System models

Objectives

- To introduce some types of system model that may be developed as part of the requirements engineering and system design processes.
- Understand how graphical models can be used to represent software systems
- have been introduced to some of the diagram types in the Unified Modeling Language (UML) and how these diagrams may be used in system modeling

Introduction

- User Requirement should be written in natural language.
- System Requirements may be expressed in a more technical way
- Document System specification →Set of system models
- These models are graphical representation that describe
    - Business processes
    - Problem to be solved
    - System that is to be developed

System Modelling

"System modeling is the process of developing abstract models of a system, with each model presenting a different view or perspective of that system."

- External perspective, where you model the context or environment of the system.
- Structural perspective, where you model the organization of a system or the structure of the data that is processed by the system.
- Behavioral perspective, where you model the dynamic behavior of the system and how it responds to events.

- Based on notations in the Unified Modeling Language (UML).

System Modeling

- Models are used during the requirements engineering process to help derive the requirements for a system and during the design process to describe the system.

- You may develop models of both the existing system and the system to be developed:

  - Models of the existing system are used during requirements engineering. They help clarify what the existing system does and can be used as a basis for discussing its strengths and weaknesses. These then lead to requirements for the new system.

  - Models of the new system are used during requirements engineering to explain the proposed requirements to other system stakeholders.

Model types

- Data processing model - how the data is processed at different stages.
- Composition model - How entities are composed of other entities.
- Architectural model - principal sub-systems.
- Stimulus/response model - the system’s reaction to events.

UML- Unified Modeling Language

- The Unified Modeling Language is a set of 13 different diagram types that may be used to model software systems.
- A major revision (UML 2) was finalized in 2004.
- The UML is universally accepted as the standard approach for developing models of software systems.

UML Diagrams

1. Activity diagrams, which show the activities involved in a process or in data processing.
2. Use case diagrams, which show the interactions between a system and its environment.
3. Sequence diagrams, which show interactions between actors and the system and between system components.
4. Class diagrams, which show the object classes in the system and the associations between these classes.
5. State diagrams, which show how the system reacts to internal and external events.
6. Data Flow diagrams which show how data is processed as it moves through the system

Context models

- At an early stage in the specification of a system, you should decide on the system boundaries.
- This involves working with system stakeholders to decide what functionality should be included in the system and what is provided by the system's environment.
- In some cases ,the boundary between a system and its environment is relatively clear
- e.g. Where an automated system is replacing an existing manual system...(how)
- Context models are used to illustrate the context of a system - they show what lies outside the system boundaries.

System boundaries

- System boundaries are established to define what is inside and what is outside the system.
- They show other systems that are used or depend on the system being developed.
- The position of the system boundary has a profound effect on the system requirements.

Context Diagram (CD)

- Produce a high-level model of an **existing** or **planned** system

- It defines the boundary of the system of interest and its interactions with the critical elements in its environment.

  A **Context Diagram** is a single picture that has the system of interest at the center, with no details of its interior structure or function, surrounded by those elements in its environment with which it interacts.

Why do it?

- There are many reasons for constructing a Context Diagram.

