TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

Systems analysis activities
Modeling requirements

Viet-Trung Tran
trungtv.github.io
School of Information and Communication Technology

Outline

- Use cases
- Types of events
- Identifying events
- Use case descriptions
- “Things” in the system problem domain

Use cases

- Use cases is the key to defining **functional requirements**
- Use case is an activity that the system performs in response to a request by the user (or an agent)
- Two techniques for developing use cases
  - User goal technique
  - Event decomposition technique

Use case examples

Use Case	Description
Look up supplier	Using supplier name, find supplier information and contacts
Enter/update supplier information	Enter (new) or update (existing) supplier information
Look up contact	Using contact name, find contact information
Enter/update contact information	Enter (new) or update (existing) contact information
Look up product information	Using description or supplier name, look up product information
Enter/update product information	Enter (new) or update (existing) product information
Upload product image	Upload images of the merchandise product

User goal technique

- Identify all users
- Classify all users in terms of their functional role
- Classify all users by organizational level
- Interview all users (or all types)
  - what are their specific goals?
  - frame all in imperative verb-noun format
    - Update order
    - Add customer
- Create a list of use cases, organized by type of user
- Look for duplicates, resolve inconsistencies
- Identify where different users have the same use case
- Review updated list with users and stakeholders

Event decomposition technique

- Identify use cases based on business events the system must handle and respond to
- Easier said than done
  - Identifying the events that are associated with use cases ***at the right level of detail is difficult***
- Example
  - enter a name on a form
  - add a new customer to the system
  - add customers and update customer records

Which is best level of detail?

Event decomposition

- The right level of detail corresponds to a task
  - That is performed by one person
  - adds measurable business value
  - leaves the system and data in a stable state
- We call this: Elementary Business Process (EBP)
  - Fill a shopping cart: Is this a EBP?
- Ask ourselves
  - What business events happen that the system will need to respond to?

Types of events

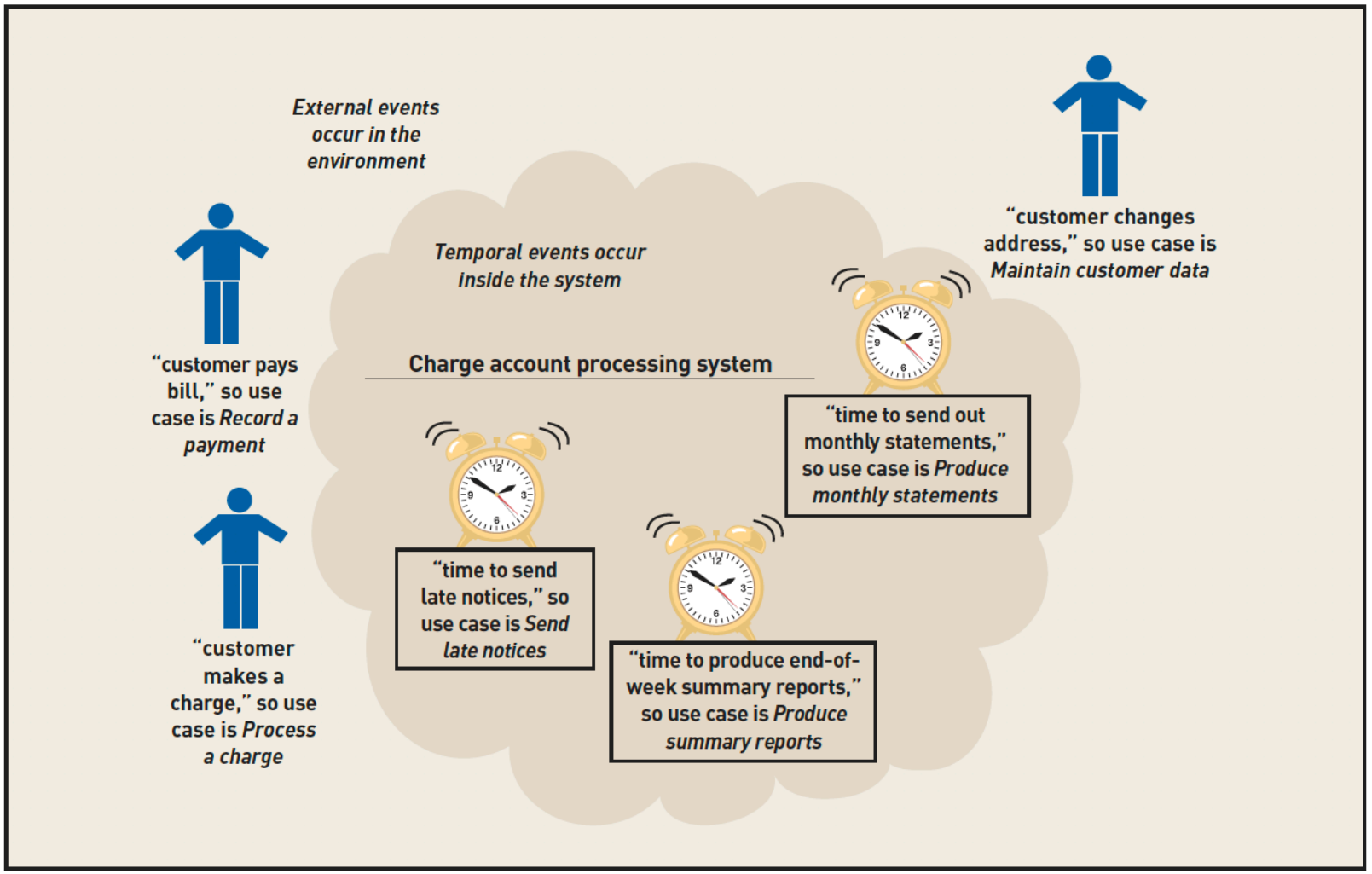
- External events
  - can also be triggered by employees or departments
  - Checklist
    - external agent wants something, that results in a transaction
    - external agent wants some information
    - data changed, and needs to be updated
    - management wants some information Temporal Events State Events
- External events to look for include:
  - External agent wants something resulting in a transaction
  - External agent wants some information
  - Data changed and needs to be updated
  - Management wants some information

