


Recall The Team Skills

1. Analyzing the Problem (with 5 steps)
2. Understanding User and Stakeholder Needs
3. Defining the System
4. Managing Scope
5. **Refining the System Definition**
 1. Software Requirements: a more rigorous look
 2. Refining the Use cases
 3. Developing the Supplementary Specification
 4. On Ambiguity and Specificity
 5. Technical Methods for Specifying Requirements
6. Building the Right System

Chapter 24

Technical Methods for Specifying Requirements

- 
- Pseudo-code
 - Finite state machines
 - Decision tables and decision trees
 - Activity diagrams (flowcharts)
 - Entity-relationship models

Introduction

- There are cases in which the **ambiguity** of natural language is simply **not tolerable**
 - **Examples:** when the requirements deal with **life-and-death issues** or when the erroneous behavior of a system could have **extreme financial or legal consequences**.
- If the description of the **requirement is too complex** for natural language and if you cannot afford to have the specification misunderstood, **you should** consider writing or **augmenting** that portion of the requirements set **with a "technical methods"** approach.

Technical Specification Methods

- Pseudo-code
- Finite state machines
- Decision tables and decision trees
- Activity diagrams (flowcharts)
- Entity-relationship models
- ...

Pseudo-code

- Pseudo-code is a "quasi" programming language
 - An attempt to **combine the informality** of natural language **with the strict syntax and control structures** of a programming language.
- It usually consists of combinations of:
 - **Imperative sentences** with a single verb and a single object
 - **A limited set**, typically not more than 40–50, of **"action-oriented" verbs** from which the sentences must be constructed
 - **Decisions** represented with a formal IF-ELSE-ENDIF structure
 - **Iterative activities** represented with DO-WHILE or FOR-NEXT structures

Example of Pseudo-code

- A pseudo-code specification of an algorithm for calculating deferred-service revenue earned for any month is:

Set Sum(x)=0

FOR each customer X

IF customer purchased paid support

AND ((Current month) \geq (2 months after ship date))

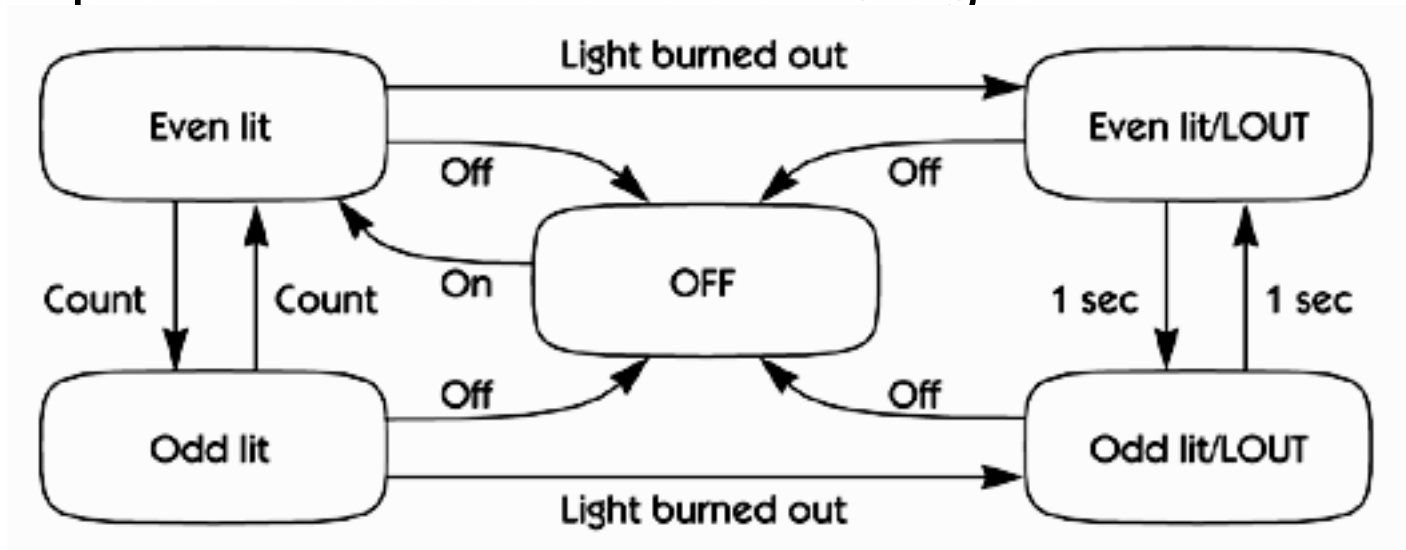
AND ((Current month) \leq (14 months after ship date))

THEN Sum(X)=Sum(X) + (amount customer paid)/12

END

Finite State Machines

- Example of a state transition diagram



- The natural language expression "the other light shall flash every 1 second" was ambiguous.
- The above state transition diagram is not ambiguous since it illustrates that duty cycle B was indeed the right choice.

