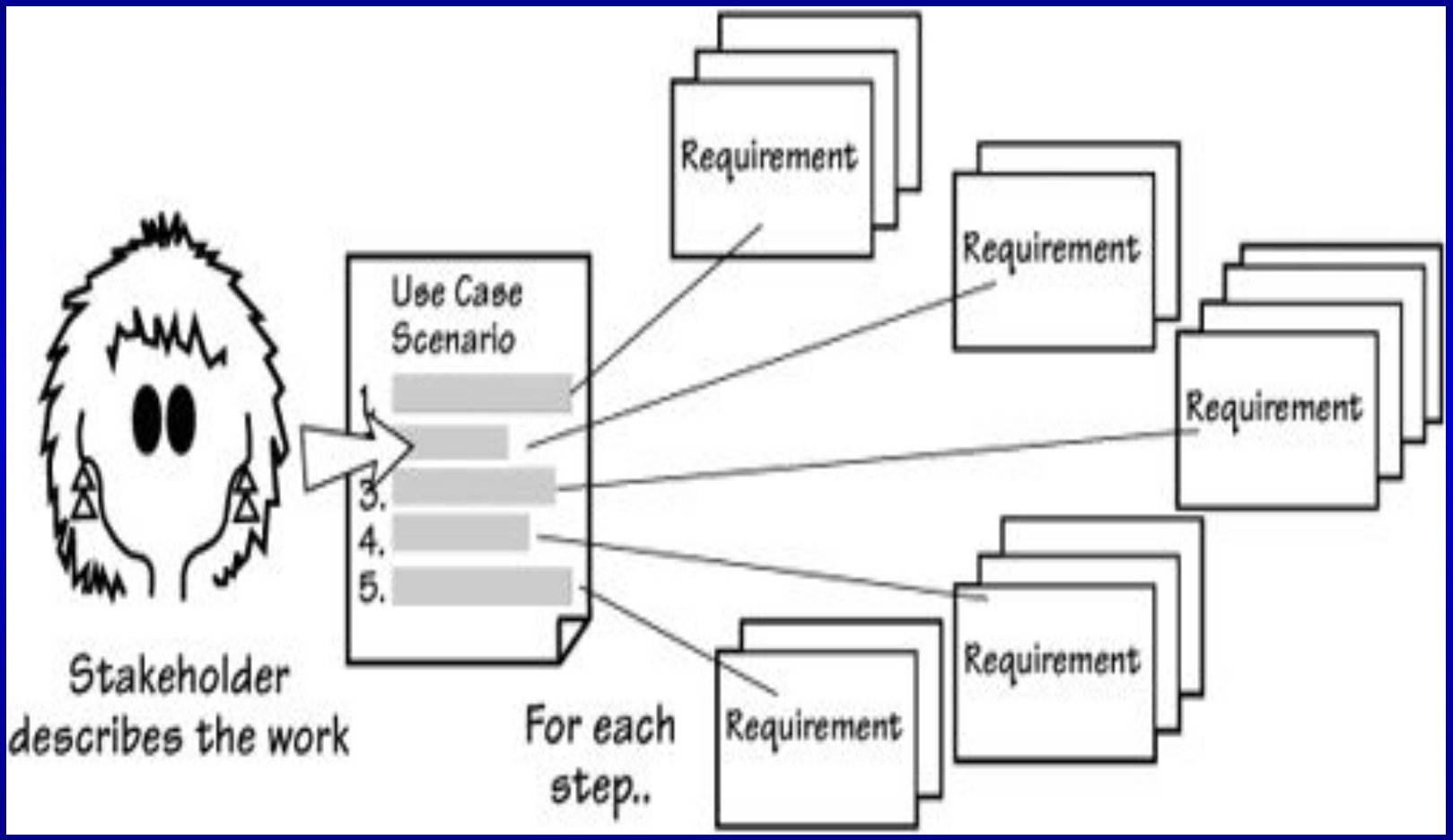
Requirement Shell



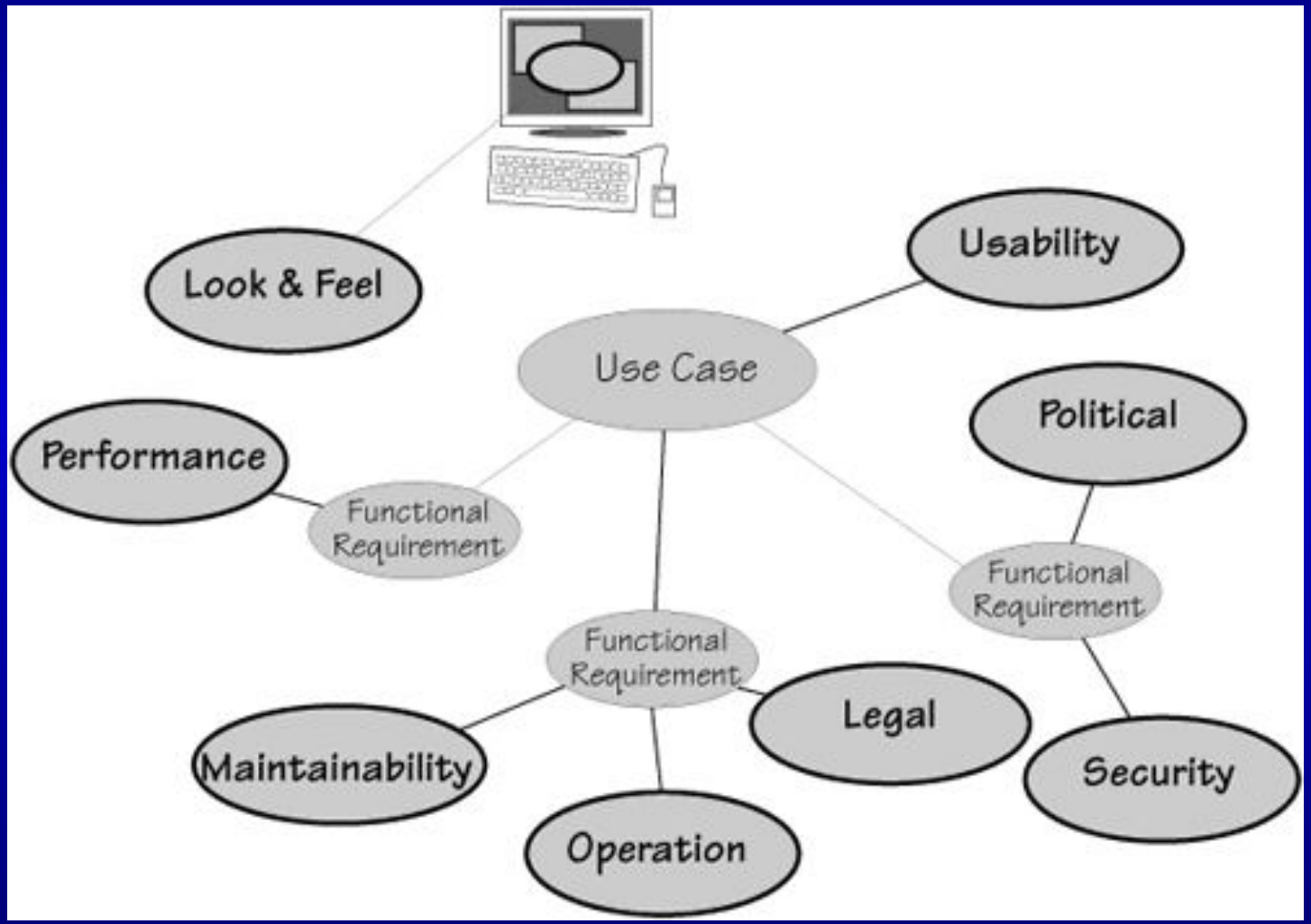
Other Requirement Attributes

- Author who wrote the requirement
- Person who is responsible for ensuring that the requirement is satisfied
- Owner of the requirement or a list of stakeholders (to make decisions about proposed changes)
- Requirement status
- Stability / Volatility
- Traceability Links

Deriving SRS



Nonfunctional Requirements in Use Cases



IEEE 830 SRS standard

1. **Introduction**
   - 1.1 Purpose
   - 1.2 Document Conventions
   - 1.3 Intended Audience and Reading Suggestions
   - 1.4 Project Scope
   - 1.5 References
2. **Overall Description**
   - 2.1 Product Perspective
   - 2.2 Product Features
   - 2.3 User Classes and Characteristics
   - 2.4 Operating Environment
   - 2.5 Design and Implementation Constraints
   - 2.6 User Documentation
   - 2.7 Assumptions and Dependencies
3. **System Features**
   - 3.x System Feature X
     - 3.x.1 Description and Priority
     - 3.x.2 Stimulus/Response Sequences
     - 3.x.3 Functional Requirements
4. **External Interface Requirements**
