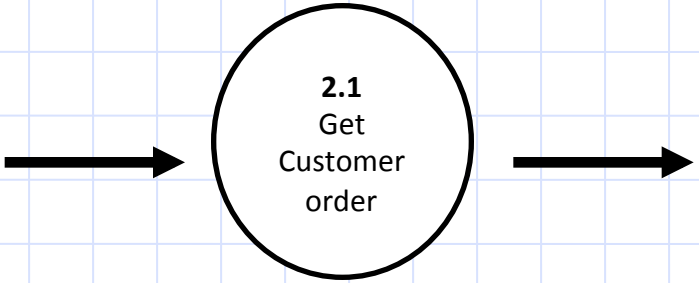
What is a Process Spec (PSPEC)

PSPEC

- Can be used during analysis phase and during design phase.
- During the system analysis the PSPEC can be used to describe the internals of each process.
  - inner workings of a process represented in a data flow diagram.
  - the input to a process and the algorithm
  - restrictions and limitations imposed on the process (function) performance
    - For complex processes you may add a state machine or even a flowchart to help.
- Process specs are normally provided for the low level processes (2^{nd} or 3^{rd} level)
- Process specifications are not needed for:
  - Simple input and/or output processes.
  - simple data validation.
  - Processes for which prewritten code already exists.

Process Specs Format During Analysis



- Process < Number> - Process Label
  - Preconditions:
    - What is required to invoke the process
  - Post condition:
    - What it does…

Process 2.1: Get Customer Order

Description:

- This process provides a client side process to deliver the customer order to the web and database server.

Preconditions:

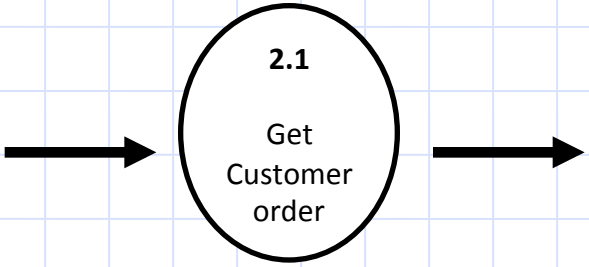
- Needs valid customer information and valid order items

Post conditions:

- store the order

SPEC's Format During Design

- For low-level processes
- Format:
  - Process < Number> - Process Label
    - Preconditions:
      - What is required to invoke the process
    - Postposition:
      - What it does… (more implementation details)
    - Possibly reference to a state transition diagram
    - Possibly reference to a flowchart



Design PSPEC

Process 1: Get Customer Order
Preconditions:
- Needs valid customer information
- Needs valid order items

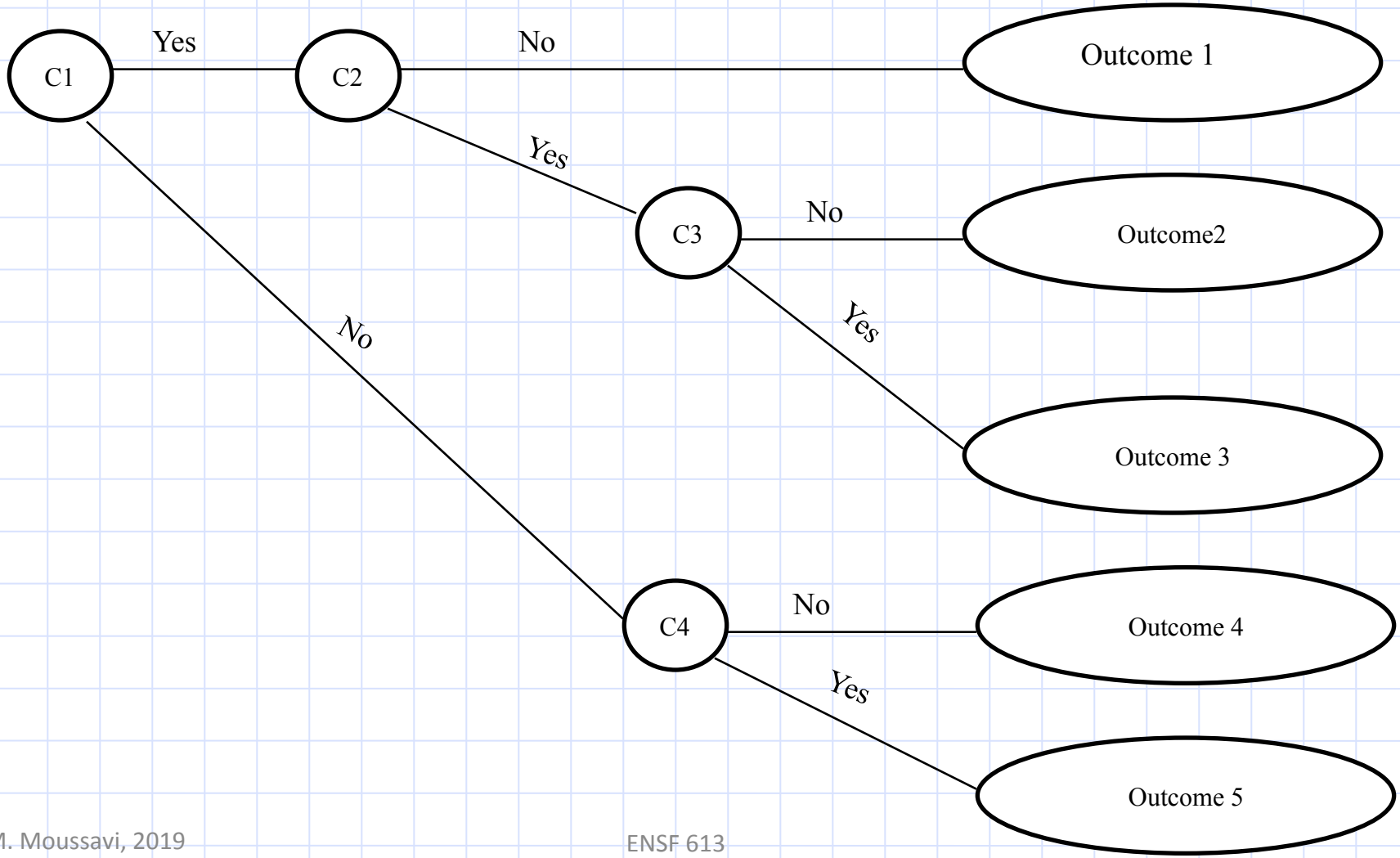
Post conditions:
- If valid order
  - If valid customer info
    - Save into order database table
- Else
  - Display the message, and …
- Promises to store the order
- STD reference #
- Flowchart reference #

Supporting Artifact

- C-specs document can be supported by :
  - Decision Tables.
  - Decision trees.
  - State Transition Diagrams (STDs)
  - Structured English
- Use a combination of tools based on:
  - Client / analyst preference.
  - Nature of the process.