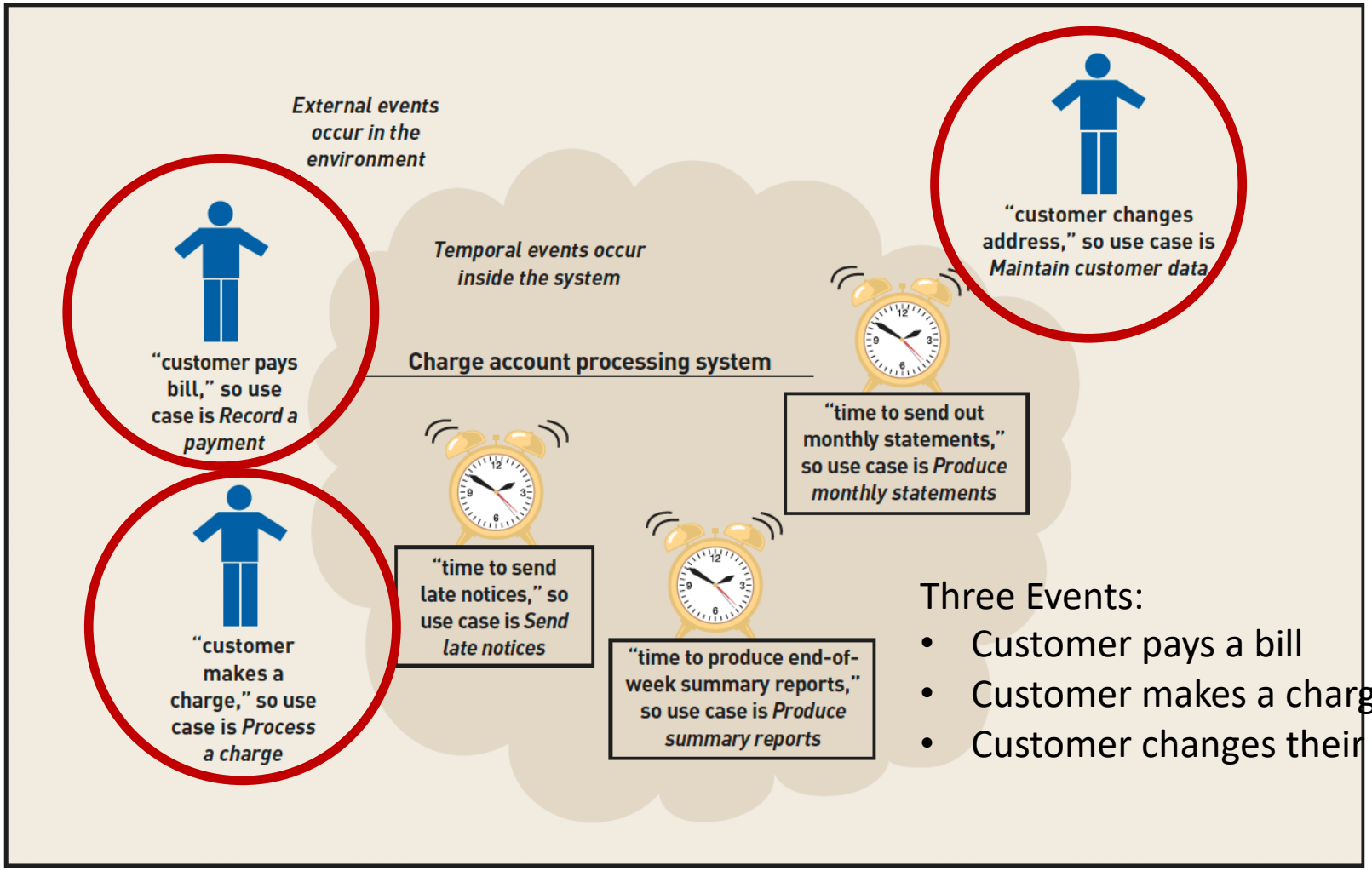
Temporal events

- An event that occurs as a result of reaching a point in time
- Ask:
  - What deadlines occur?
  - What outputs are associated with those deadlines?
- Temporal events to look for include:
  - Internal outputs needed
  - Management reports (summary or exception)
  - Operational reports (detailed transactions)
  - Internal statements and documents (including payroll)
  - External outputs needed
  - Statements, status reports, bills, reminders

State events

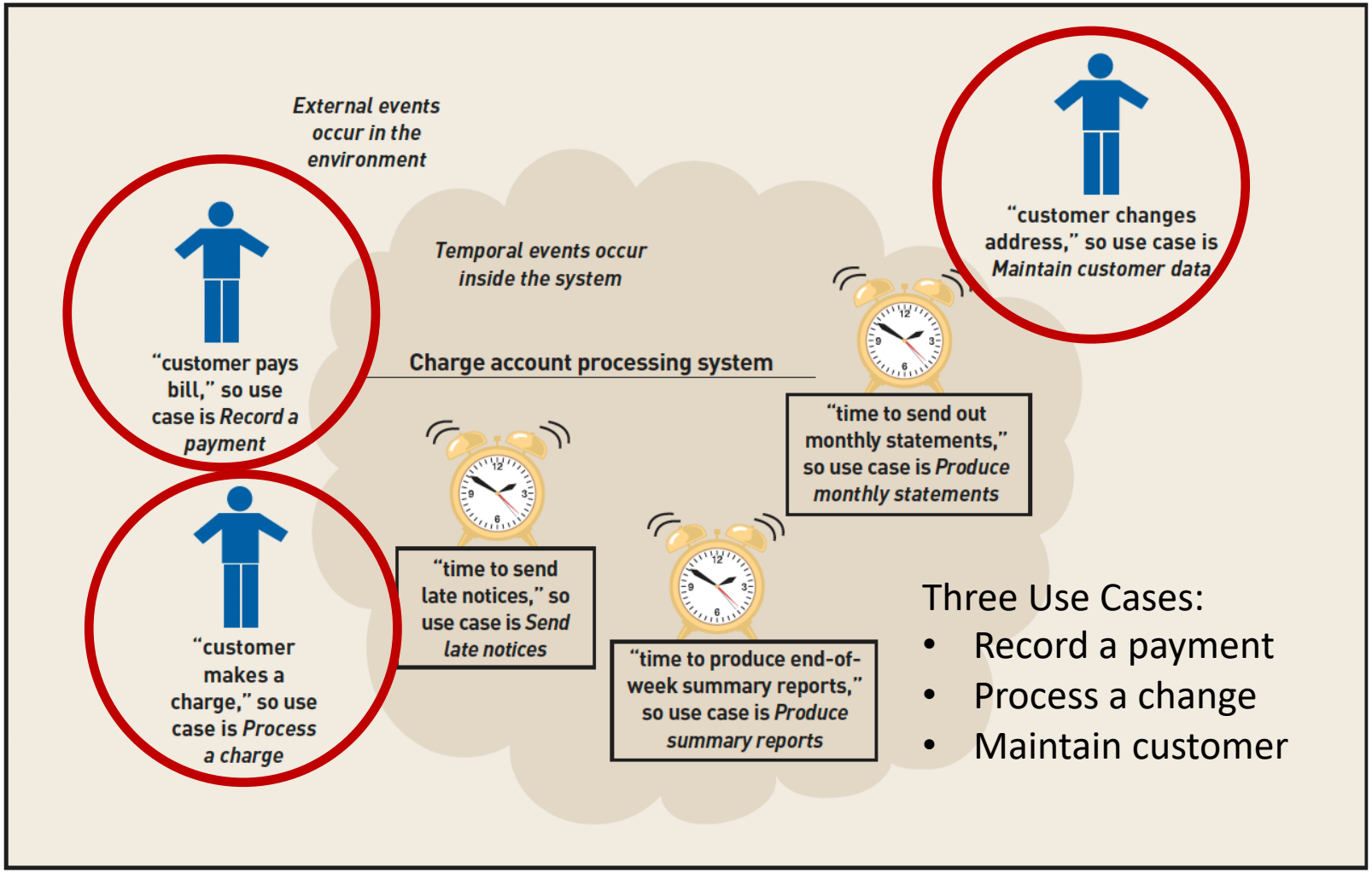
- an event that occurs when something happens inside the system that triggers the need for processing (also called internal events)