   - 4.1 User Interfaces
   - 4.2 Hardware Interfaces
   - 4.3 Software Interfaces
   - 4.4 Communications Interfaces
5. **Other Nonfunctional Requirements**
   - 5.1 Performance Requirements
   - 5.2 Safety Requirements
   - 5.3 Security Requirements
   - 5.4 Software Quality Attributes
6. **Other Requirements**

Appendix A: Glossary

Appendix B: Analysis Models

Appendix C: Issues List

Requirement Statement Criteria

- Individual: is a single traceable element
- Unique: is uniquely identified
- Clear: is clearly understandable
- Precise: is precise and concise
- Abstract: does not impose a solution
- Quantified: has acceptance criteria
- Prioritized: importance & urgency
- Testable: can be validated/verified
- Traceable: trace to other layers and tests

Requirement Specification Criteria

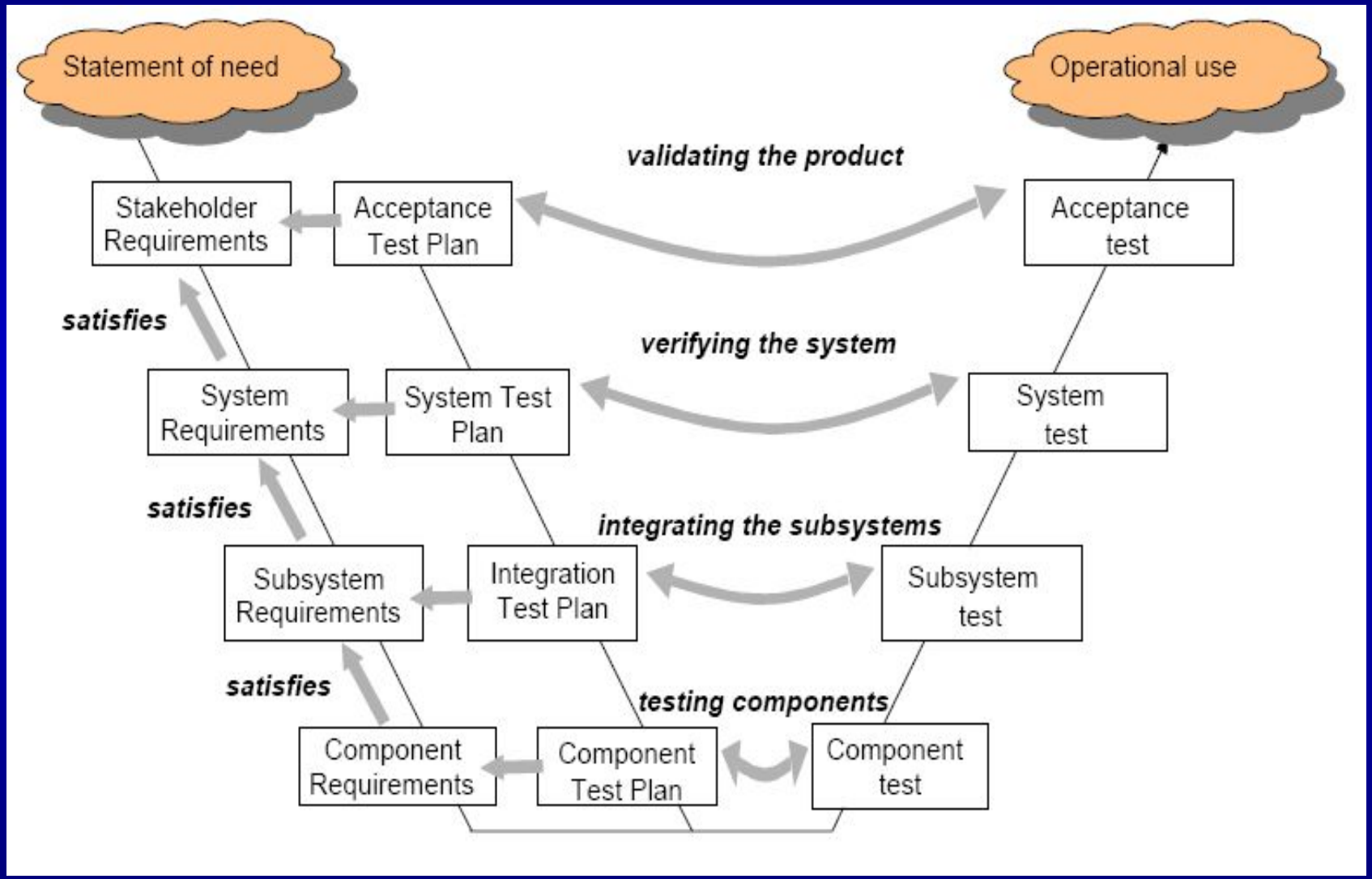
- Complete/Sufficient: all needed requirements present
- Consistent: no 2 requirements in conflict
- Non-redundant: each express only once
- Modular: cohesively grouped
- Structured: clear structure
- Modifiable: easy to modify
- Satisfied: design traceability
- Evaluated: test traceability

