Systems Development as a Translation Process

Planning

User Business Objectives

Analysis

Implicit User Requirements

Analysis

Explicit User Requirements

Analysis

User Requirement Models

Design

System Design Models

Programming

Source Code

Compiling

Testing/Debugging

Prototyping

Object Code

Installation

Acceptance Testing

Installed System

Implementation

User Feedback

Implemented System

Maintenance

Maintained System
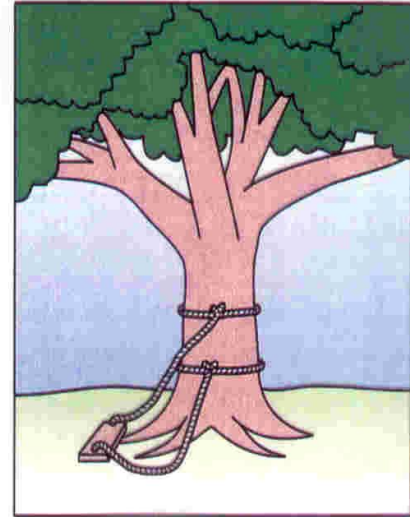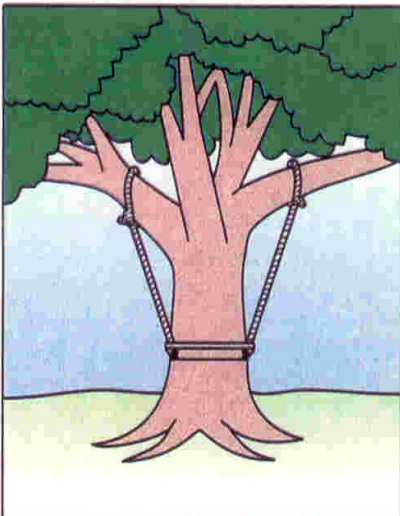
Poor Communication in SDLC: A Case Study
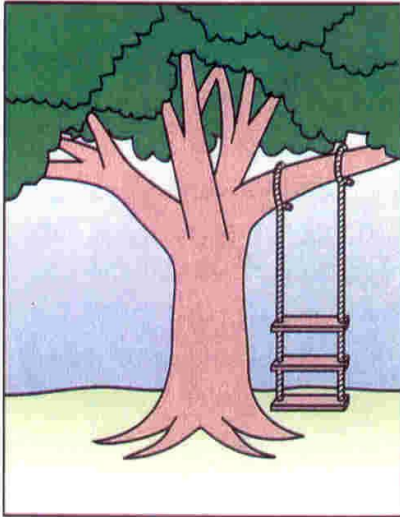
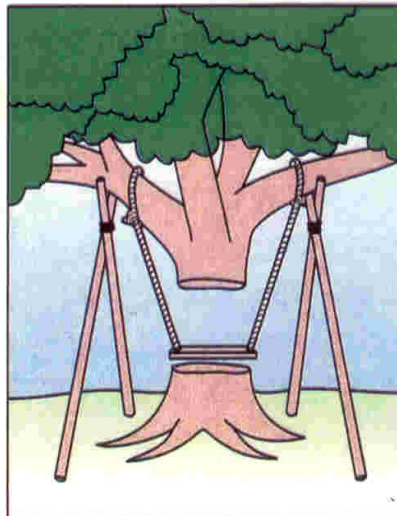As Stated in the Functional Requirements
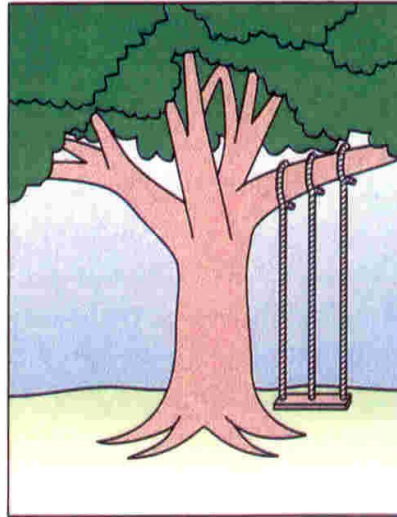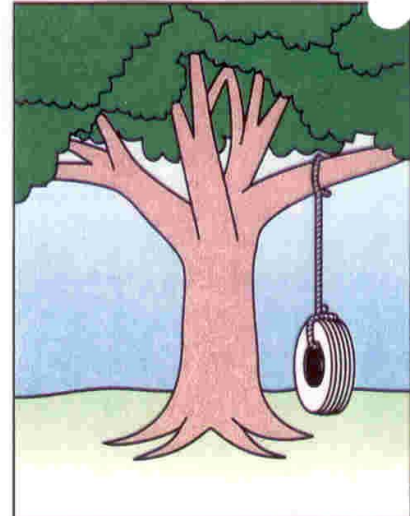
As Outlined in the System Specifications