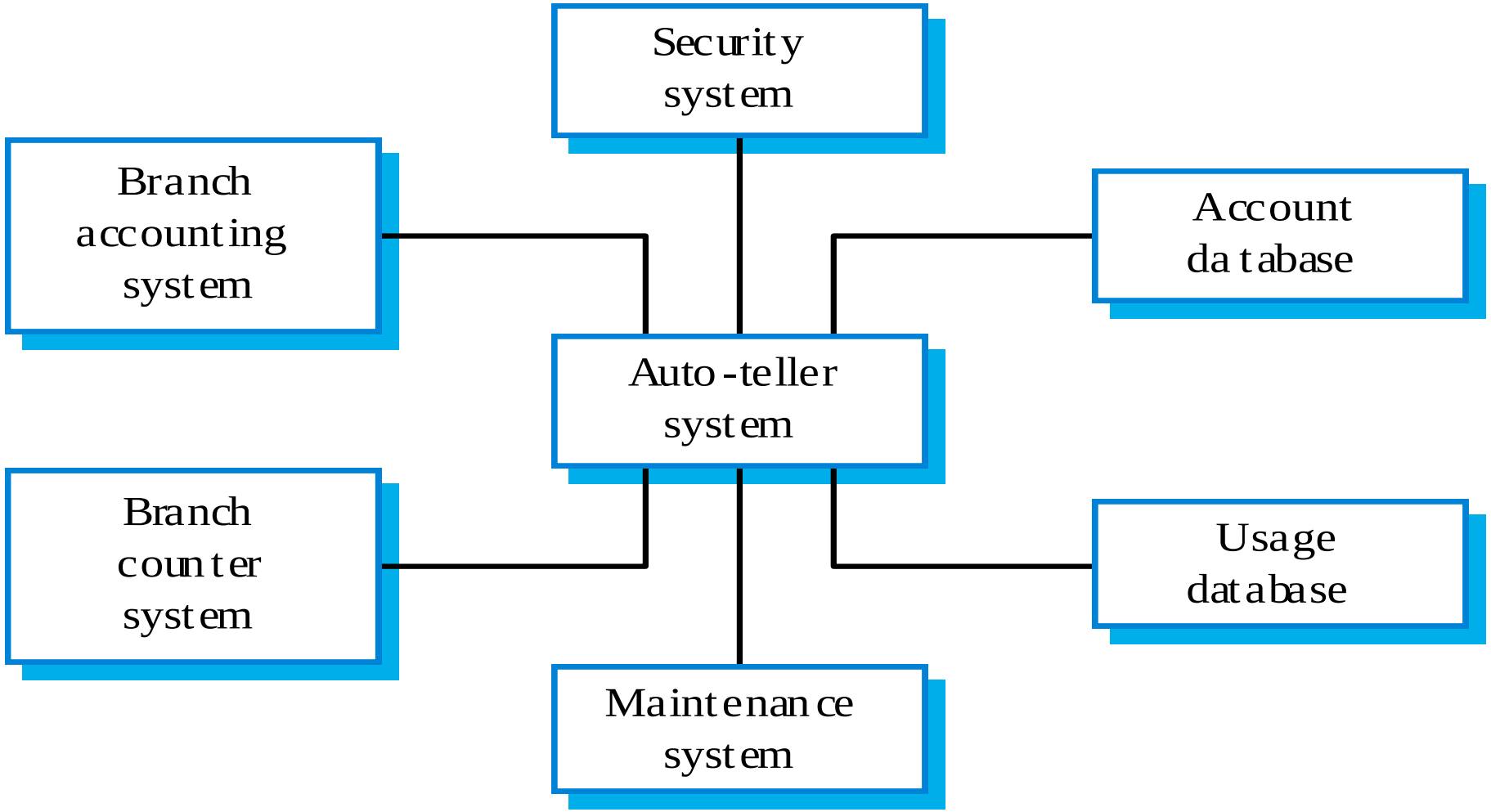
A Context Diagram can:

- define the scope or boundary of the system of interest
- provide a simple high-level picture of the system of interest.
- identify the elements in the environment of the system of interest that it interacts with
- allows the whole team to share information and agree at a common understanding

Where and when to use it?

- Context Diagrams can be used throughout a systems life cycle, but they are particularly useful in:
  - Understanding requirements for a new system.
  - Analyzing existing system

The context of an ATM system



How to do it?

- Constructing a Context Diagram is a relatively simple process.
- It starts with a single circle in the centre of a piece of paper, flip chart or white board.
- The circle should be labeled with the system name.
- It is also best practice to also write the Operational Requirements of the system in the circle to remind the team (or individual) of the system’s purpose and context.

Context Diagram (CD)

Context Diagram uses only three notational symbols.

The notational elements are:

SYSTEM of interest transforms input flows into output flows

TERMINATOR an external function or object that is the source or destination of flows external to the system of interest

FLOW an input or output quantity (information, control, material or energy)

Example of the day-to-day operation of a works Cafeteria.