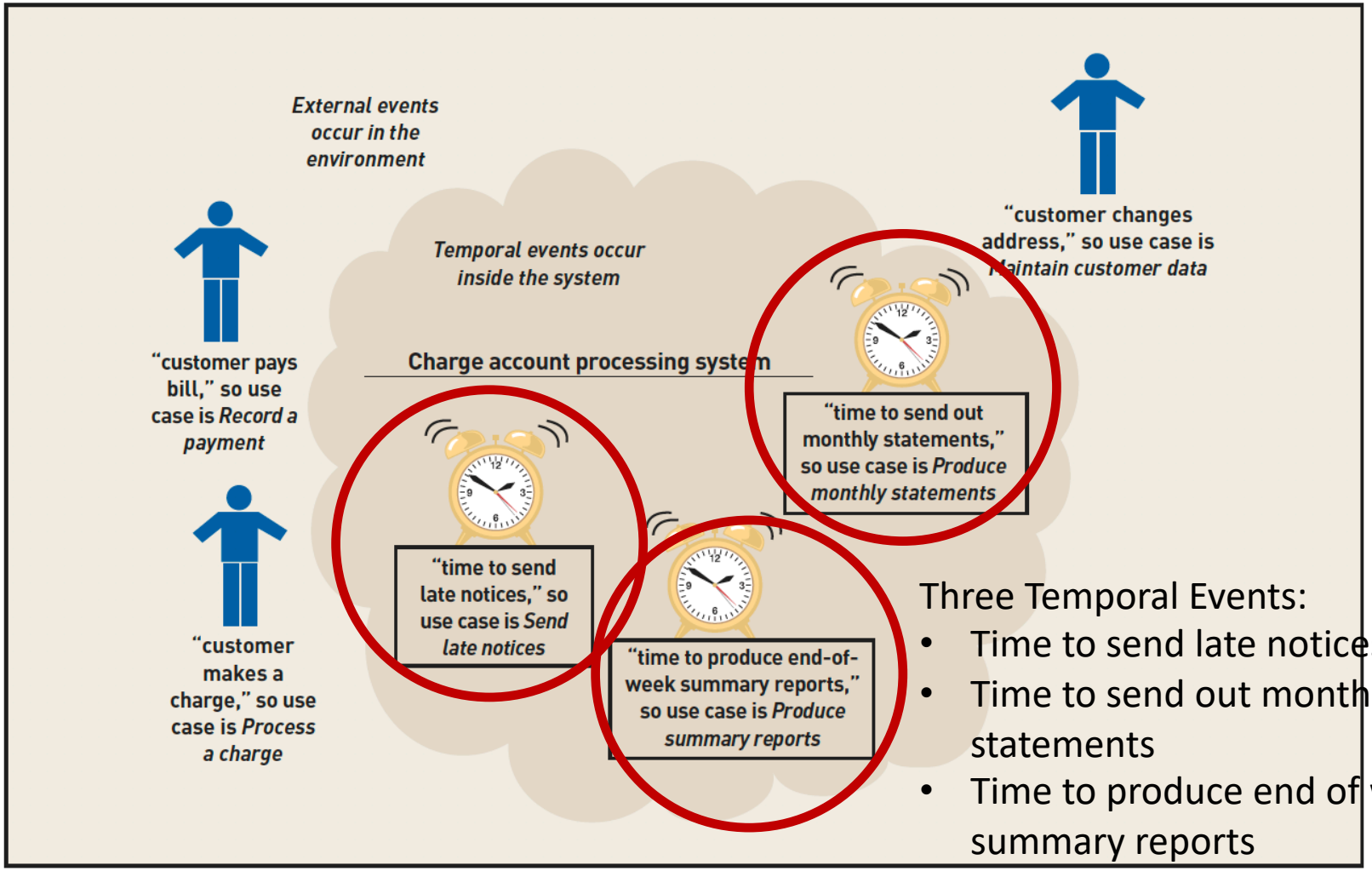

External events occur in the environment
"customer changes address," so use case is Maintain customer data
Temporal events occur inside the system
"customer pays bill," so use case is Record a payment
Charge account processing system
"time to send out monthly statements," so use case is Produce monthly statements
"time to send late notices," so use case is Send late notices
"customer makes a charge," so use case is Process a charge
"time to produce end-of-week summary reports," so use case is Produce summary reports



Three Events:

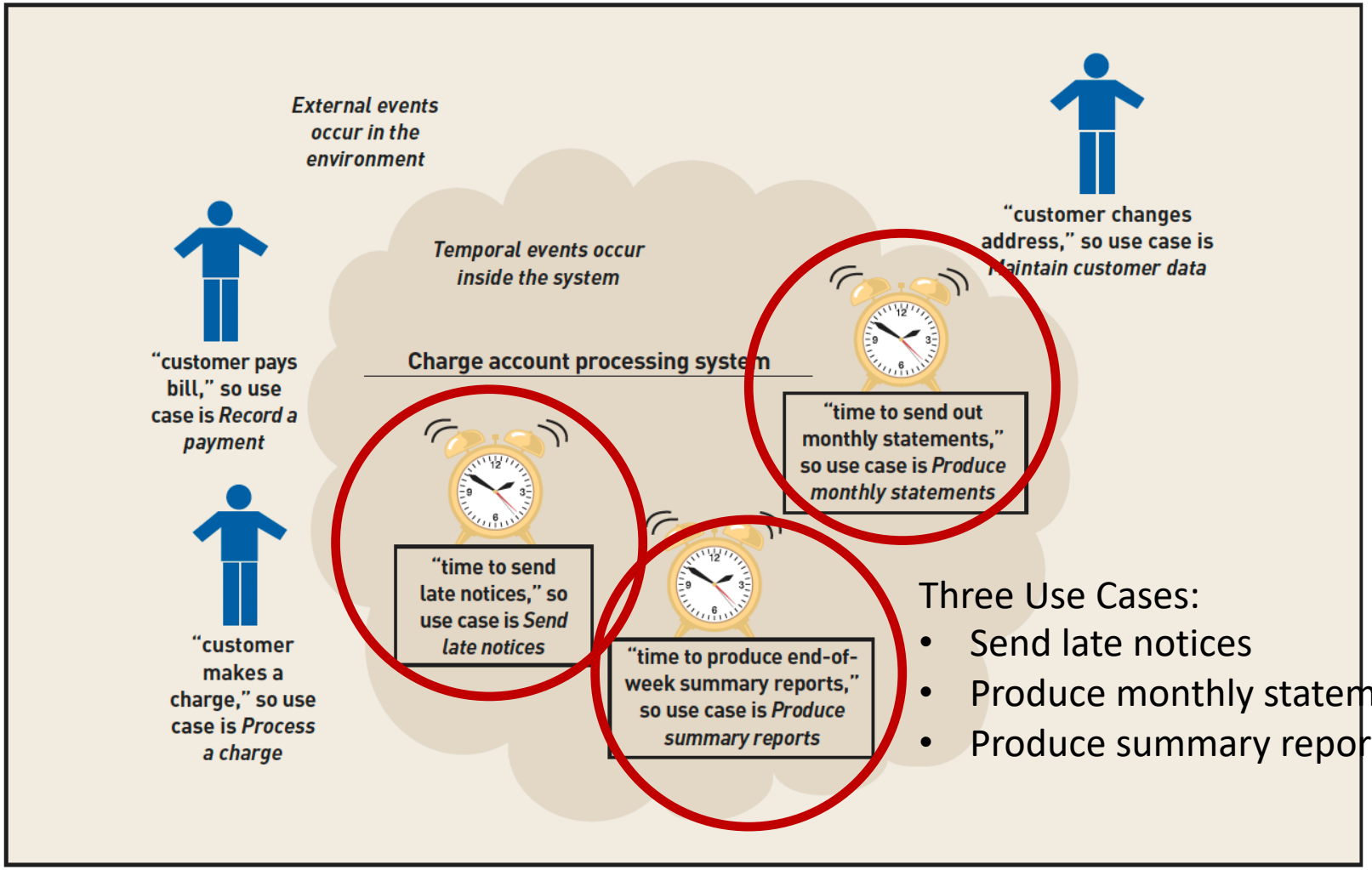
- Customer pays a bill
- Customer makes a charge
- Customer changes their address


External events occur in the environment
Temporal events occur inside the system
Charge account processing system
"customer pays bill," so use case is Record a payment
"customer makes a charge," so use case is Process a charge
"customer changes address," so use case is Maintain customer data
"time to send out monthly statements," so use case is Produce monthly statements
"time to send late notices," so use case is Send late notices
"time to produce end-of-week summary reports," so use case is Produce summary reports
Three Use Cases:
• Record a payment
• Process a change
• Maintain customer



Three Temporal Events:

- Time to send late notices
- Time to send out monthly statements
- Time to produce end of week summary reports

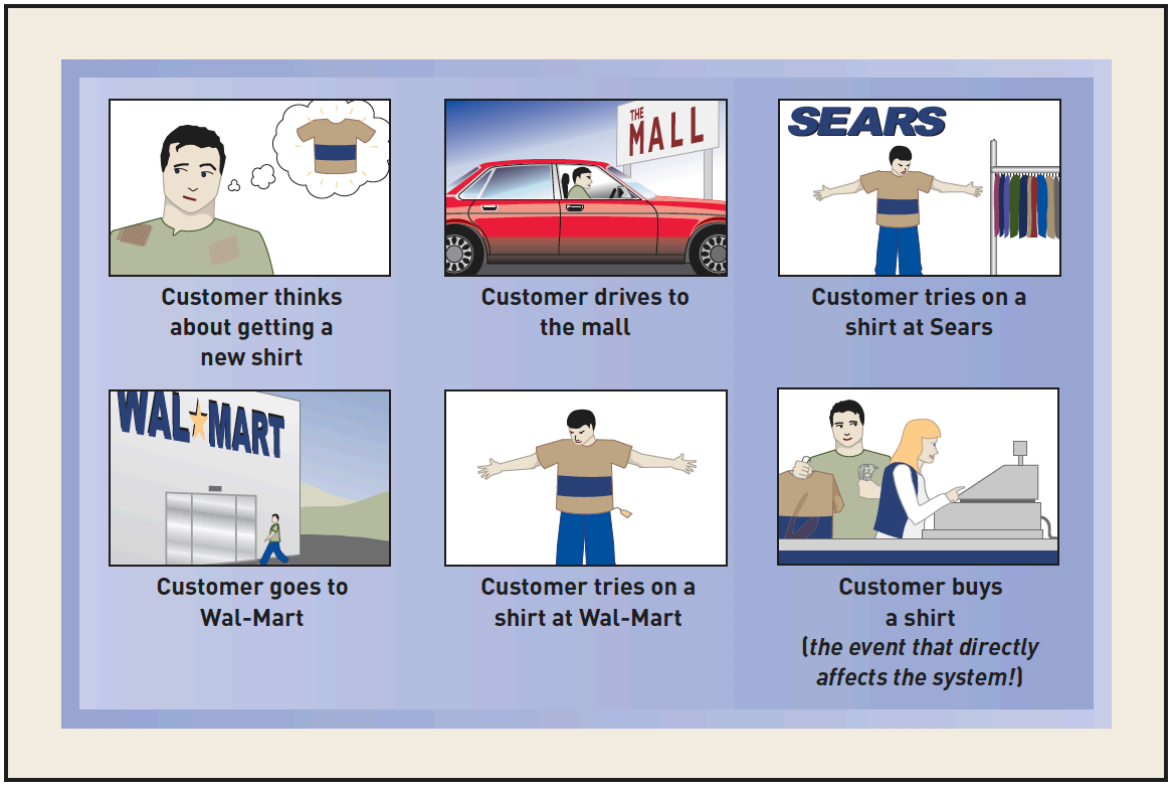


Three Use Cases:

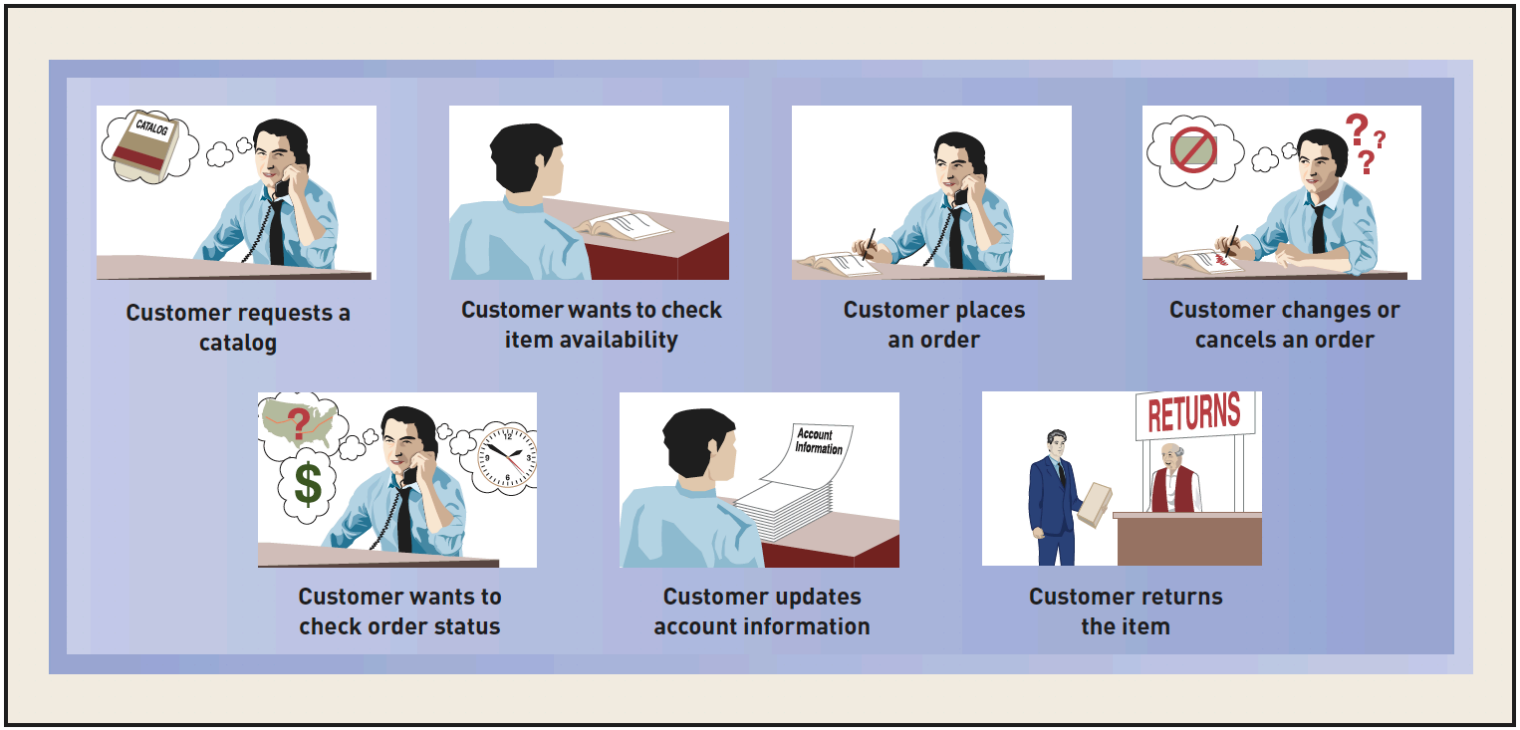
- Send late notices
- Produce monthly statements
- Produce summary reports

Identifying events

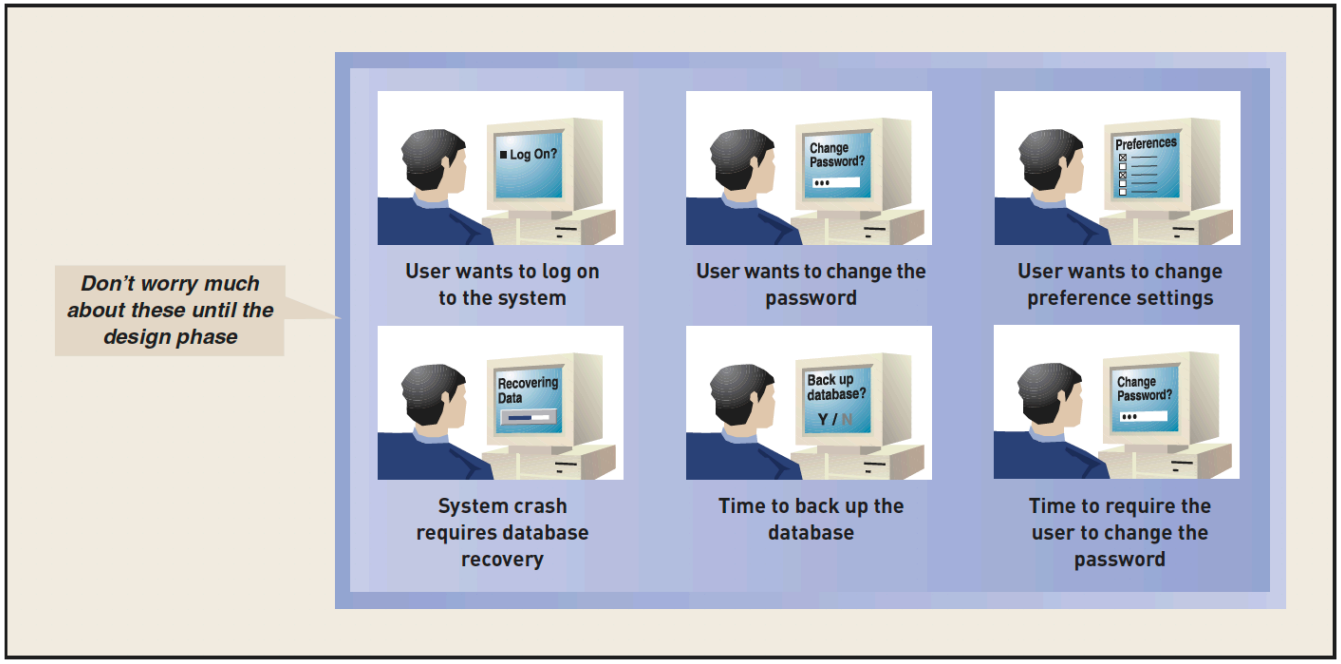
- Sequence of actions that lead up to only one event affecting the system