維護需求追溯

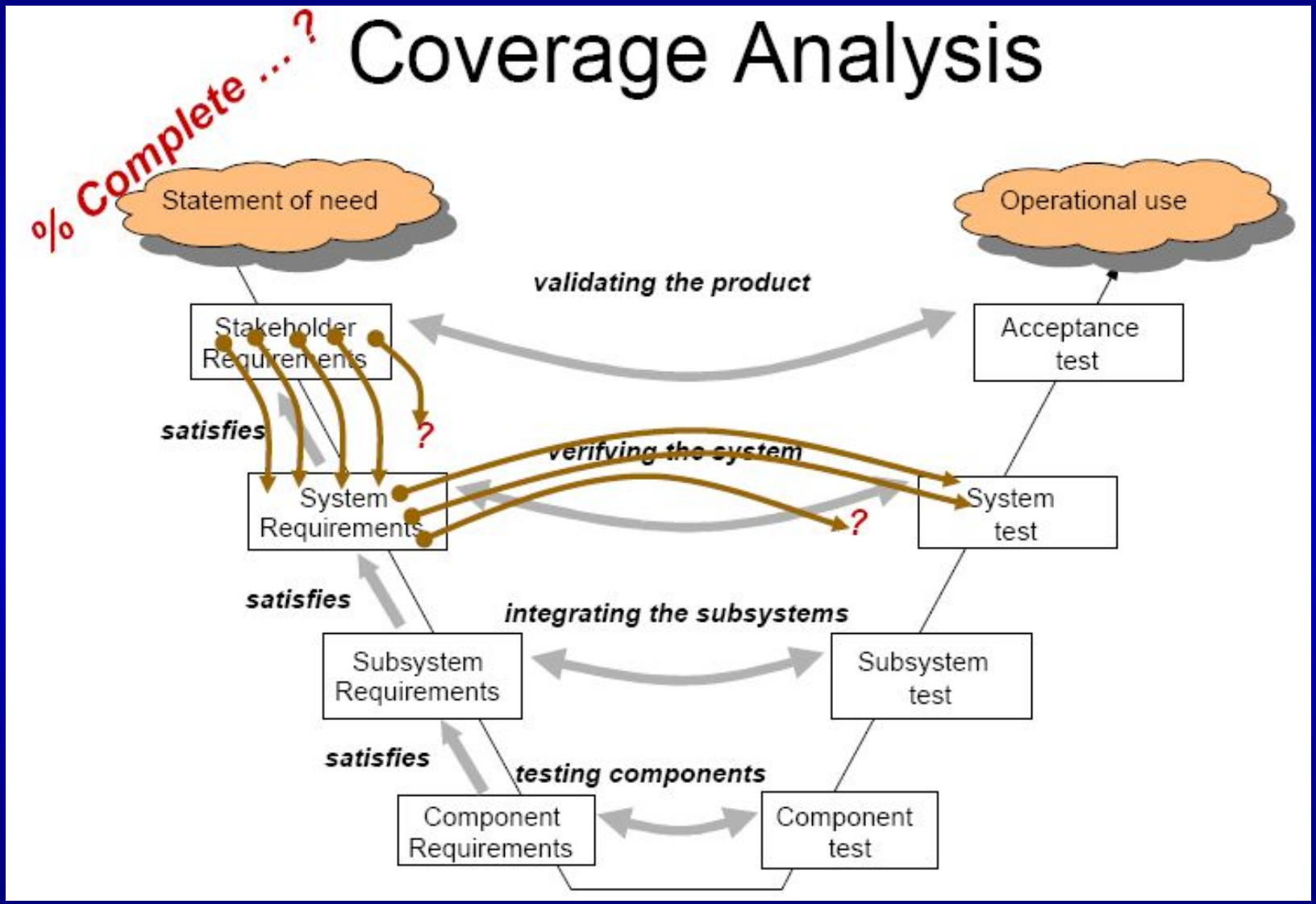
H14M51-需求追溯表

- 往前(來源)追溯：所依據的客戶需求(合約、訪談記錄、客戶使用案例等)，也可能是後續TS階段導出的衍生需求規格
- 往後(工作產品)追溯：後續的產品單元如設計規格、測試案例皆標記其對應的需求規格
- 水平關聯：需求與需求之間的相依關係