The starting point for drawing a Context Diagram: the System of Interest as a bubble with its Operational Requirements

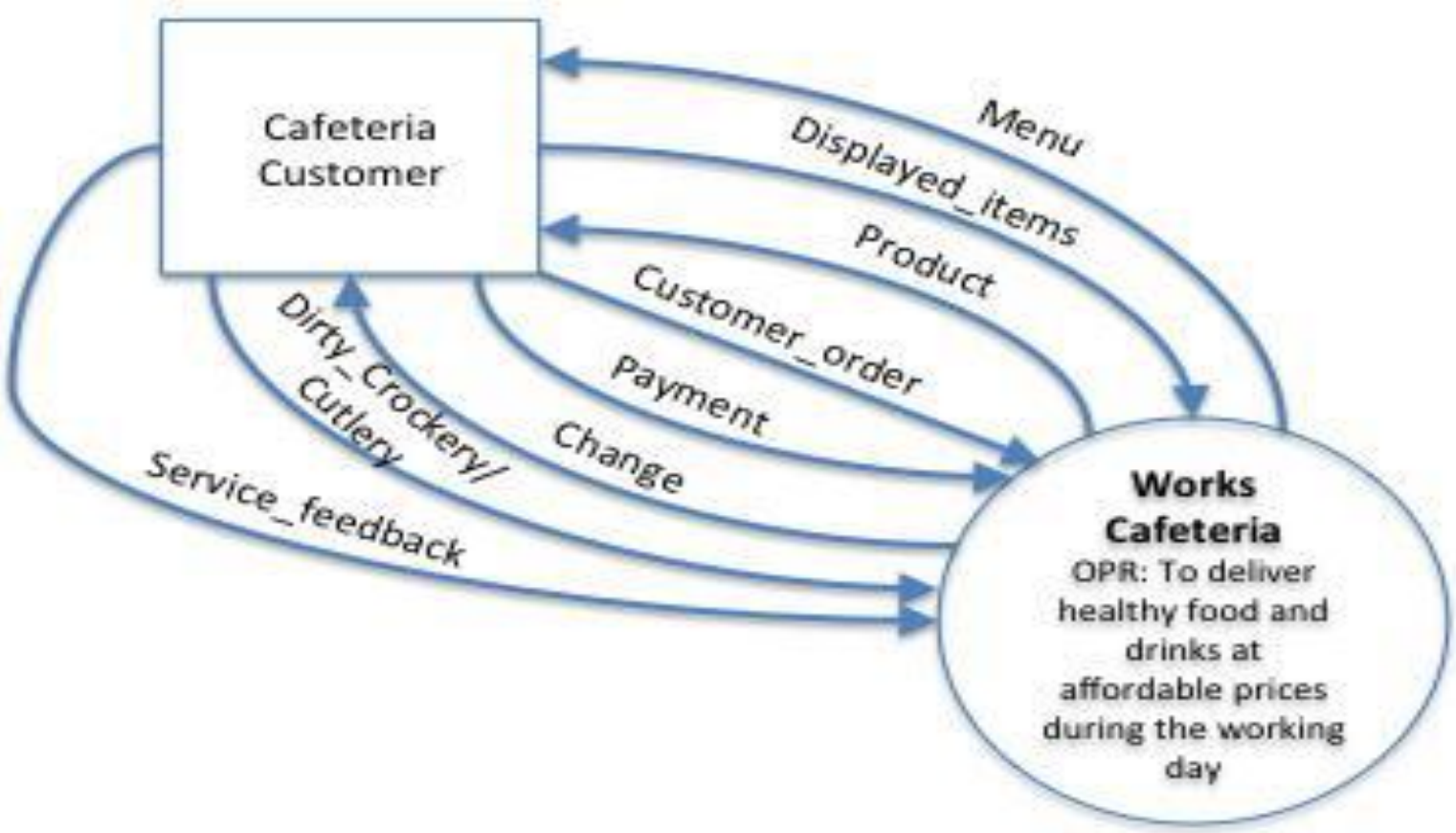
The next steps are concerned with identifying and documenting the terminators that the System of Interest interacts with .