Decision Trees

- A graphical representation of a decision situation using:
  - Decision points represented by circles, actions by ovals, and connections between decision points and actions by lines with values for conditions.



Decision Tables

- Decision tables might be used for complex processes
- A matrix representing the logic of a complex decision.
- Inputs as columns, actions (outputs) as rows.

conditions	Domestic?	Y	Y	Y	Y	N	N	N	N
	≥half full?	Y	Y	N	N	Y	Y	N	N
	≥$350/seat	Y	N	Y	N	Y	N	Y	N
outcomes	Serve cocktails?	✓	✓	✓	?	✓	?	?	?
	Free cocktails?					✓			

Structured English

- Modified form of the English language used to specify the logic of information systems processes.
- A subset of English which includes:
  - Verbs such as READ, WRITE, ADD, SUBTRACT, etc.
  - Noun phrases to represent data and data structures (defined in the data dictionary).
- Does not include adverbs and adjectives.
- Usually 40 to 50 verbs.
- No standard version

Structured English Example

Process 2.3: Borrow a Book

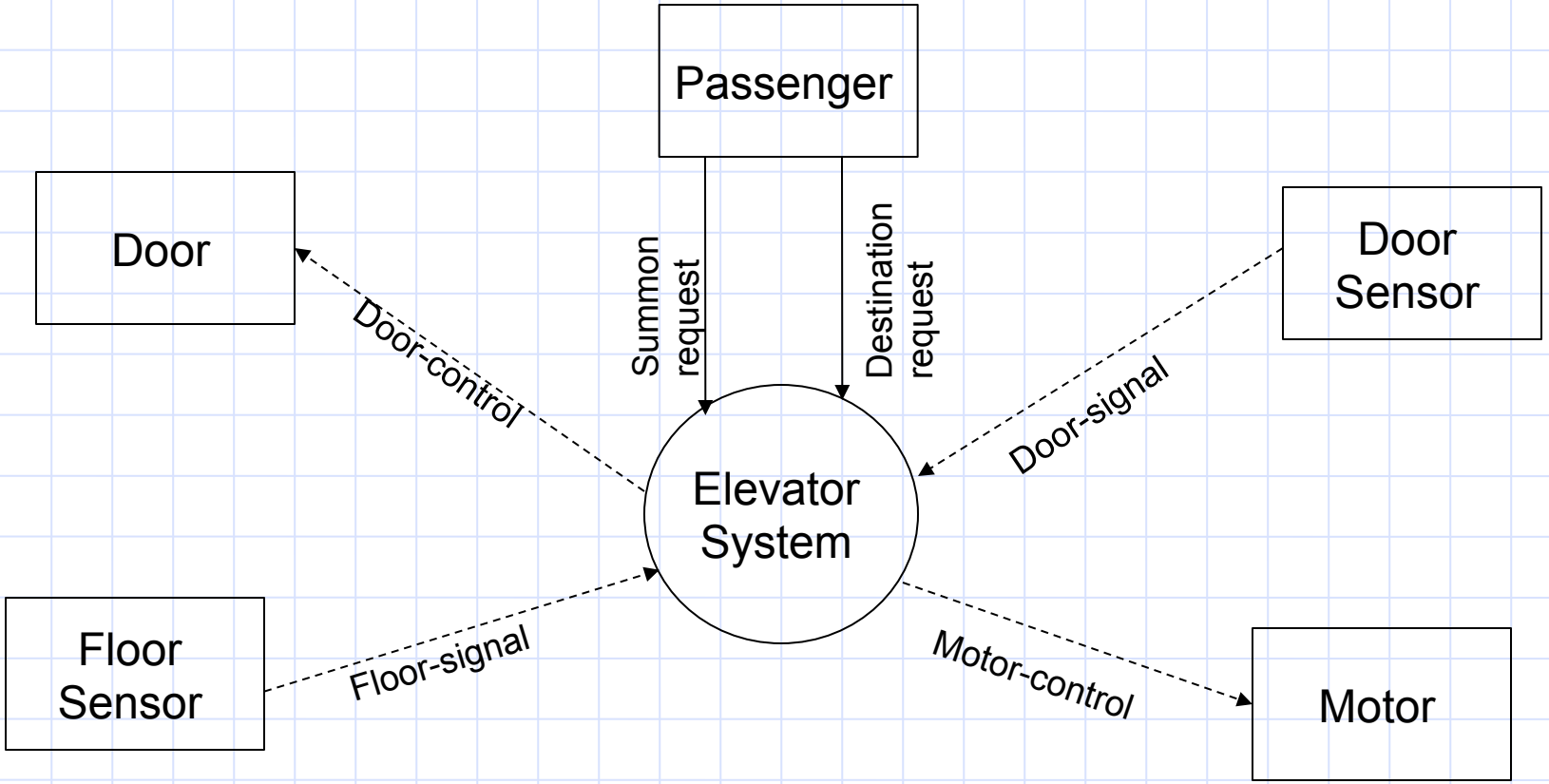
Remarks: This process allows patrons to borrow a book by a in a library system.
IF user is a **PATRON**
IF PATRON has no outstanding fines
-- Loan a **Book**
ELSE IF pays his fines
Issue **Receipt**
Loan a **Book**
ELSE
Suspend the Card
Reject loan
Else If has a valid Credit Card
Take Deposit
Loan Book
Else
Reject Loan