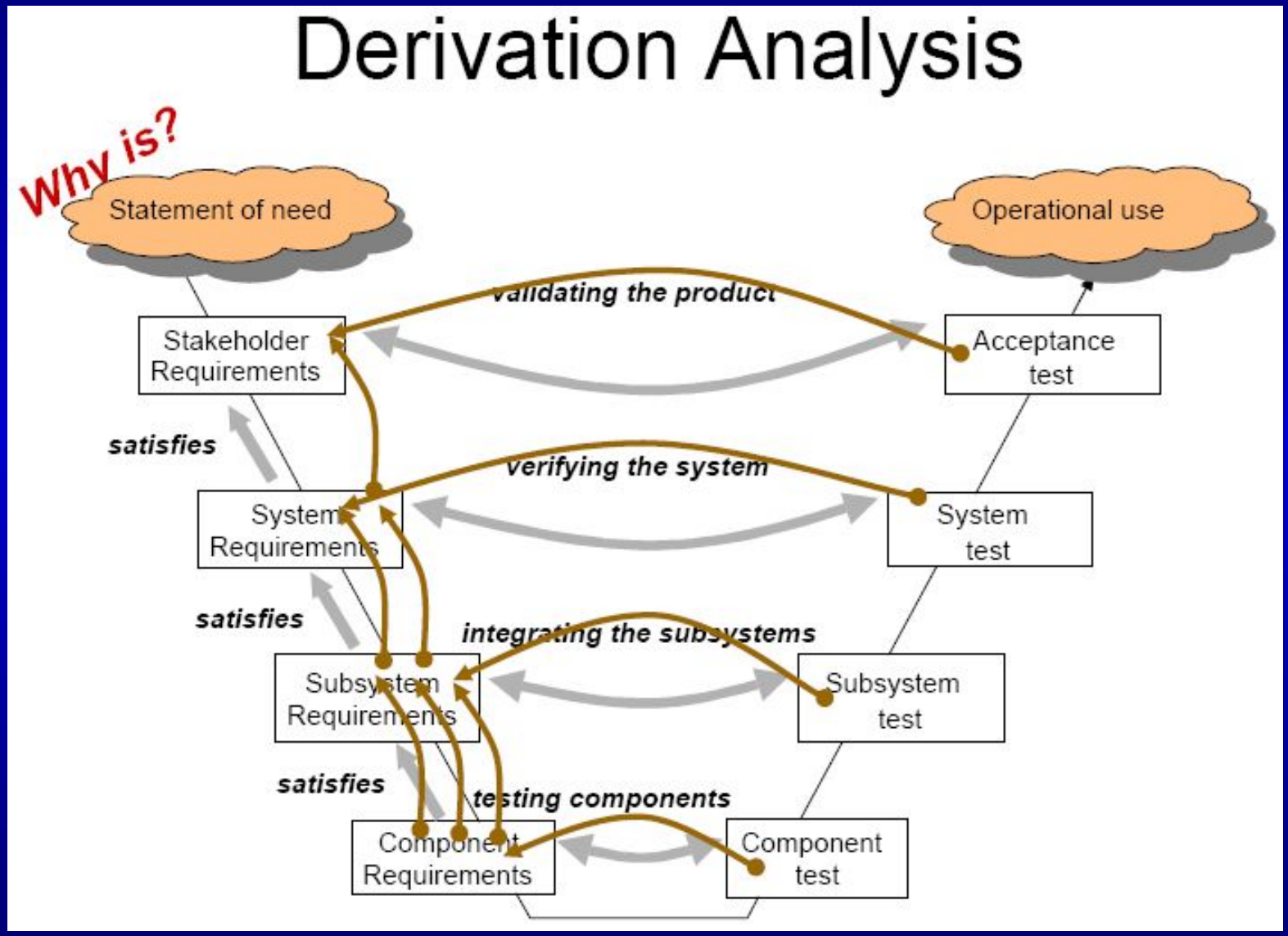
Validation W-model



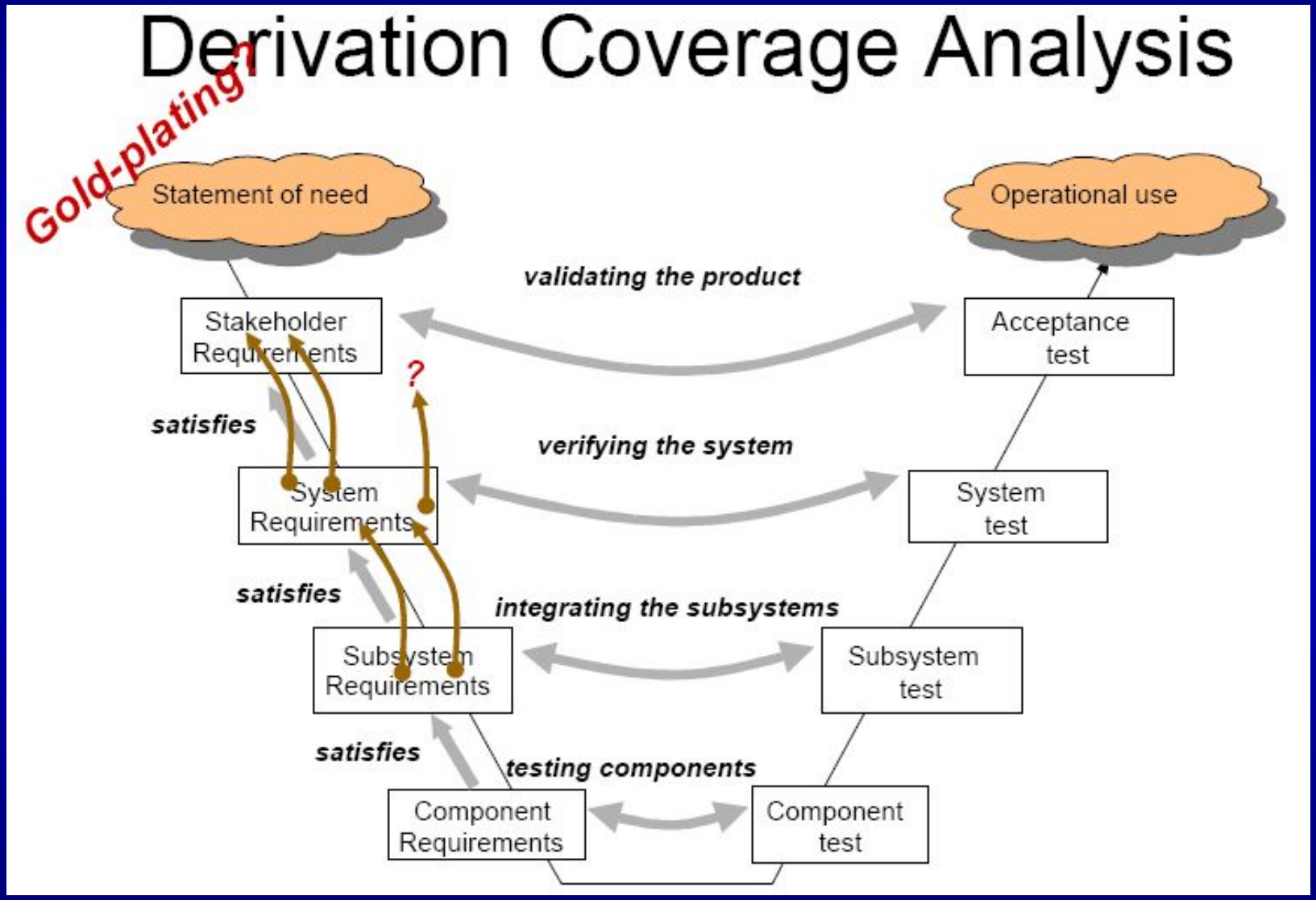
Traceability -- Forward



Traceability -- Backward



Traceability – Gold Plating



需求垂直追溯表

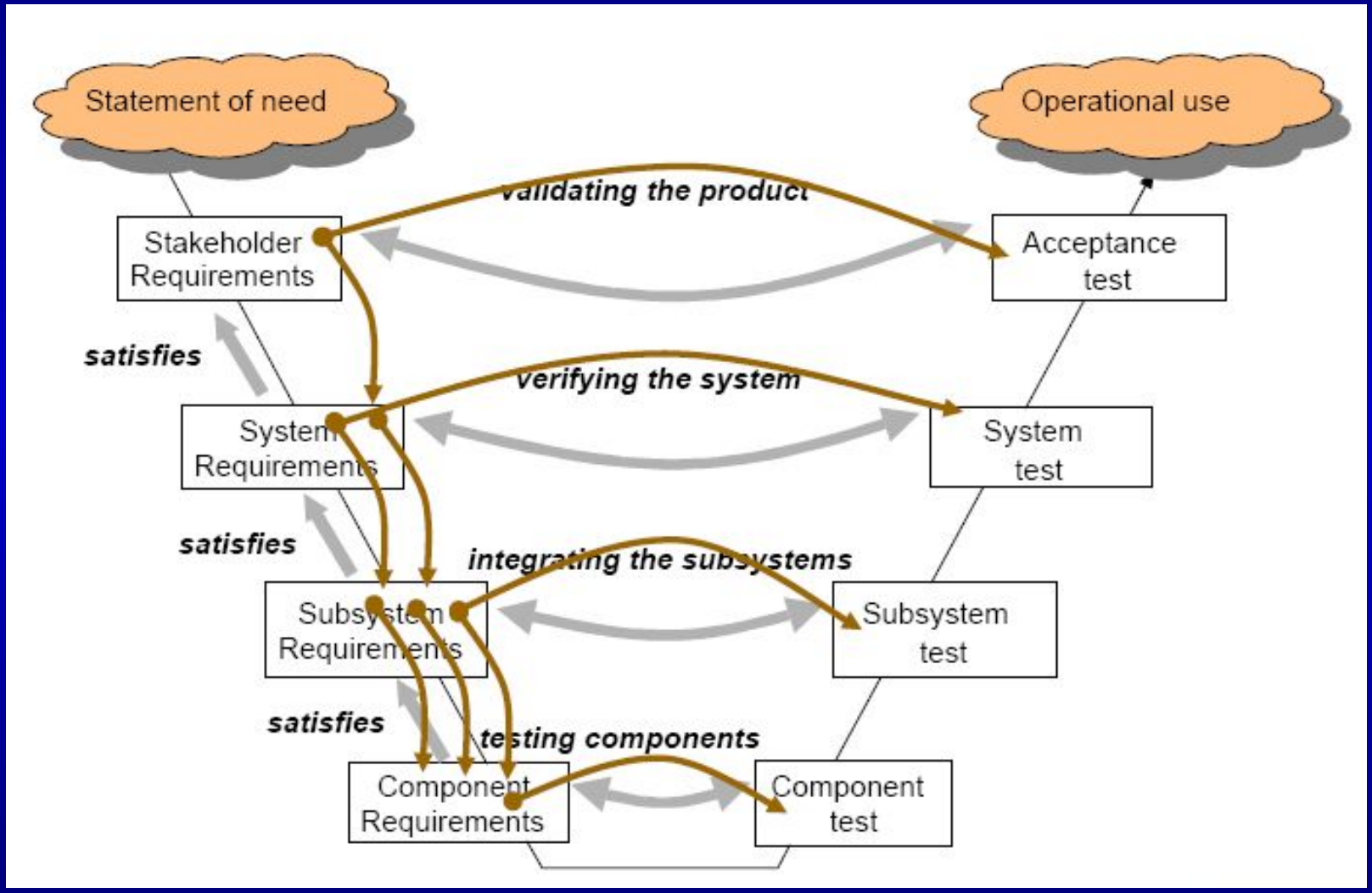
需求編號	原始需求	WBS	系統規格	軟體需求規格	軟體設計文件	軟體程式文件	軟體測試文件	軟體測試個案
UR.1	RFP/ Proposal/ 合約/ 會議記錄	WBS-1	RD.1	RS.1.1	DM.1.1.1	CM.1.1.1	TP.1.1.1	TC.1.1.1
				RS.1.2	DM.1.2.1	CM.1.2.1	TP.1.2.1	TC.1.2.1
					DM.1.2.2	CM.1.2.2	TP.1.2.2	TC.1.2.2
				RS.1.3	DM.1.3.1	CM.1.3.1	TP.1.3.1	TC.1.3.1
UR.2	RFP/ Proposal/ 合約/ 會議記錄	WBS-2	RD.2	RS.2.1	DM.2.1.1	CM.2.1.1	TP.2.1.1	TC.2.1.1
					DM.2.1.2	CM.2.1.2	TP.2.1.2	TC.2.1.2
				RS2.2	DM.2.2.1	CM.2.2.1	TP.2.2.1	TC.2.2.1