Finite State Machines

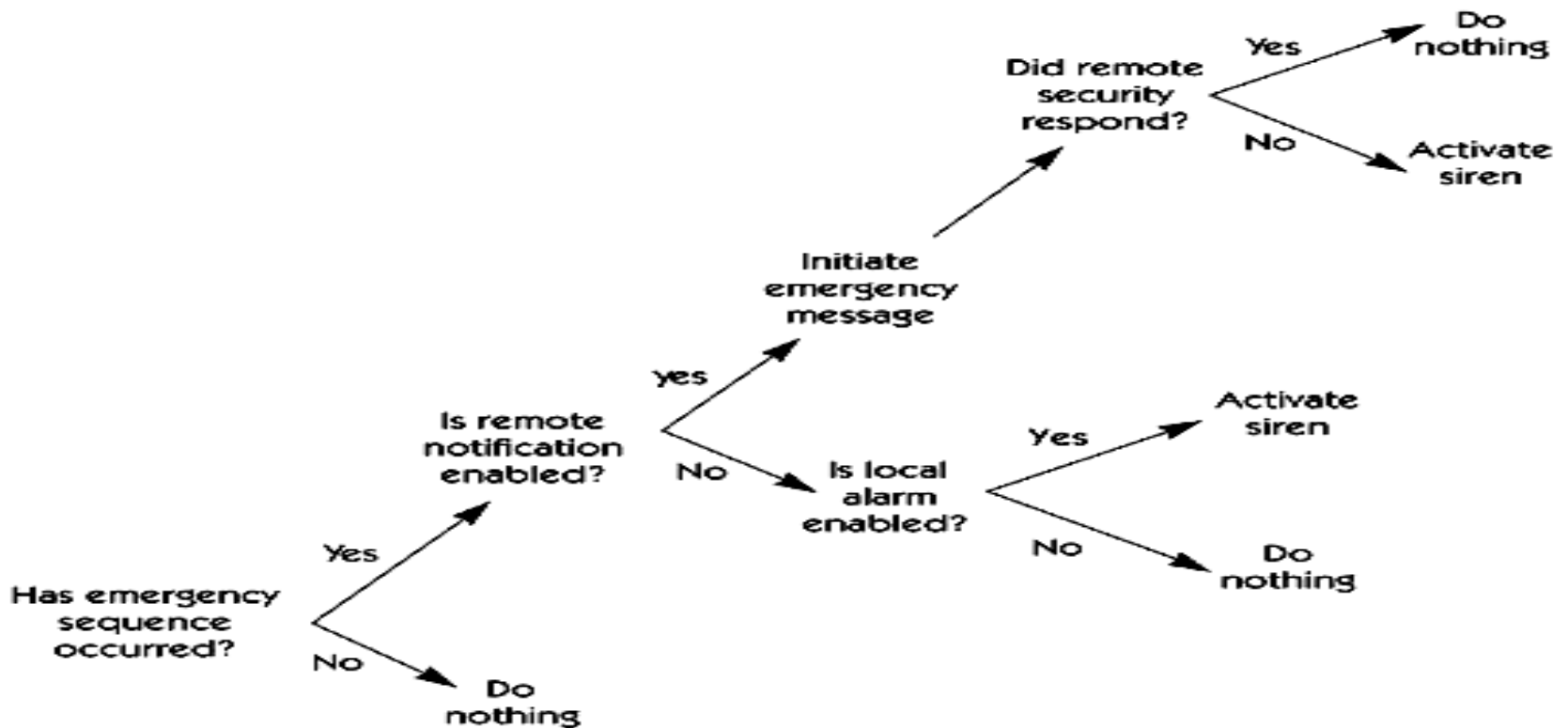
- Example of a State Transition Matrix for an On/Off Counting Device:

State	Event					
	On Press	Off Press	Count Press	Bulb Burns Out	Every Second	Output
<i>Off</i>	Even lit	–	–	–	–	Both off
<i>Even lit</i>	–	Off	Odd lit	LO/Even lit	–	Even lit
<i>Odd lit</i>	–	Off	Even lit	LO/Odd lit	–	Odd lit
<i>Light out/Even lit</i>	–	Off	–	Off	LO/Odd lit	Even lit
<i>Light out/Odd lit</i>	–	Off	–	Off	LO/Even lit	Odd lit

- What happens if the user presses the On switch and the device is already on? Answer: Nothing.
- What happens if both bulbs are burned out? Answer: The device powers itself off.