Appendices

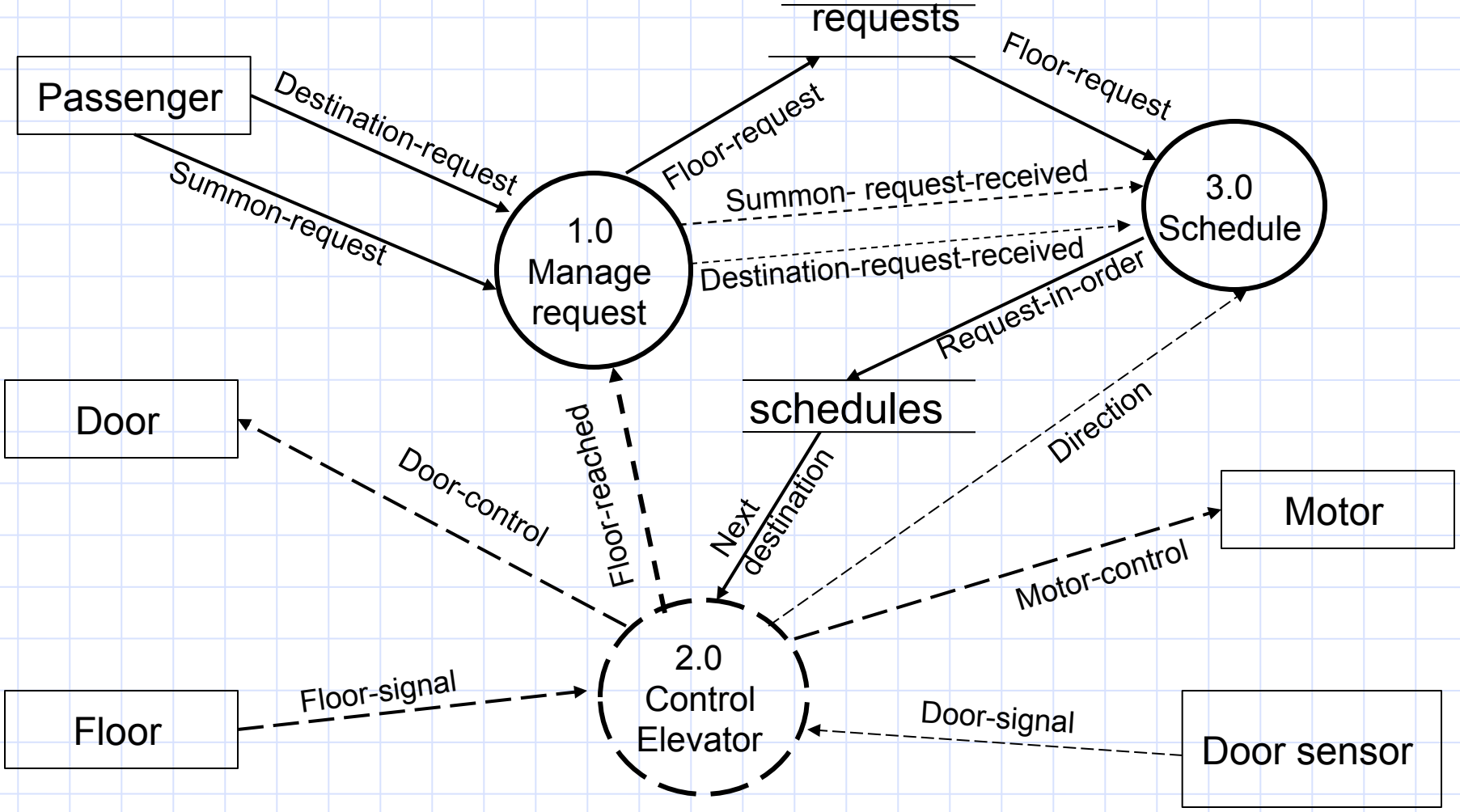
- Structured Analysis & Structured Design

A Single Tower Elevator System

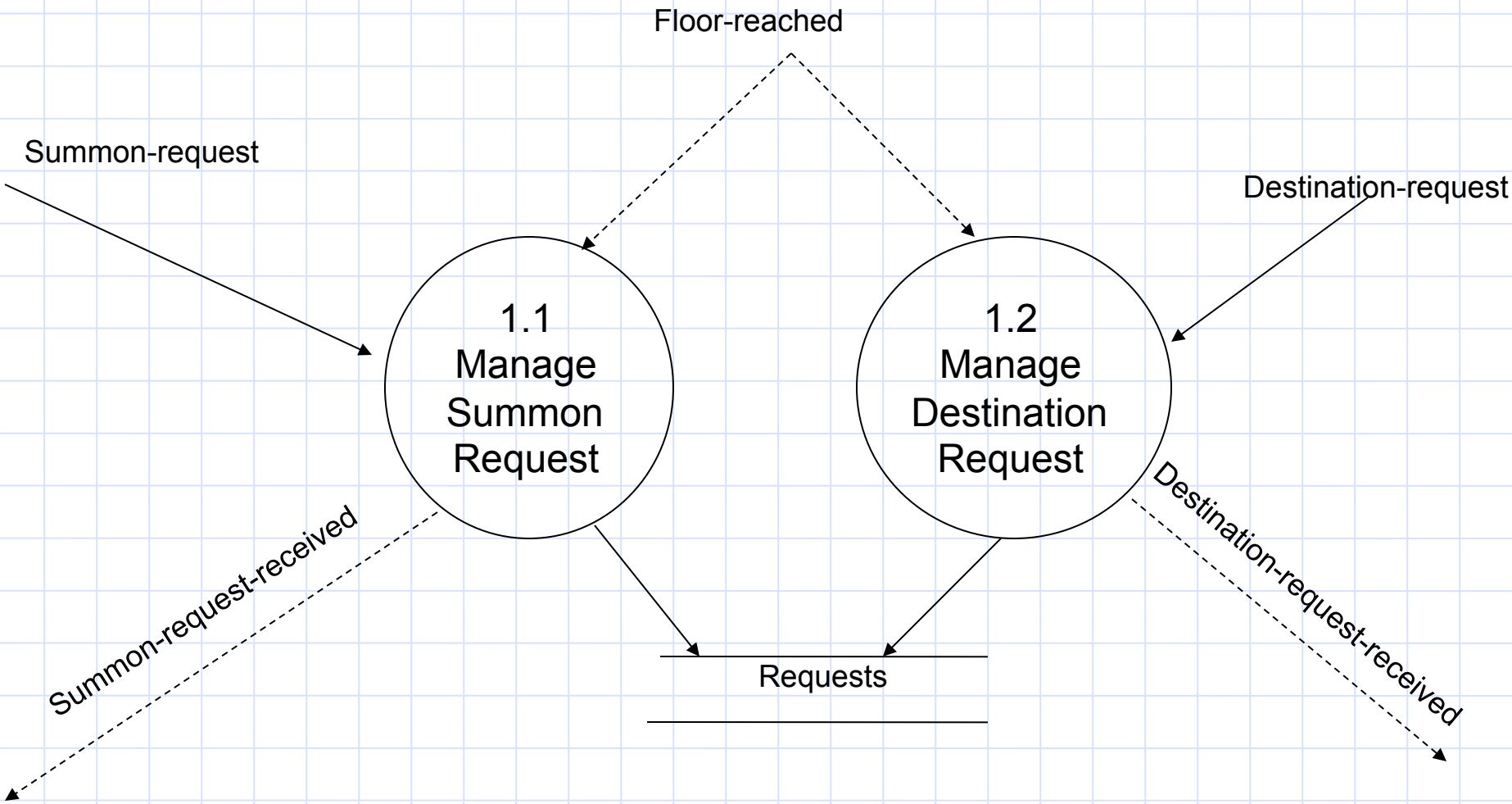
Simplified Context Diagram for Elevator System