需求水平追溯表

相依關係

需求編號	R1	R2	R3	R4	R5	R6
R1			*	*		
R2					*	*
R3				*	*	
R4		*				
R5						*
R6						

Impact Analysis



Traceability Benefits

- Change Impact Analysis
- Certification
- Maintenance
- Project Tracking
- Reengineering
- Reuse
- Risk Reduction
- Testing

SRS 審查與承諾

- 所有專案開發關係人員(PM, RA/SA, Dev, Test)應參加需求基線審查會議，評估需求對現有承諾的影響，協商與取得專案參與人員對需求的承諾，並記錄監控承諾事項
- 內部審查
  - 取得專案成員對需求的承諾
- 客戶審查
  - Not Sign Off, But Sign On
- 申請發佈SRS進Baseline，成為需求基線(H15B11-基線發佈申請單)

需求變更

提出者：開發方所有專案成員
客戶方資深工程師
接收者：開發方指定一名SA作為接收者
要求：所有變更請求都要填寫“變更申請單”
CMR更新“變更追溯表”
輸出：變更申請單，更新的“變更追溯表”

接收者判斷是否為重大需求變更
要求：接收者判斷意見紀錄在“變更申請單”中，
提交專案經理，如果不是重大變更，
專案經理可以直接同意變更
輸出：更新的“變更申請單”

根據接收者的判斷意見，專案經理組織臨時評估組對變更請求進行影響分析
輸出：更新的“變更申請單”和“變更追溯表”

CCB及客戶方專案經理根據評估結果作出是否實施變更的決策
要求：CCB以會議方式進行決策，要有會議紀錄
如果CCB批准變更，專案經理要做出變更計畫，並確定驗證方式
輸出：CCB會議紀錄、“變更申請單”、“變更追溯表”

專案成員根據專案經理的安排對相關工作產品進行修改
要求：專案經理追蹤工作產品修改的進度
輸出：完成修改的工作產品
更新的“變更申請單”和“變更追溯表”

經過CCB批准的變更，在工作產品修改完成後，要及時用變更計畫中確定的驗證方式進行驗證
沒做影響分析和和CCB審查的變更，在工作產品修改完成後可以非正式地驗證，或等待一定時機再驗證
輸出：驗證紀錄、通過驗證的工作產品
更新的“變更申請單”和“變更追溯表”

根據基線發佈流程發布基線
輸出：更新的“變更申請單”和“變更追溯表”
“基線狀態報告”

在“變更申請單”中紀錄取消原因。
輸出：更新的“變更申請單”和“變更追溯表”。

變更流程

H13B51-變更管制指引

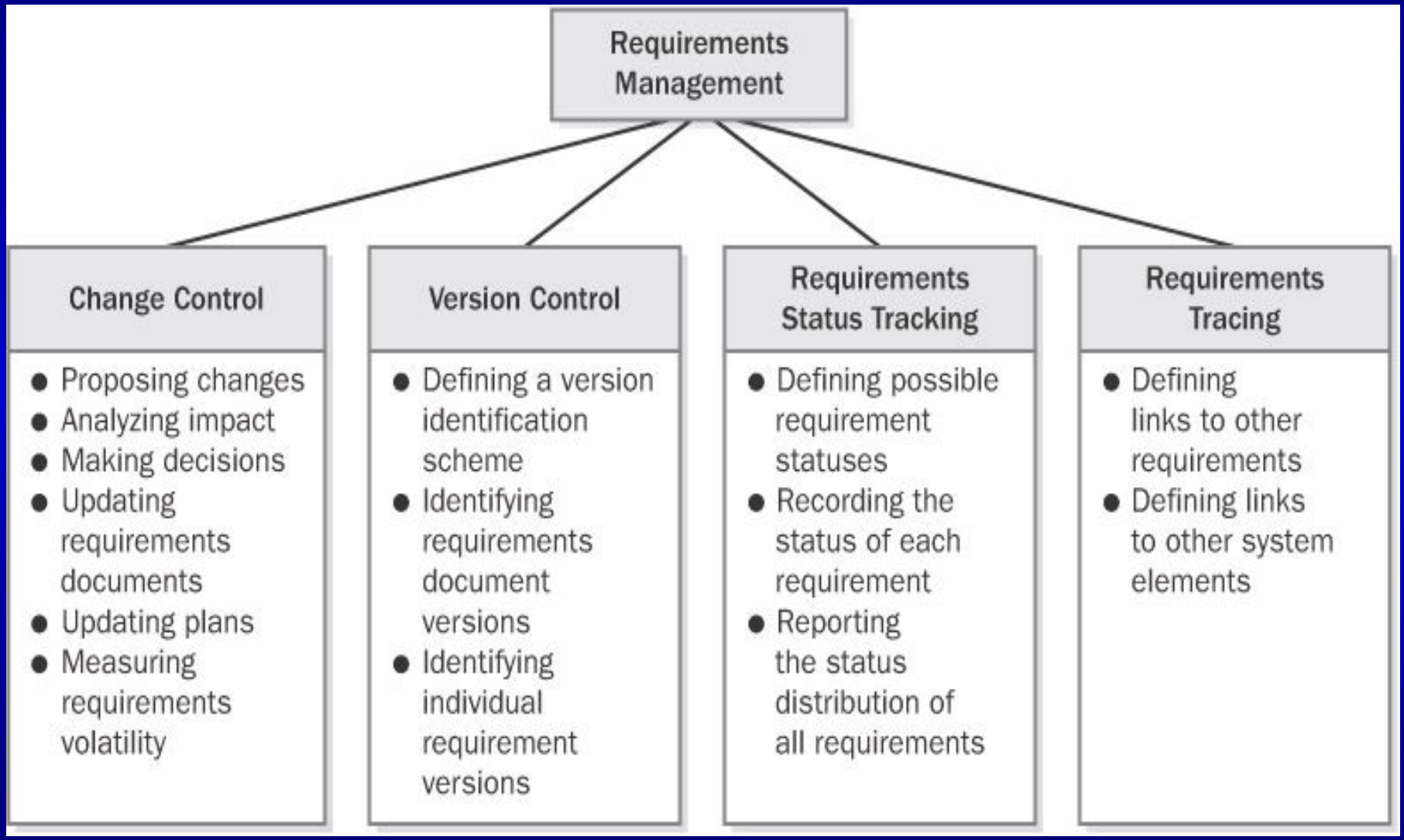
- 提出變更申請
- 影響評估
- CCB批核
- 實施變更
- 驗證變更結果
- 發佈變更結果