As Designed by the Systems Analyst

As Implemented by Information Services

As Operated by the End User

What the End User Really Needed

User Business Objectives

To maximize profits

⬇

To maximize revenues (vs. to minimize cost)

⬇

To project an image of safety (vs. reliability, economy, availability, etc.)
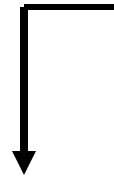
⬇

To avoid fatal accidents (vs. near misses, emergency landings,)

⬇

To prevent acts of terrorism (vs. mechanical problems, maintenance problems)

⬇

To prevent terrorists from checking in explosive-carrying luggage and then failing to board (vs. kamikaze acts)

> To assure passengers missing connecting flights that their luggage will stay where they are.

To prevent luggage from being on-board without owner traveling

Implicit User Requirements

To assure that every passenger who has checked in luggage will also board the aircraft, or his/her luggage will be removed prior to takeoff.

Policy: Passengers and their luggage should never be separated.

Explicit User Requirements

We need a system that will enable us

- **To <u>remove</u> the LUGGAGE of all those PASSENGERS who have <u>checked in</u> luggage but who have not <u>boarded</u> the DEPARTING AIRPLANE yet (by identifying that luggage to luggage control).**

User Requirement Models

What physical operations take place in the user environment?
- ➤ <u>Check</u> in luggage
- ➤ <u>Board</u> passenger
- ➤ <u>Remove</u> luggage

Information systems support physical, real-world operations.

What information operations (data processes) take place in the user environment?
- ➤ <u>Check</u> in luggage
- ➤ <u>Board</u> passenger
- ➤ <u>Remove</u> luggage