There are two approaches here:

1. One-terminator at a time

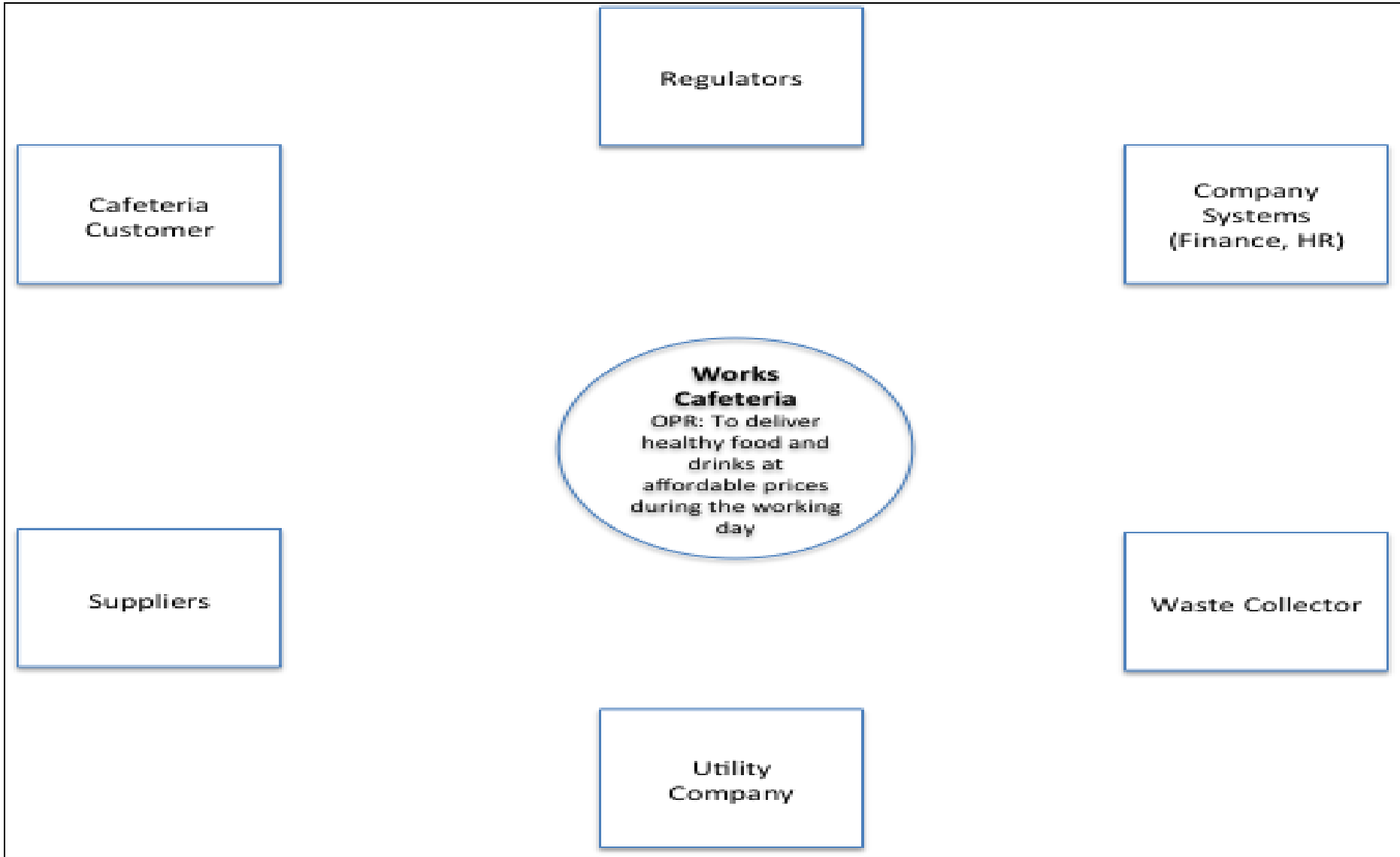
   In this approach each terminator is considered one at a time