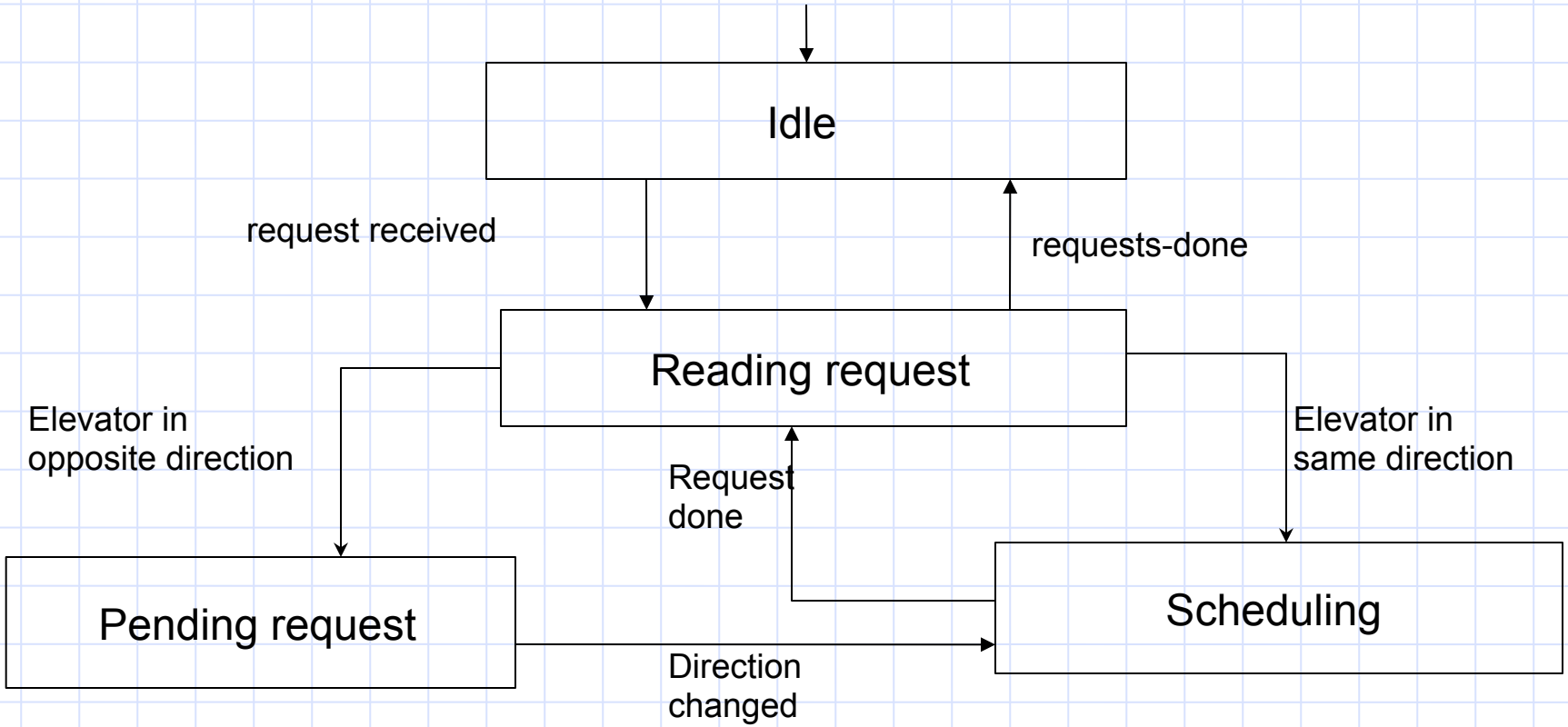
Simplified Level-1 for Elevator System



Level-2 for Control Manage Request



STD for Schedule Process



Process Spec for Schedule Process

- BEGIN
- with summon/destination request received
- DO WHILE request exists
- IF elevator in-direction
- Move elevator to destination
- Clear request
- ELSE
- Pending the request
- ENDIF
- END DO
- END

Example of Data Dictionary for Elevator System

- Destination request = Floor number
- Direction = [up | down]
- Floor number = 1{legal digits}4
- Floor reached = signal
- Motor control = [on | off] + direction
- Requests = summon request + destination requests.
- Summon request = Floor number

Structured Analysis & Structured Design
AcmePress Case Study

The Statement of Purpose

- The purpose of AcmePress system is to maintain information needed to sell books to customers, such as:
  - Customers book orders.
  - Inventory control
  - Producing invoices
  - Producing reports