The Required Operational System

Passenger

Luggage

1. Check in Luggage

Passenger

2. Board Passenger

Luggage

Passenger

The Plane's "Belly"   **The Plane's Cabin**   **Cockpit**

Luggage

3. Remove Luggage

Luggage

Temporary Storage Area
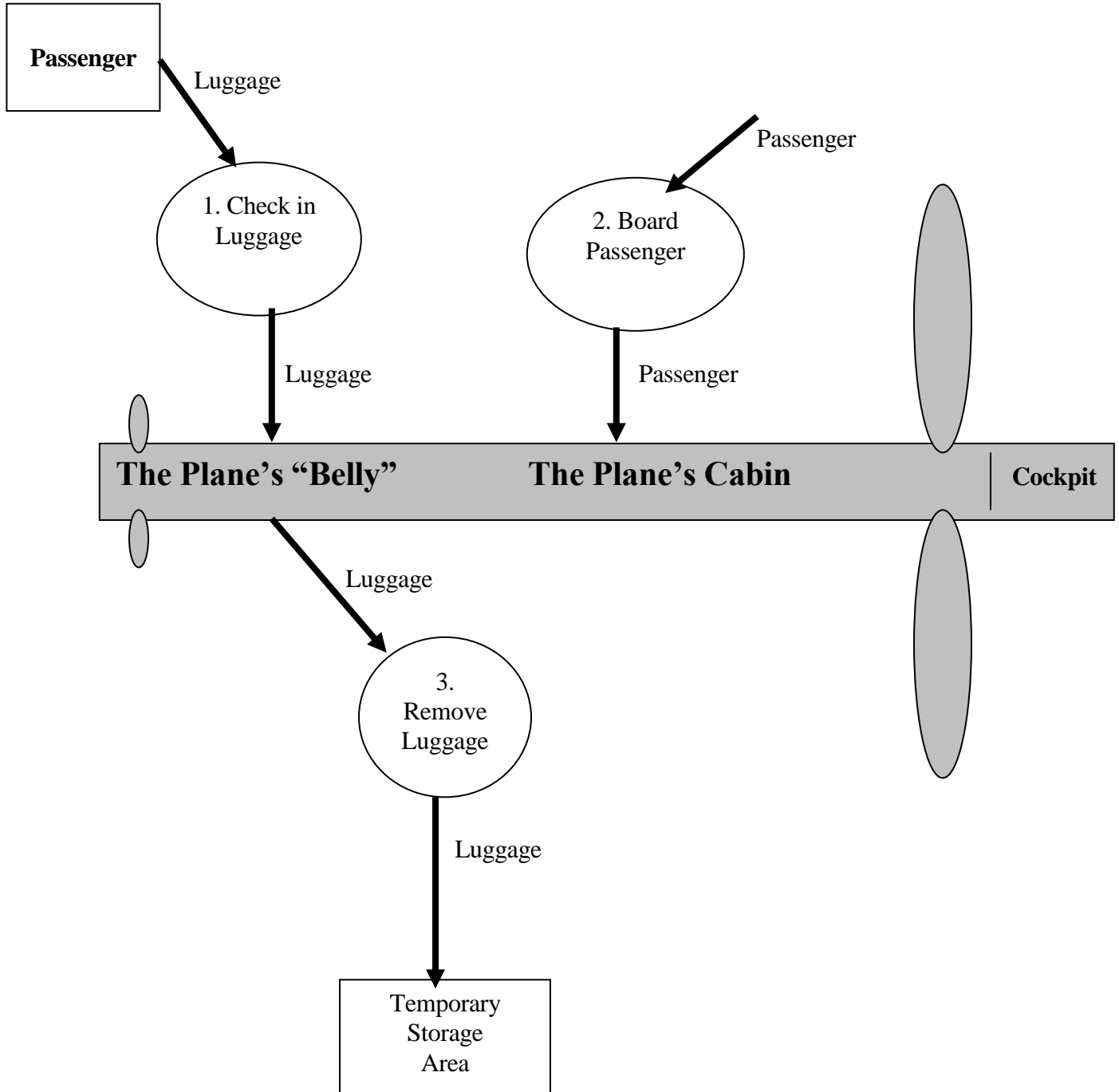
The Required Information System