- The sequence of “transactions” for one specific customer resulting in many events



- Events deferred until design, ignore these at start of project
  - System Controls
    - Checks
    - Safety procedures
    - Protection of integrity of system
  - Technology-dependent Events



Summary of decomposition steps

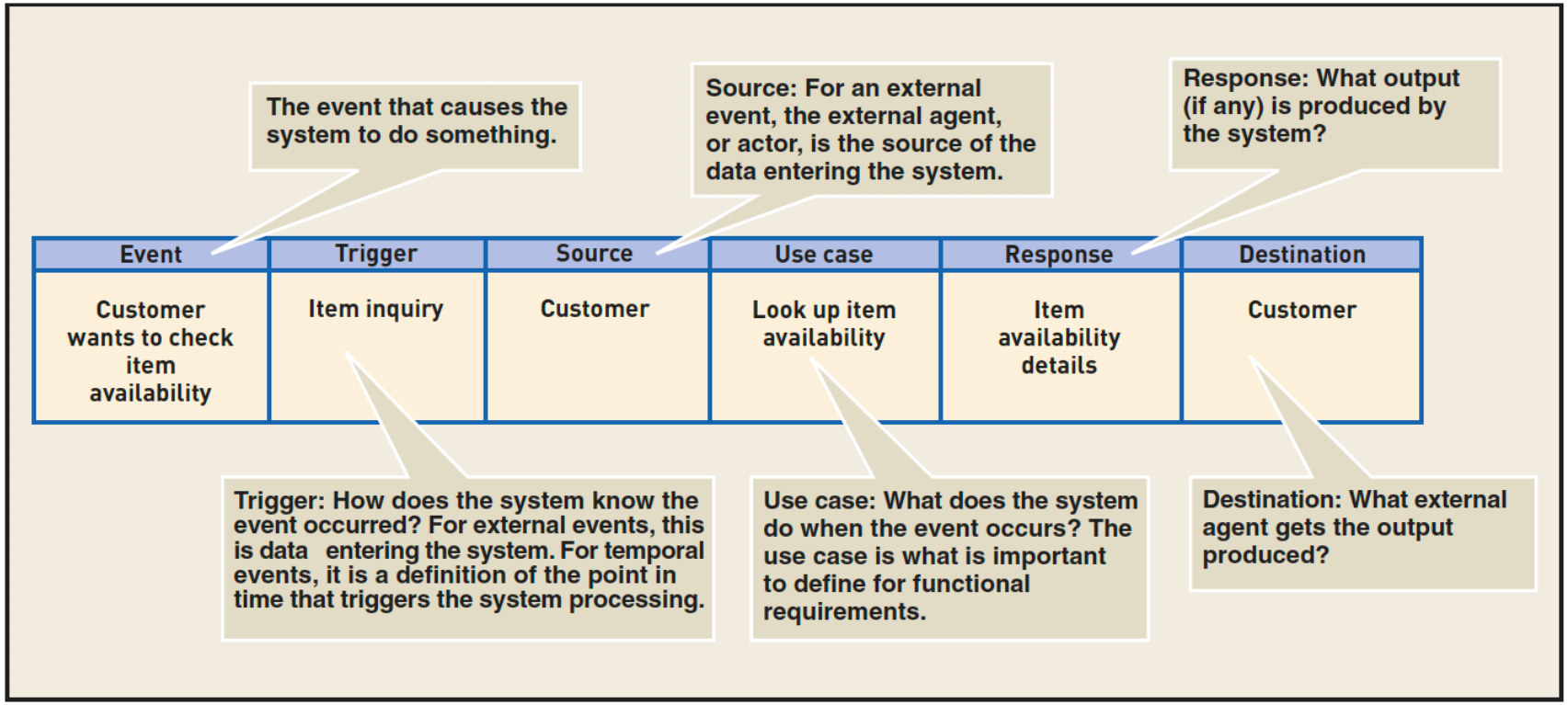
- Identify external events that require response from the system
- For each event, identify and name a use case
- Identify the temporal events that need a system response
- For each event, identify and name a use case the system requires
  - then identify the point in time that triggers the use case
- Identify the state events the system might respond to
- For each state event, identify and name the use case the system needs and the state change
- Assuming perfect technology, check each use case is needed, and ignore system controls for now

Naming events

- Should include
  - agent is named
  - identify the action
- Example
  - customer *places an order*

Event table

- Use an event table as a catalog of information about the use cases that make up the functional requirements of the system.



Example: Customer support system event table

Customer support system event table					
Event	**Trigger**	**Source**	**Use case**	**Response**	**Destination**
1. Customer wants to check item availability	Item inquiry	Customer	Look up item availability	Item availability details	Customer
2. Customer places an order	New order	Customer	Create new order	Real-time link Order confirmation Order details Transaction	Credit bureau Customer Shipping Bank
3. Customer changes or cancels order	Order change request	Customer	Update order	Change confirmation Order change details Transaction	Customer Shipping Bank
4. Time to produce order summary reports	"End of week, month, quarter, and year"		Produce order summary reports	Order summary reports	Management
5. Time to produce transaction summary reports	"End of day"		Produce transaction summary reports	Transaction summary reports	Accounting
6. Customer or management wants to check order status	Order status inquiry	Customer or management	Look up order status	Order status details	Customer or management

Use case descriptions

- Intermediate description of the Web order scenario for Create new order

Flow of activities for scenario of *Customer creates Web order*

Main Flow:

1. Customer connects to the RMO home page and then links to the order page.
2. If this is a new customer, customer links to the customer account page and adds the appropriate information to establish a customer account.

2a If existing customer, customer logs on.

3. The system starts a new order and displays the catalog frame.
4. Customer searches the catalog.
5. When customer finds the correct item, he/she requests it be added to the order; the system adds it to the shopping cart.
6. Customer repeats steps 4 and 5.
7. Customer requests end of order; system displays a summary of the ordered items.
8. Customer makes any changes.
9. Customer requests payment screen; system displays payment screen.