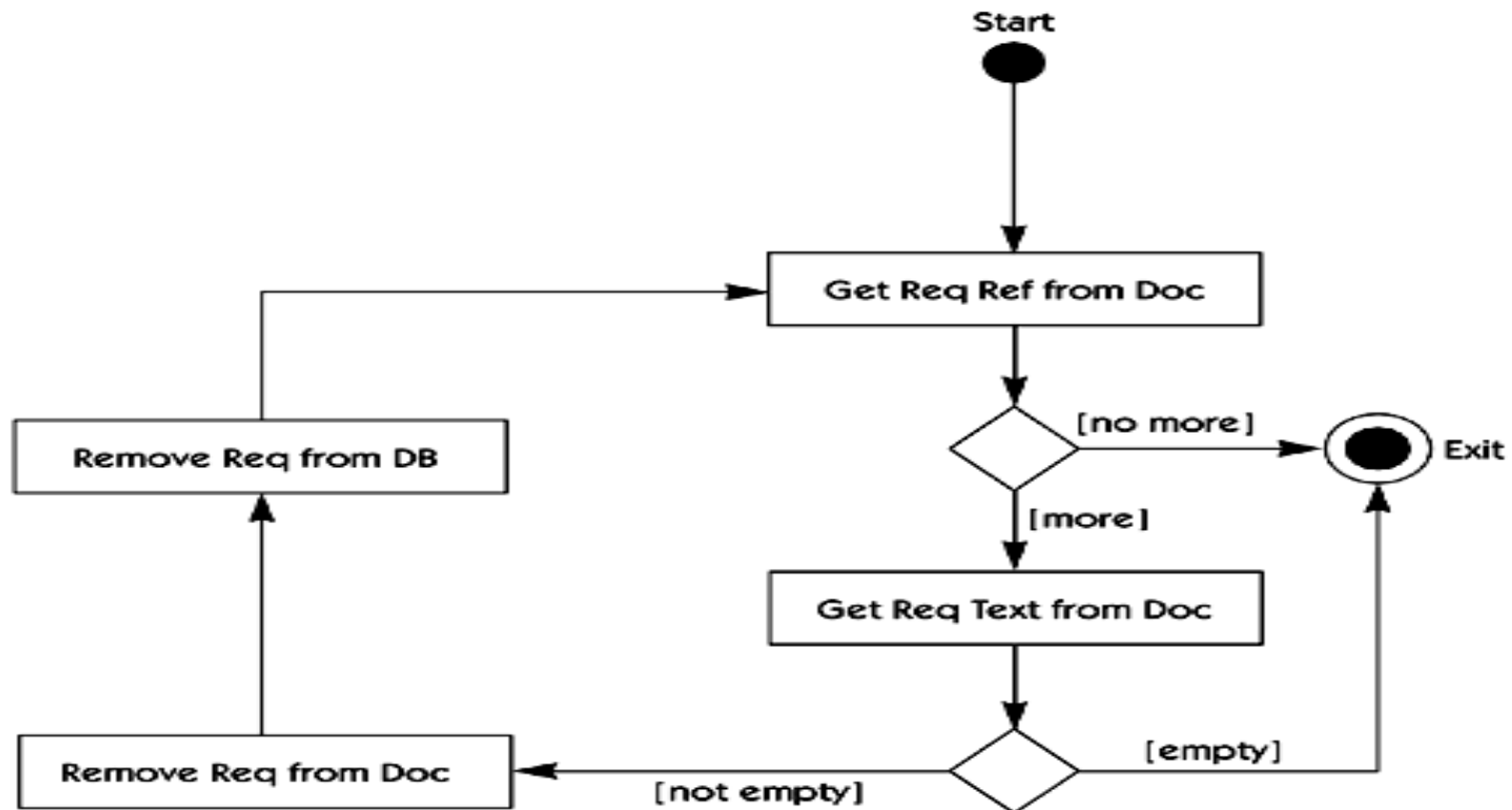
Decision Tables and Decision Trees

- It's common to see a requirement that deals with a combination of inputs; different combinations of those inputs lead to different behaviors or