Passenger

Luggage

Passenger

Boarding Pass

1. Check in Luggage

2. Board Passenger

Luggage Details

Boarding Status

FLIGHT

Boarding Data

3. Produce Remove-Luggage Report

Remove-Luggage Report

Luggage Control
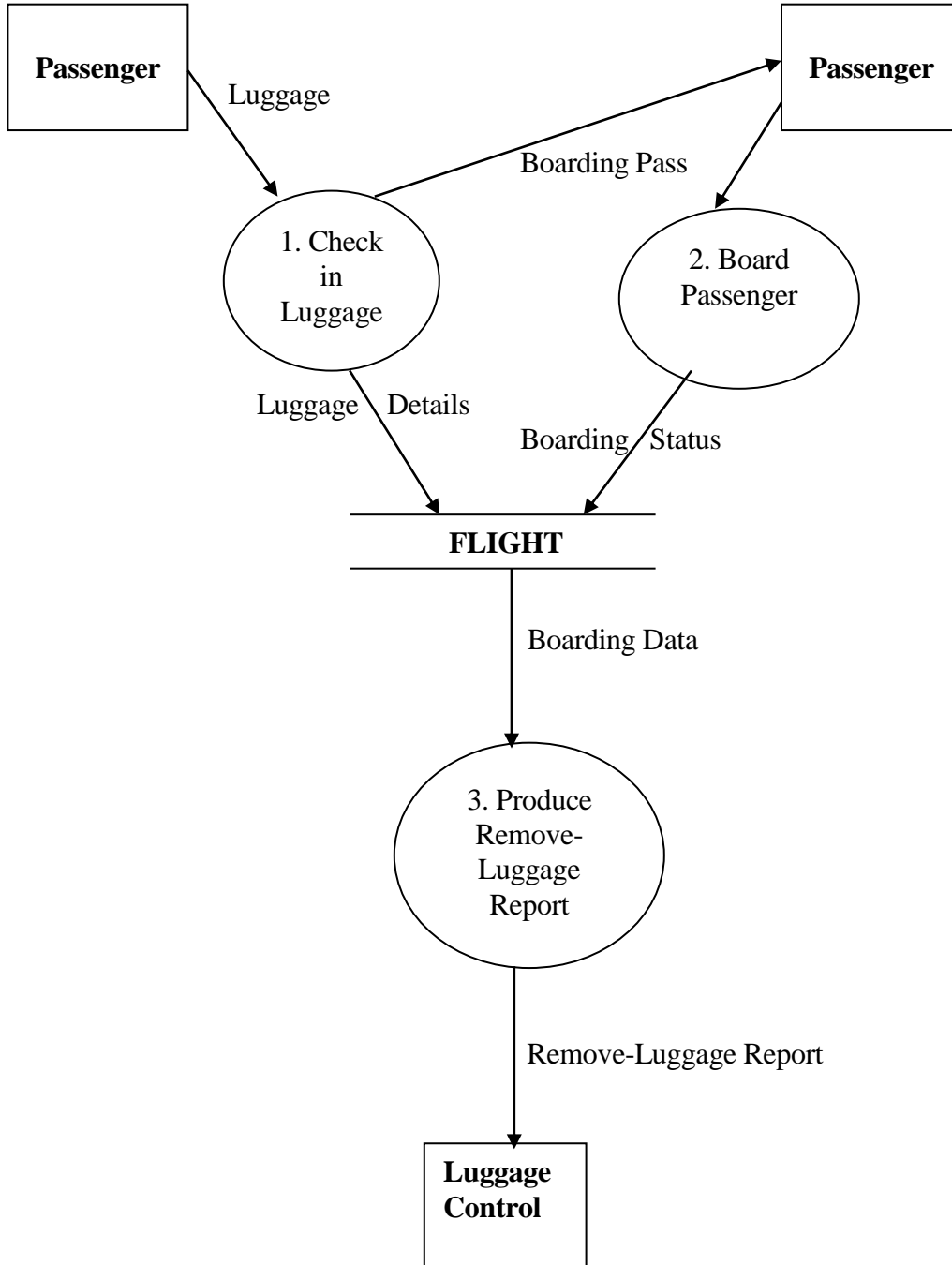
Systems Design Components
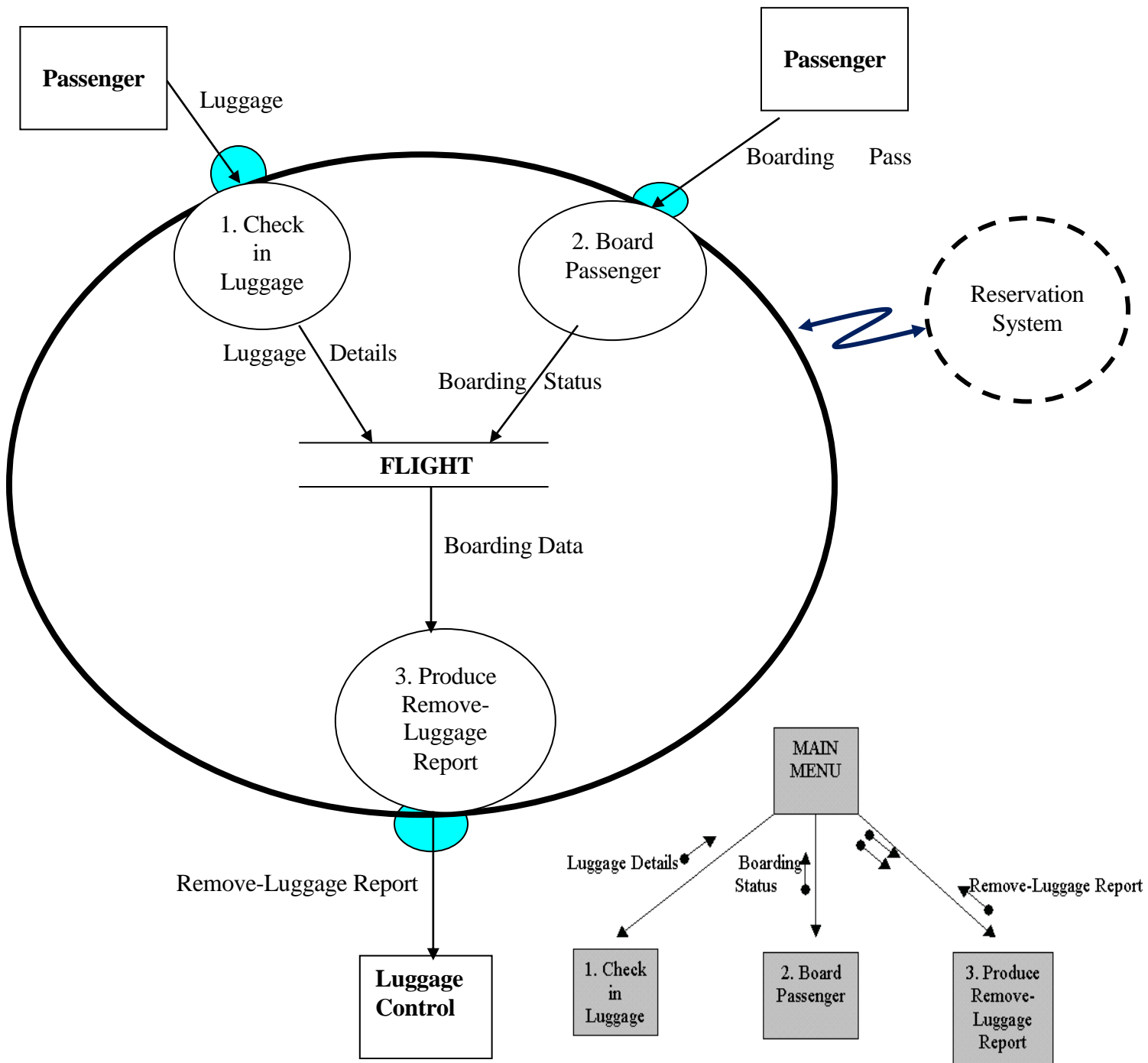
Design of
- data elements
- database
- process logic
- user interface
- system interfaces
- program structure