9a Customer enters payment information; system displays summary information and sends confirmation e-mail.

10. System finalizes order.

Exception Conditions:

2a If existing customer forgets password, then
   a. customer can invoke forgotten password processing, or
   b. customer can create a new customer account.

9a If customer payment is rejected due to bad-credit verification, then
   a. customer can cancel the order, or
   b. order is put on hold until check is received.

Fully developed use case description

Use Case Name:	*Create new order*
Scenario:	Create new telephone order
Triggering Event:	Customer telephones RMO to purchase items from the catalog.
Brief Description:	When customer calls to order, the order clerk and system verify customer information, create a new order, add items to the order, verify payment, create the order transaction, and finalize the order.
Actors:	Telephone sales clerk.
Related Use Cases:	Includes: *Check item availability.*
Stakeholders:	Sales department: to provide primary definition. Shipping department: to verify information content is adequate for fulfillment. Marketing department: to collect customer statistics for studies of buying patterns.
Preconditions:	Customer must exist. Catalog, Products, and Inventory items must exist for requested items.
Postconditions:	Order and order line items must be created. Order transaction must be created for the order payment. Inventory items must have the quantity on hand updated. The order must be related (associated) to a customer.
Flow of Activities:	**Actor** / **System**
	1. Sales clerk answers telephone and connects to a customer.
	2. Clerk verifies customer information. — 2.1 Display customer information.
	3. Clerk initiates the creation of a new order. — 3.1 Create a new order.
	4. Customer requests an item be added to the order.
	5. Clerk verifies the item (*Check item availability* use case). — 5.1 Display item information.
	6. Clerk adds item to the order. — 6.1 Add an order item.
	7. Repeat steps 4, 5, and 6 until all items are added to the order.
	8. Customer indicates end of order; clerk enters end of order. — 8.1 Complete order.
	8.2 Compute totals.
	9. Customer submits payment; clerk enters amount. — 9.1 Verify payment.
	9.2 Create order transaction.
	9.3 Finalize order.
Exception Conditions:	2.1 If customer does not exist, then the clerk pauses this use case and invokes *Maintain customer information* use case. 2.2 If customer has a credit hold, then clerk transfers the customer to a customer service representative. 4.1 If an item is not in stock, then customer can a. choose not to purchase item, or b. request item be added as a back-ordered item. 9.1 If customer payment is rejected due to bad-credit verification, then a. order is canceled, or b. order is put on hold until check is received.

Fully developed use case descriptions

- Scenario:
  - A unique set of internal activities with the use case
  - Dependent on which actor triggers the event
  - A single use case may have multiple different fully developed use case description
    - Based on which scenario
    - The processes will differ for different scenarios
- Triggering Event
  - Will also depend on use case scenario
- Actors
  - Outside the system
  - Depend on scenario

Fully developed use case descriptions

- Related Use Cases
  - Can help you cross reference
  - Useful for documenting users' requirements
- Stakeholders
  - Who else cares about this scenario?
  - What are their motivations?
- Preconditions
  - The required state of the system before the scenario can begin
  - What objects must exist?
  - What information is needed?
  - What is the condition of the actor?
- Postconditions
  - What must be true after the scenario plays out
  - What new objects are created
  - What objects are updated
  - What associations must exist

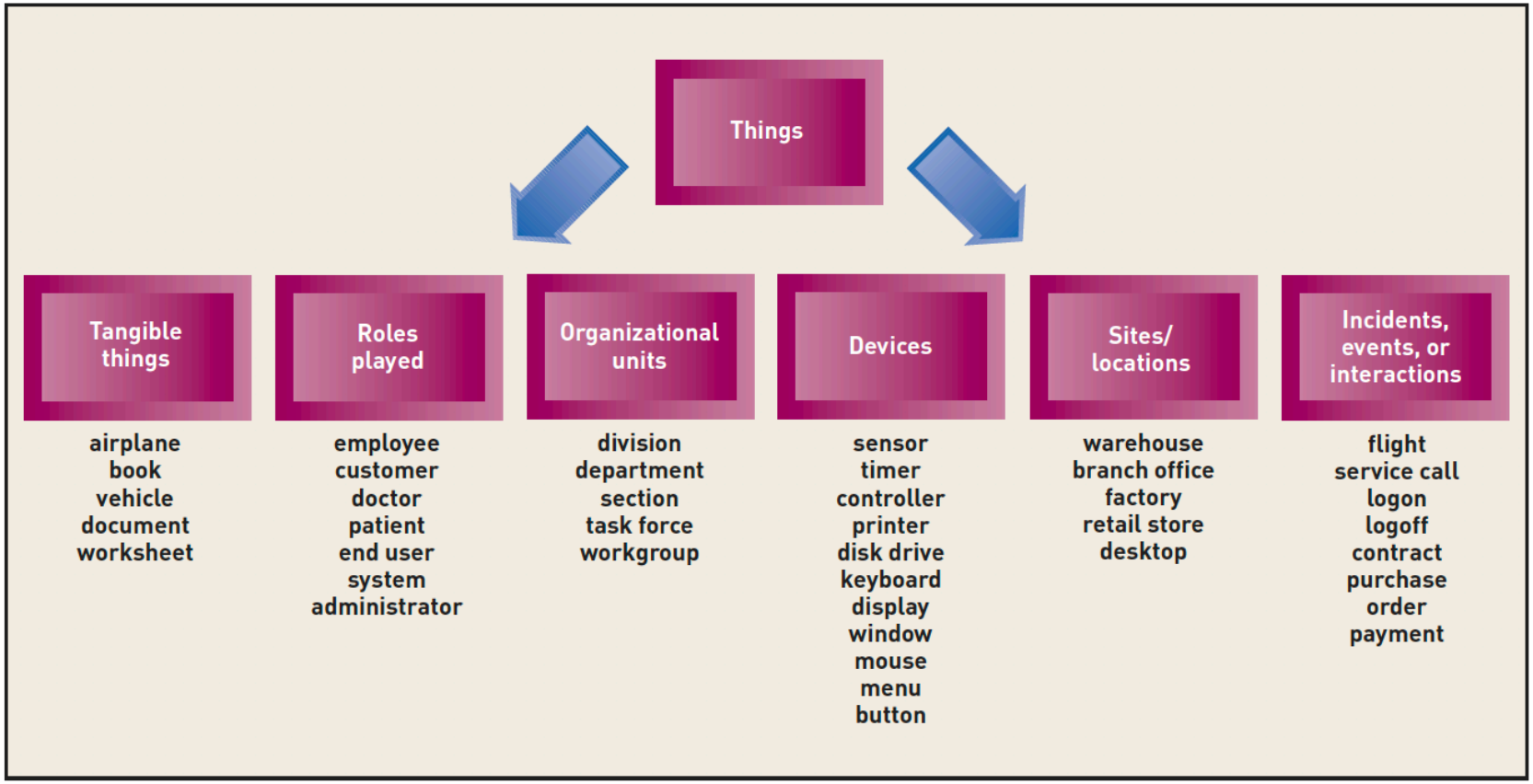
Fully developed use case descriptions

- Flow of Activities
  - Back and forth between actor and system
  - Numbered sequence of steps
- Exception Conditions
  - Numbered to link them to the activities
  - Useful in designing test cases