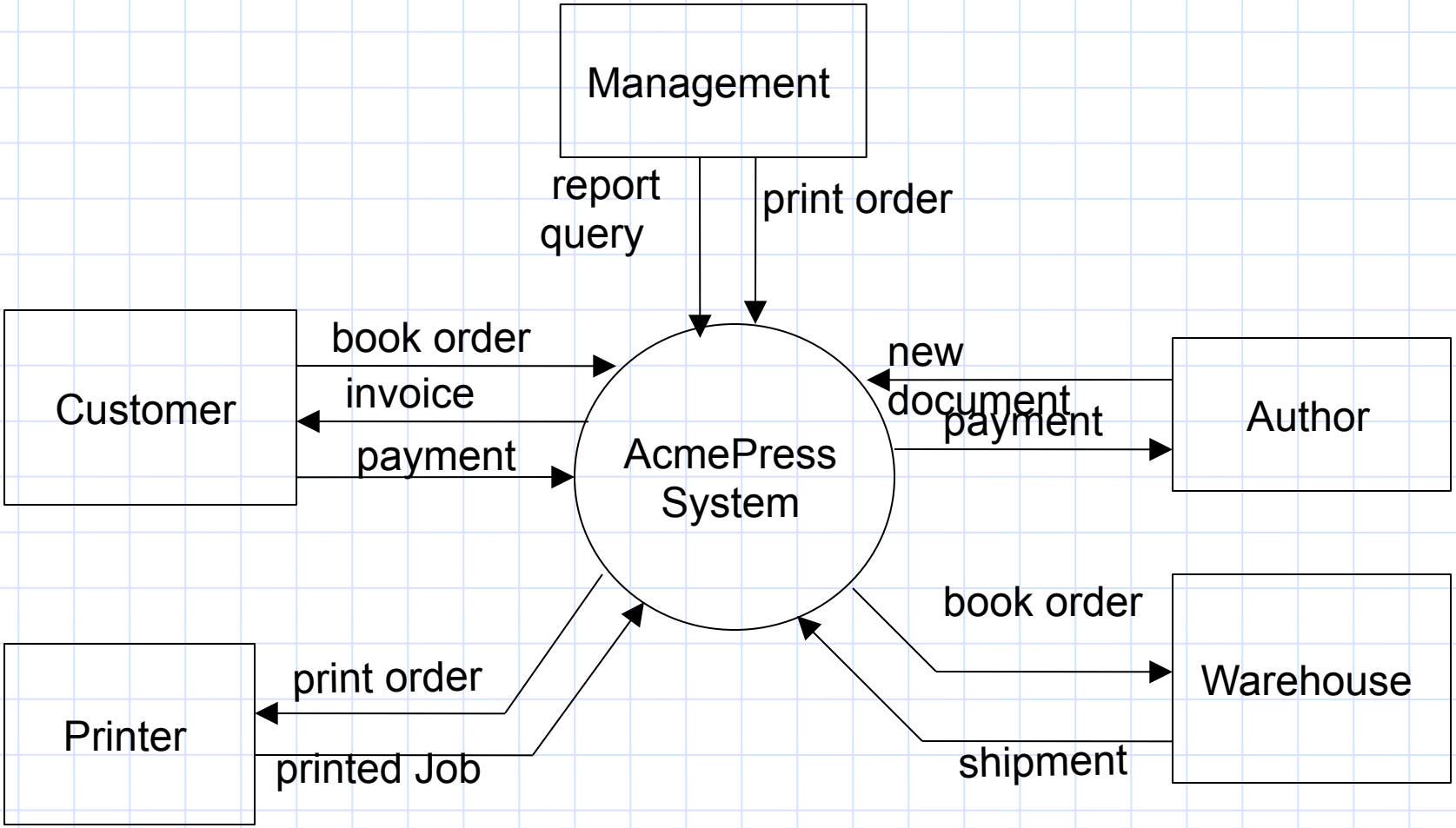
Event List

- Customer orders book.
- Customer sends payment.
- Customer receives invoice
- Management authorizes a print order.
- Printing Department receives print orders.
- Printing Department sends the printed jobs to the Warehouse

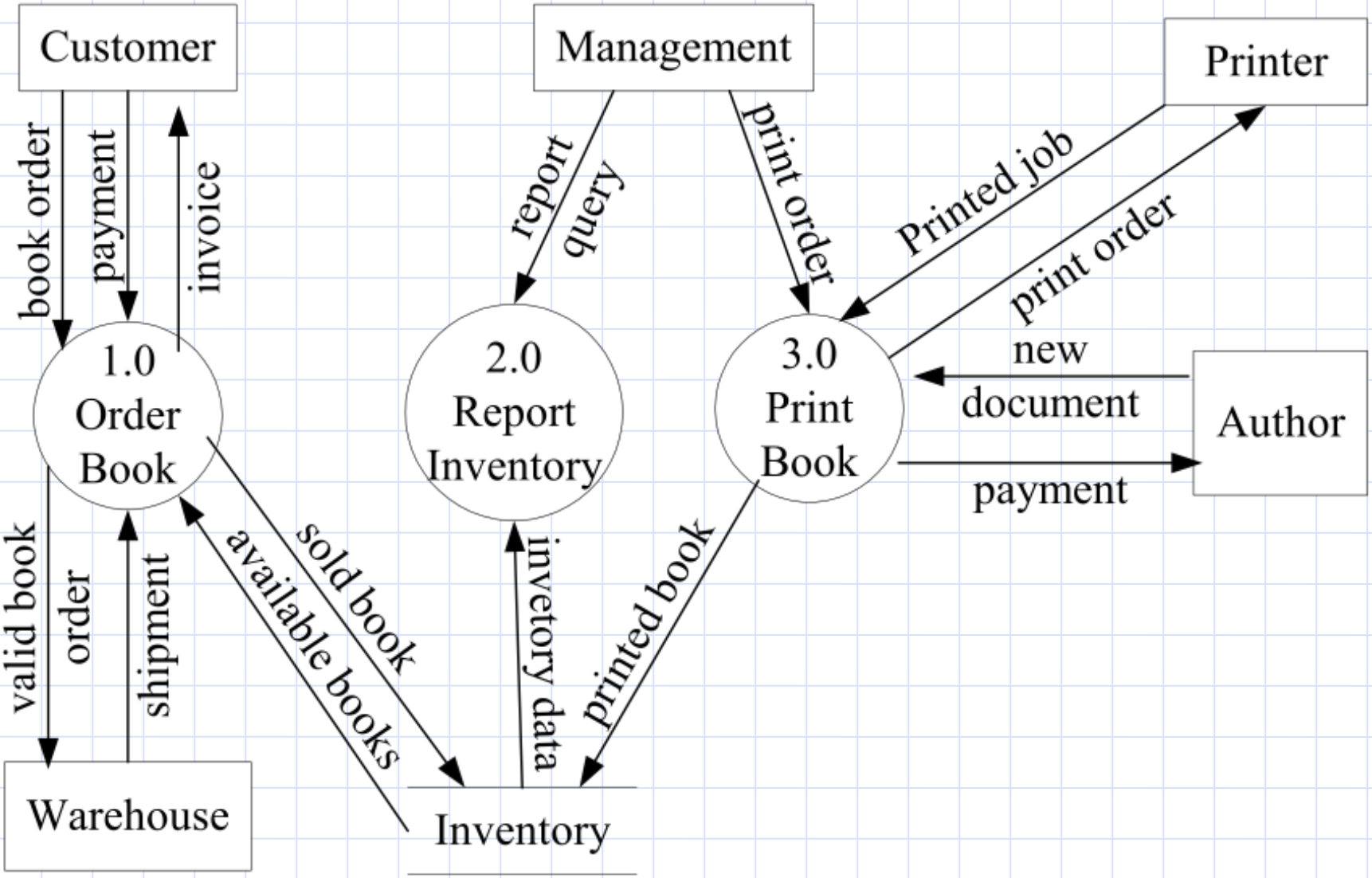
Event List (continued)