Passenger

Luggage

Passenger

Boarding   Pass

1. Check in Luggage

2. Board Passenger

Reservation System

Luggage   Details

Boarding   Status

FLIGHT

Boarding Data

3. Produce Remove-Luggage Report

MAIN MENU

Remove-Luggage Report

Luggage Control

Luggage Details

Boarding Status

Remove-Luggage Report

1. Check in Luggage

2. Board Passenger

3. Produce Remove-Luggage Report

	Boarded	Not boarded
Checked in luggage	1	2
Did not check in luggage	3	4

Passenger ID	Flight Number	Luggage 1 ID	Luggage 2 ID	Boarding Status
1	111	1A	1B	Y
2	222	2A	2B	N
3	333			Y
4	444			N

FOR every record in FLIGHT where (Luggage-1-ID OR Luggage-2-ID is not blank) AND (Boarding Status = N)

    Send **Flight-Number, Luggage-1-ID, Luggage-2-ID** to Luggage Control

ENDFOR

The Ultimate Translation

To <u>remove</u> the LUGGAGE of all those PASSENGERS who have <u>checked in</u> luggage but who have not <u>boarded</u> the DEPARTING AIRPLANE yet (by identifying that luggage to luggage control).

FOR every record in FLIGHT where (Luggage-1-ID OR Luggage-2-ID is not blank) AND (Boarding Status = N)

      Send **Flight-Number, Luggage-1-ID, Luggage-2-ID** to Luggage Control

ENDFOR

The Elements of Information System Design

Issues

	Static/Structure	**Dynamic/Flow**
Data	**1. Database** • What data need to be stored? • How are they defined? • How should these data be grouped (into files)? • How should these files be linked? *Entity-Relationship Diagram*	**2. Data Flow** What are the data flows into, within, and out of the system? • What data need to be entered? From whom? Through what interface? • What are the internal processes within the system? • What information needs to be produced by the system? To whom? Through what interface? *Data Flow Diagram* *Interface Navigation Diagram*
Process	**3. Program Structure** • What processes should the system perform? • How are these processes to be organized and related to one another? *Structure Chart* *Action Diagram*	**4. Process Logic** • What is the internal, step-by-step logic within each process, linking its given inputs to its required outputs? *Flowchart* *Structured English/Pseudocode*

A Slice of I.S. Functionality