Flow interaction between the Works Cafeteria and the Customer



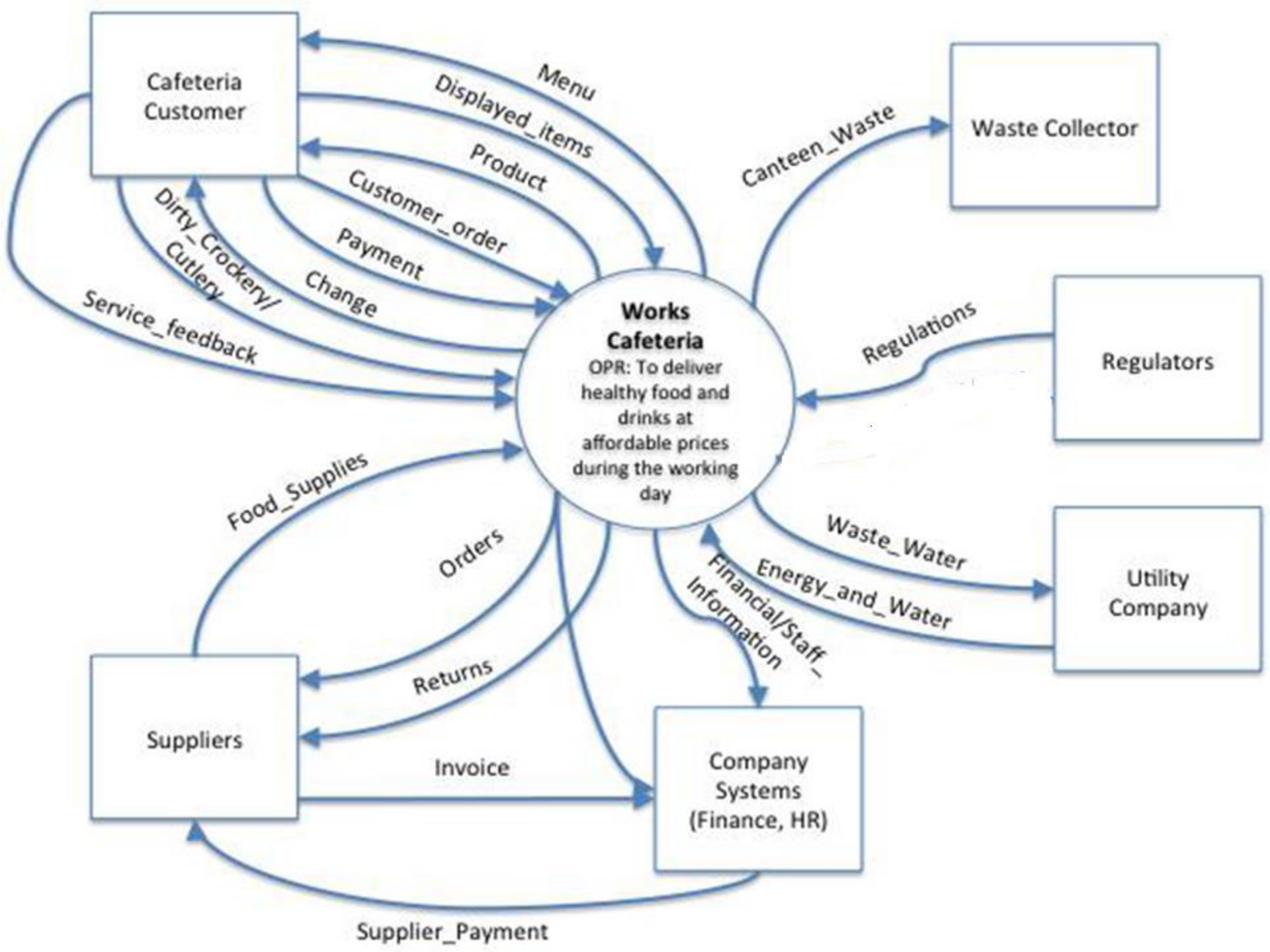
All Terminators at once

- Determine all possible terminators first as shown in Figure below.



Having identified the potential Terminators, each can be considered in turn to identify the potential flow interactions with the system of interest. The end result could look like Figure next.

Context Diagram for the Works Cafeteria


Cafeteria Customer
Menu
Displayed_items
Product
Customer_order
Payment
Change
Dirty_Crockery/ Cutlery
Service_feedback
Canteen_Waste
Waste Collector
Works Cafeteria
OPR: To deliver healthy food and drinks at affordable prices during the working day
Regulations
Regulators
Food_Supplies
Orders
Returns
Waste_Water
Energy_and_Water
Financial/Staff_ Information
Utility Company
Suppliers
Invoice
Company Systems (Finance, HR)
Supplier_Payment

The **strengths** of a Context Diagram include:

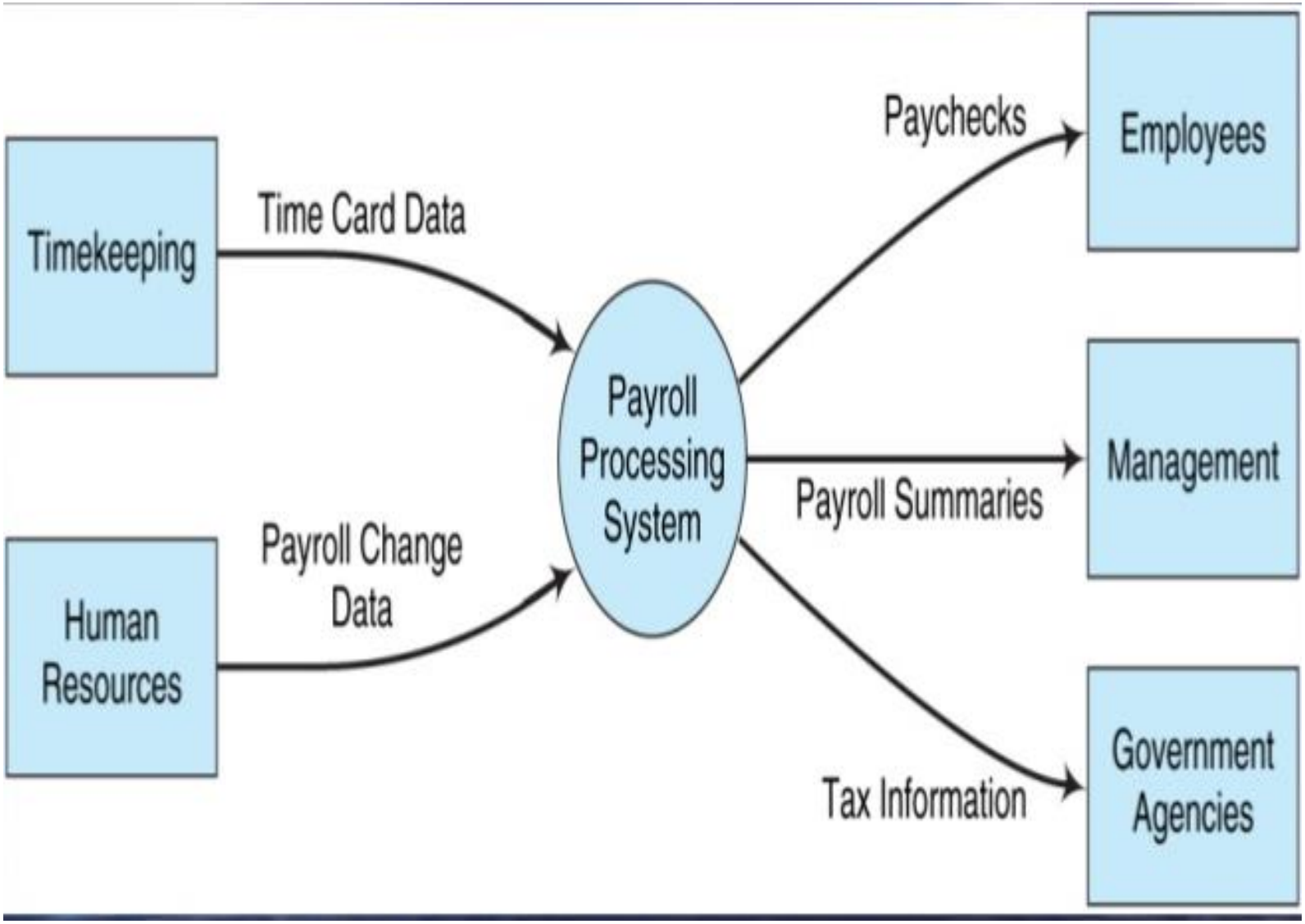
- Easy to comprehend
- Clear indication of system boundary
- Definition of external entities with which the System of Interest interacts

The **weaknesses** of a Context Diagram include

- No indication of the internal functionality

Draw a Context Diagram for Payroll Processing

It shows a Payroll Processing that interacts with 5 agents: Employee, Management, Government Agencies, Human Resources, Time keeping.


Timekeeping
Time Card Data
Human Resources
Payroll Change Data
Payroll Processing System
Paychecks
Employees
Payroll Summaries
Management
Tax Information
Government Agencies