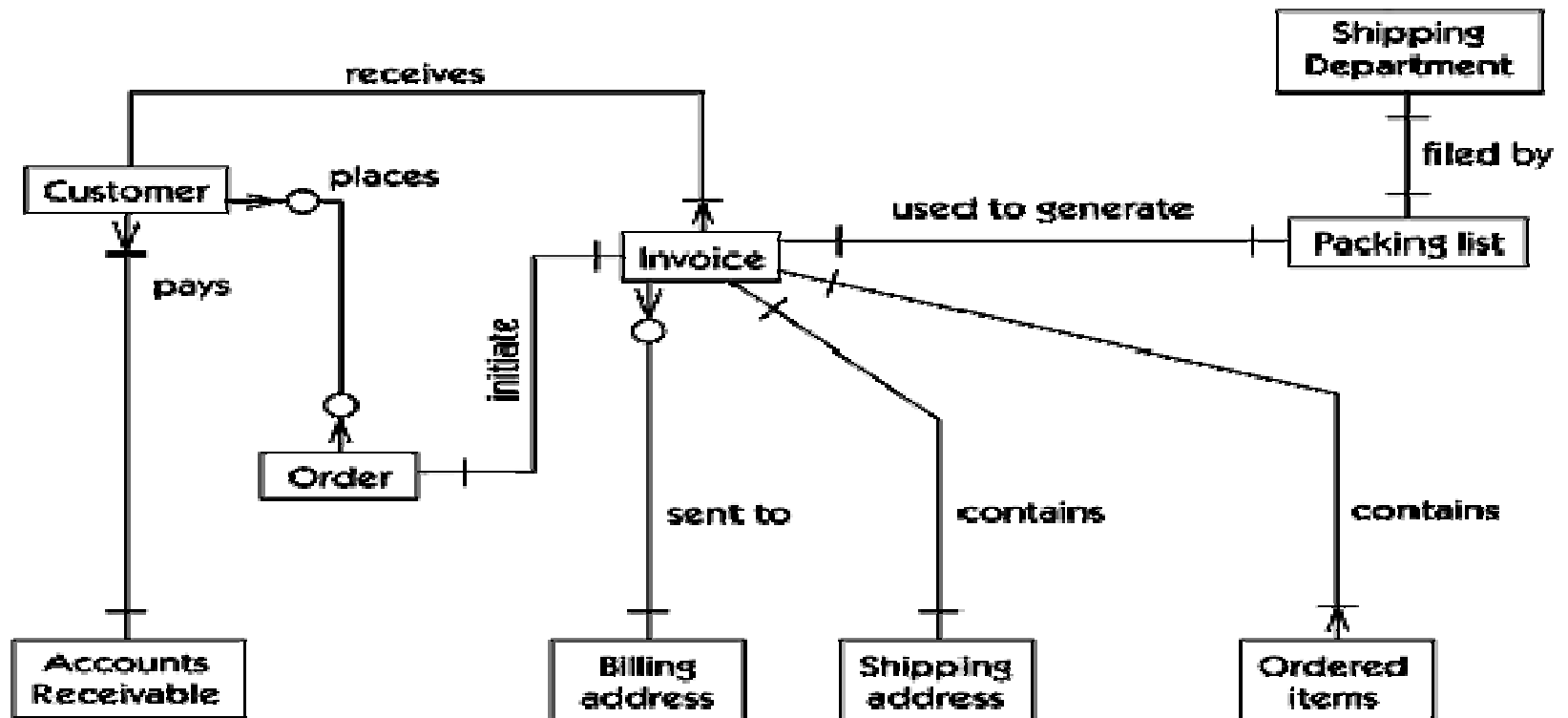
Activity Diagrams

- An activity diagram is essentially a flowchart, showing flow of control from activity to activity.



Entity-Relationship Models

- If the requirements within a set involve a description of the structure and relationships among data within the system, it's often convenient to represent that information in an entity-relationship diagram (ERD).



Key Points

- ❑ Technical methods for specifying requirements are appropriate when the requirement description is too complex for natural language or if you cannot afford to have the specification misunderstood.
- ❑ Technical methods include pseudo-code, finite state machines, decision trees, activity diagrams, entity-relationship models, and many others.