- Warehouse receives order
- Warehouse makes shipment of book.
- Author submits new document to be published.
- Author receives payment
- Warehouse conducts monthly inventory
- Management request quarterly report

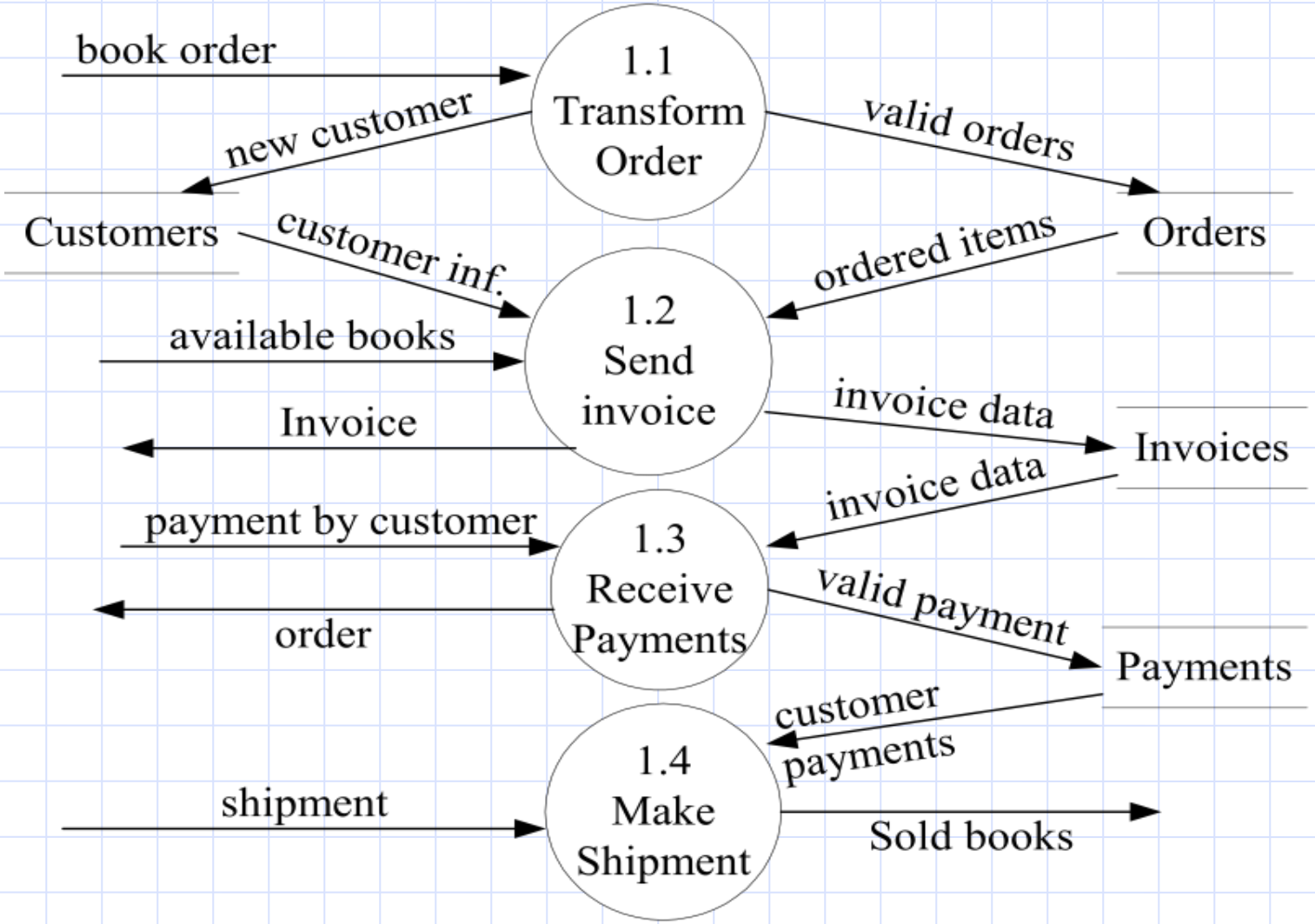
Context Diagram



DFD Level-1


Customer
Management
Printer
book order
payment
invoice
report
query
print order
Printed job
print order
1.0
Order
Book
2.0
Report
Inventory
3.0
Print
Book
new
document
Author
payment
valid book
order
shipment
available books
sold book
invetory data
printed book
Warehouse
Inventory