H13B51-變更管制指引

CCB (Configuration Control Board)

- 當然成員
  - 專案經理(主席)、QAR、CMR
- 計畫變更
  - 須有高階經理(部門主管)參與批核
- 範圍變更
  - 須有高階經理與客戶方負責人參與批核
- 需求變更
  - 應儘量邀請客戶方負責人參與批核確認

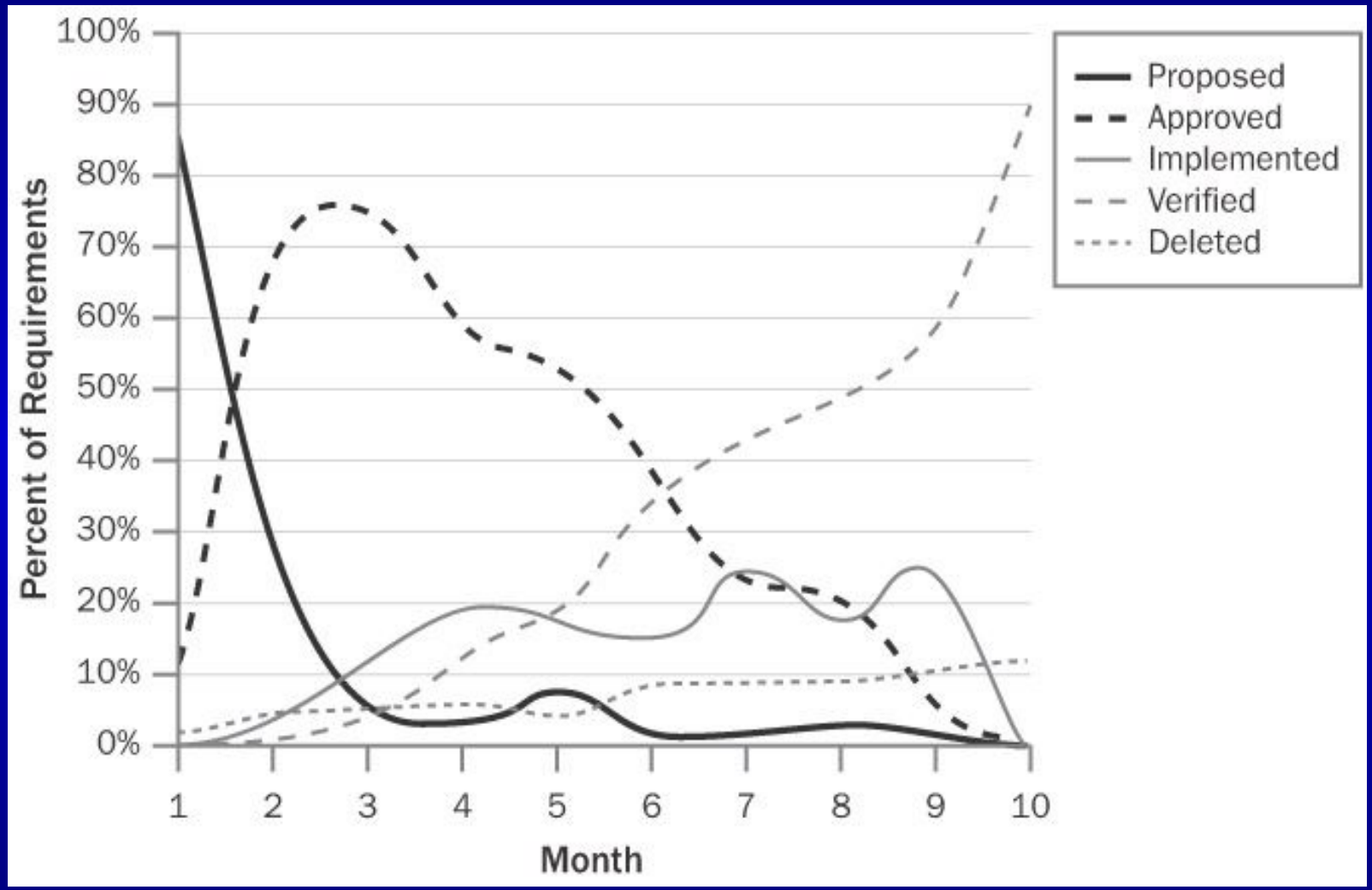
需求管理



Requirement Status

- Proposed
- Approved
- Implemented
- Verified
- Deleted
- Rejected
- Deferred

Requirement Status Trace



Reference Webs:

- Volere
  - http://www.volere.co.uk

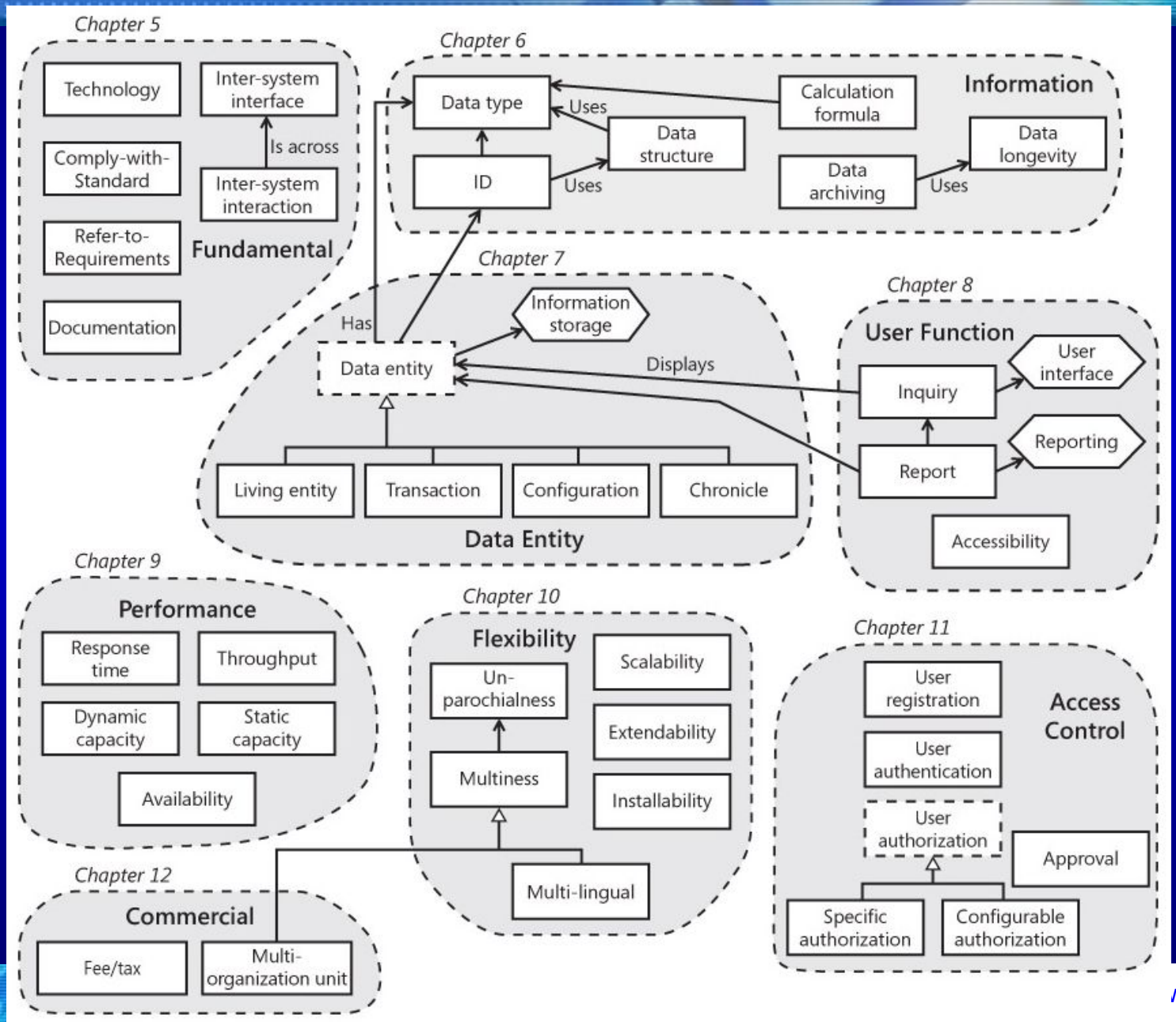
Reference Books:

- Karl Wiegers, “Software Requirements,” 2nd Ed., Microsoft Press, 2003
- Suzanne & James Robertson, “Mastering Requirements Process”, 2nd Ed., Addison Wesley, March 17, 2006
- Steve Adolph & Paul Bramble, “Patterns for Effective Use Cases”, Addison-Wesley,2003
- Donald Gause & Gerald Weinberg, “Exploring Requirements: Quality Before Design”, Dorset House, 1989

Reference Books:

- Stephen Withall, “Software Requirement Patterns,” Microsoft Press, 2007

Requirement Patterns



web.com.tw/

Fundamental

- Inter-System Interface
- Inter-System Interaction
- Technology
- Comply-with-Standard
- Refer-to-Requirements
- Documentation

Information

- Data Type
- Data Structure
- ID
- Calculation Formula
- Data Longevity
- Data Archiving

Data Entity

- Living Entity
- Transaction
- Configuration
- Chronicle
- Information Storage

User Function

- Inquiry
- Report
- Accessibility
- User Interface Infrastructure
- Reporting Infrastructure

Performance

- Response Time
- Throughput
- Dynamic Capacity
- Static Capacity
- Availability

Flexibility

- Scalability
- Extendability
- Unparochialness
- Multiness
- Multi-Lingual
- Installability

Access Control

- User Registration
- User Authentication
- User Authorization
- Specific Authorization
- Configurable Authorization
- Approval

Commercial

- Multi-Organization Unit
- Fee/Tax

評核方式(How to Evaluate)

- 回顧一開始的 焦油坑 跟 問題徵狀,

  這裡講的方法能有效處理那些狀況嗎？

Are Your Lights On？

您想通了嗎？