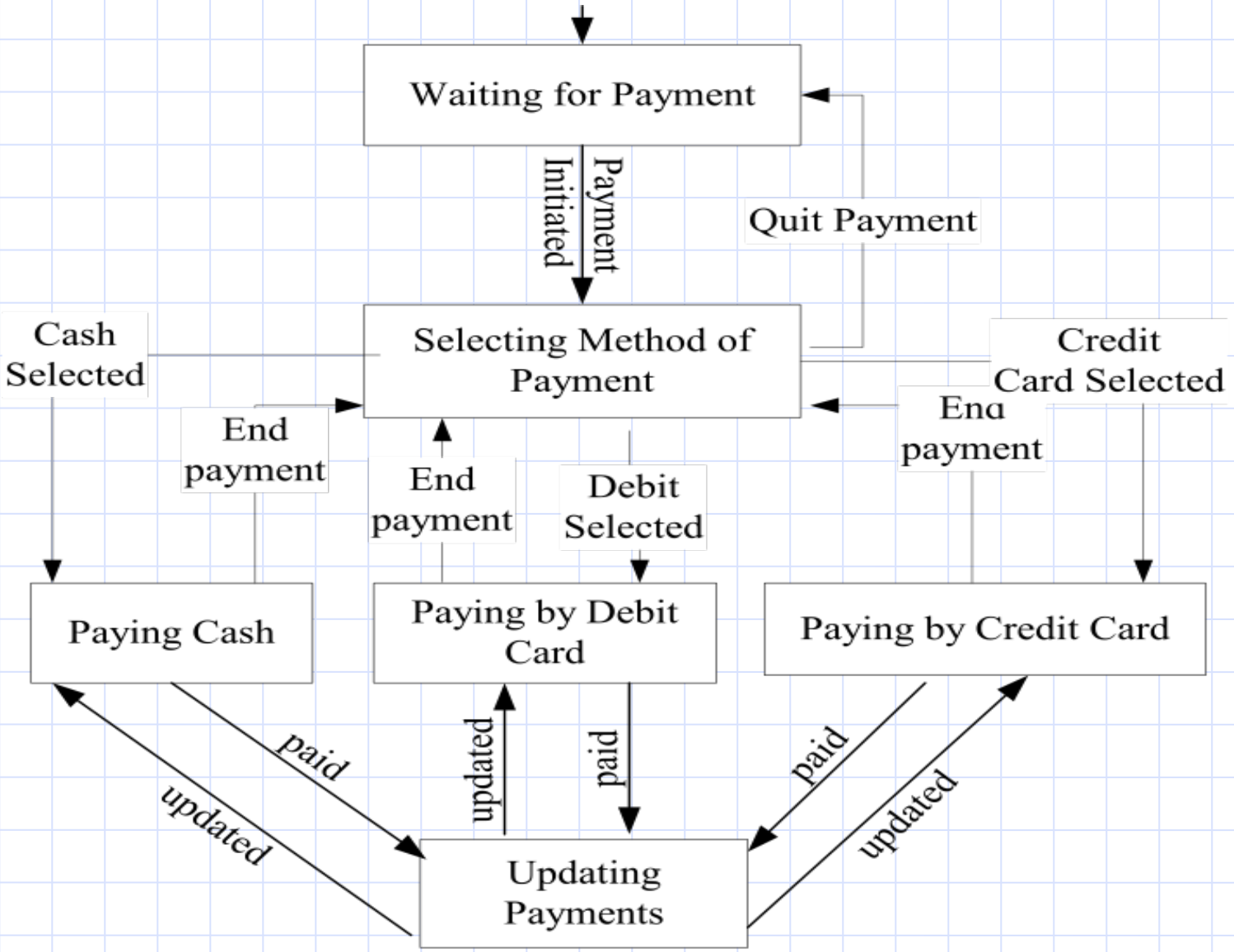
DFD Level 2 for Order Book



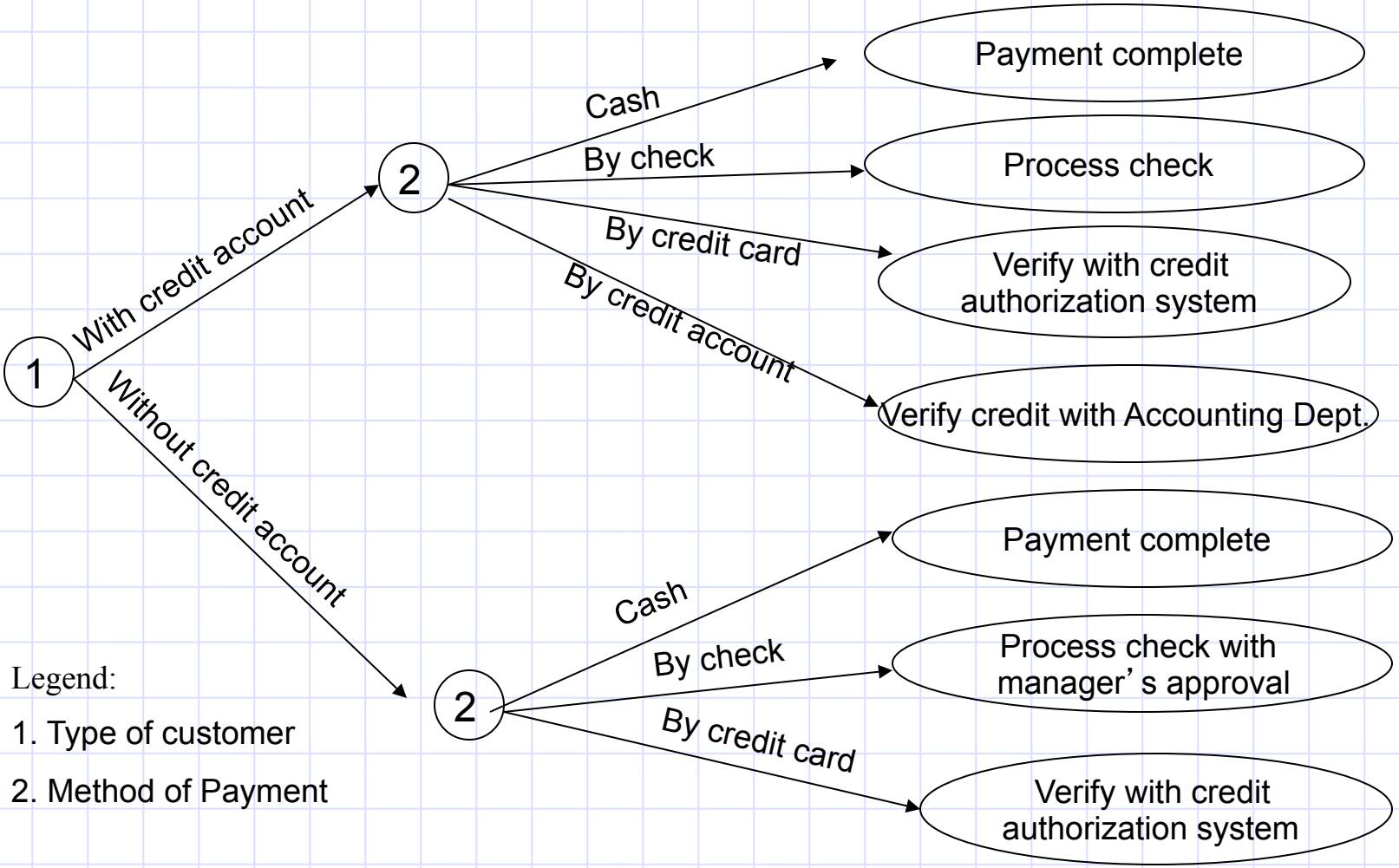
Process Spec for Receive Payment

- DO While there are more payments
  - READ the payment amount
  - SELECT the invoice
  - READ the invoice
  - DISPLAY the invoice
  - UPDATE customers account in PAYMENTS
- END DO

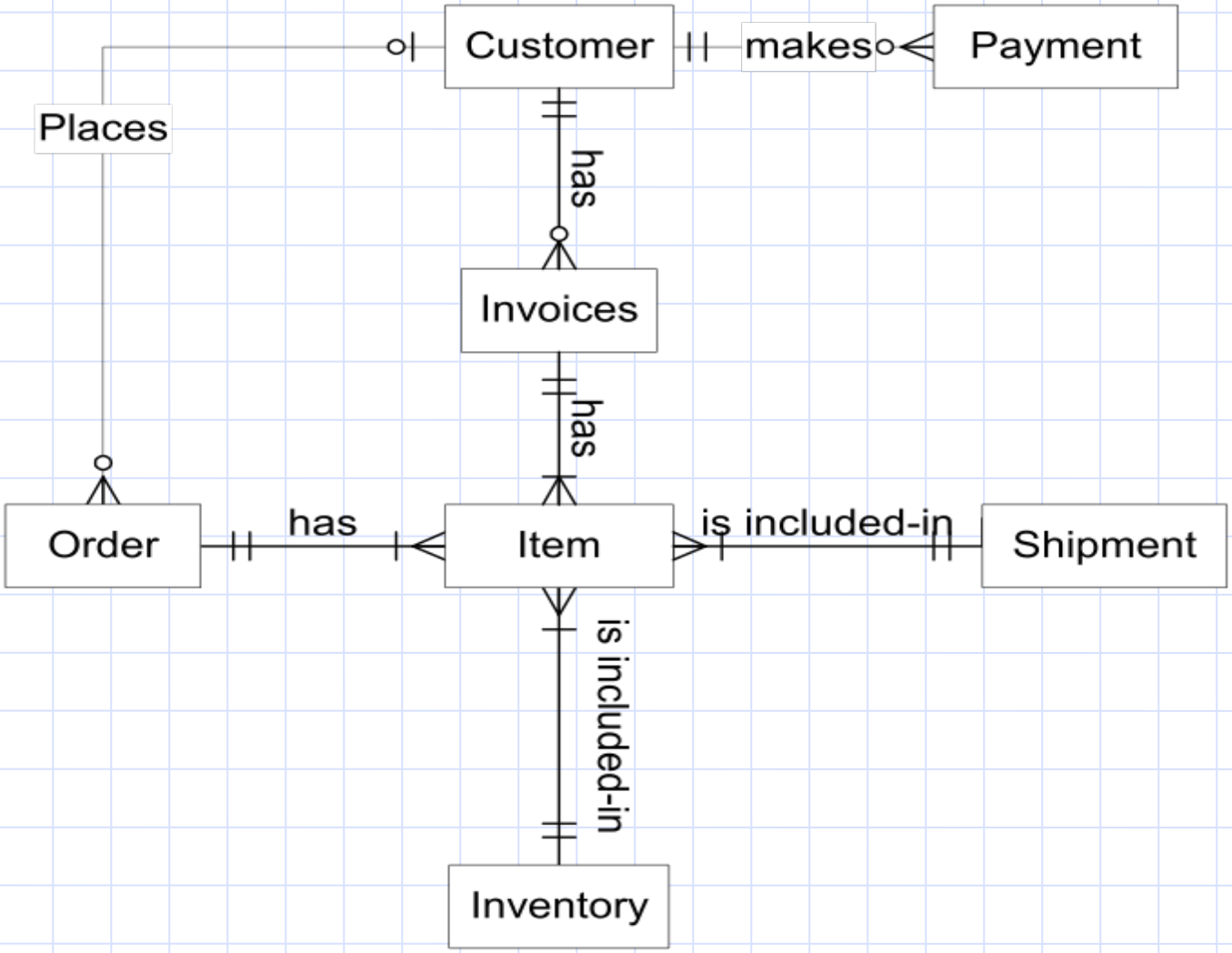
STD for Receive Payments



Decision Tree for Receive Payments



Partial ERD


Customer
makes
Payment
Places
has
Invoices
has
Order
has
Item
is included-in
Shipment
is included-in
Inventory

Data Dictionary

- Customer name = courtesy-title + first name + last name.
- First name 1 {legal characters} 20
- Order * a book order received from a customer*
  - » = @ order number + Customer name + Customer ID + Customer address + {item} + order date + shipping charges + total amount.
- Orders * a collection of book orders*
- = {book order}