“Things” in the system problem domain

- Understanding and modeling things about which the system needs to store information
  - To the users, these items are the things they deal with when they do their work— products, orders, invoices, and customers—that need to be part of the system.
  - things are similar to the external agents or actors
    - a customer external agent places an order, but the system also needs to store information about the customer
  - things are distinct from external agents
    - there is no external agent named product, but the system needs to store information about products.
- In the object-oriented approach, these things become the objects that interact in the system.

Types of things



Ask users to name all the things they work with

Procedure for developing an initial list of things

- Step One: Using the event table and information about each event, identify all of the nouns.
- Step Two: Using other information from existing systems, current procedures, and current reports or forms, add items or categories of information needed.

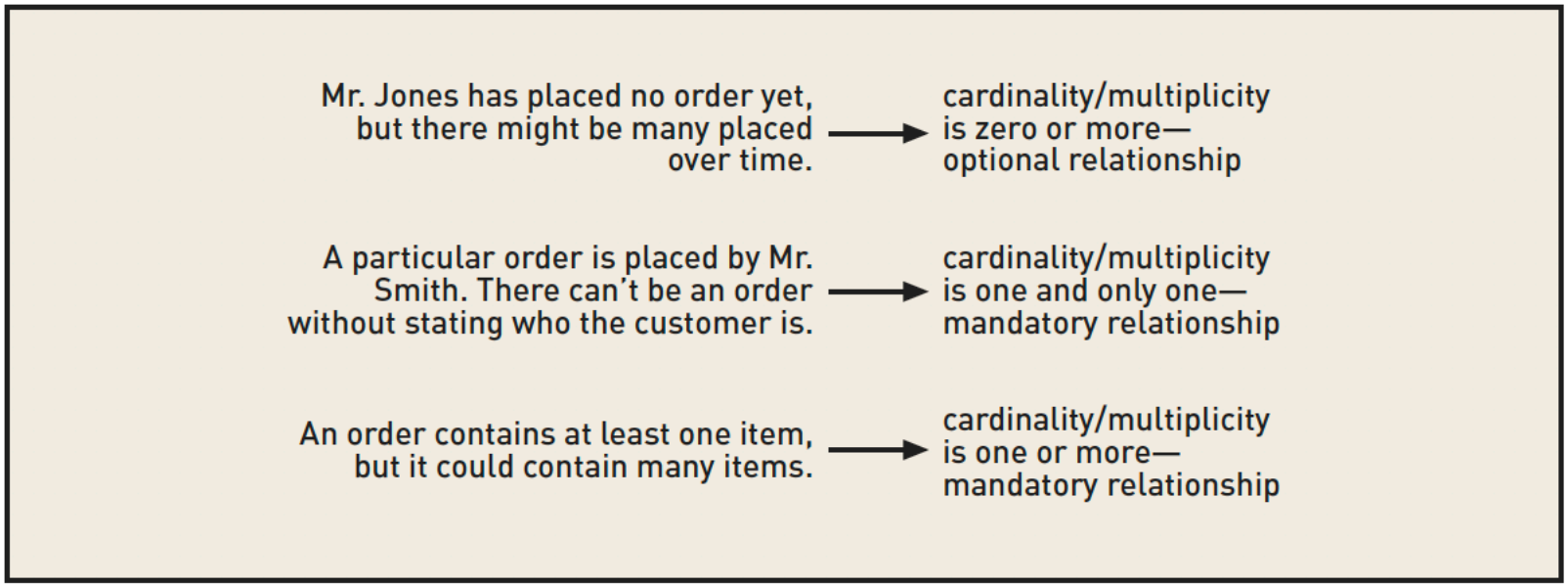
- Step Three: Refine the list and record assumptions or issues to explore.
  - include
    - Is it a unique thing the system needs to know about?
    - Is it inside the scope of the system I am working on?
    - Does the system need to remember more than one of these items?
  - Exclude
    - Is it really a synonym for some other thing I have identified?
    - Is it really just an output of the system produced from other information I have identified?
    - Is it really just an input that results in recording some other information I have identified?
  - research
    - Is it likely to be a specific piece of information (attribute) about some other thing I have identified?
    - Is it something that I might need if assumptions change?

Identified Noun	Notes	
Accounting	There's only one, we don't need to store it.	
Back order	A kind or order? A value of an order status?	
Back-order information	Output produced with other info.	
Bank	Only one, we don't need to store.	
Catalog	Different ones for different seasons.	
Catalog activity reports	Output produced from other info:	
Catalog details	Same as catalog? Same as product items in catalog?	
Change request	Input resulting in changes to an order.	
Charge adjustment	Input resulting in a transaction.	
Color	a piece of info about a product:	
Confirmation	Output produces from other info.	
Credit card info	Part of an order? Part of customer info?	
Customer	A key thing with lots of details:	
Customer account	Maybe needed if RMO payment plan is included.	

Identified Noun	Notes	
Accounting	There's only one, we don't need to store it.	Exclude
Back order	A kind or order? A value of an order status?	Research
Back-order information	Output produced with other info.	Exclude
Bank	Only one, we don't need to store.	Exclude
Catalog	Different ones for different seasons.	Include
Catalog activity reports	Output produced from other info:	Exclude
Catalog details	Same as catalog? Same as product items in catalog?	Research
Change request	Input resulting in changes to an order.	Exclude
Charge adjustment	Input resulting in a transaction.	Exclude
Color	a piece of info about a product:	Exclude
Confirmation	Output produces from other info.	Exclude
Credit card info	Part of an order? Part of customer info?	Research
Customer	A key thing with lots of details:	Include
Customer account	Maybe needed if RMO payment plan is included.	Research

Relationships among things

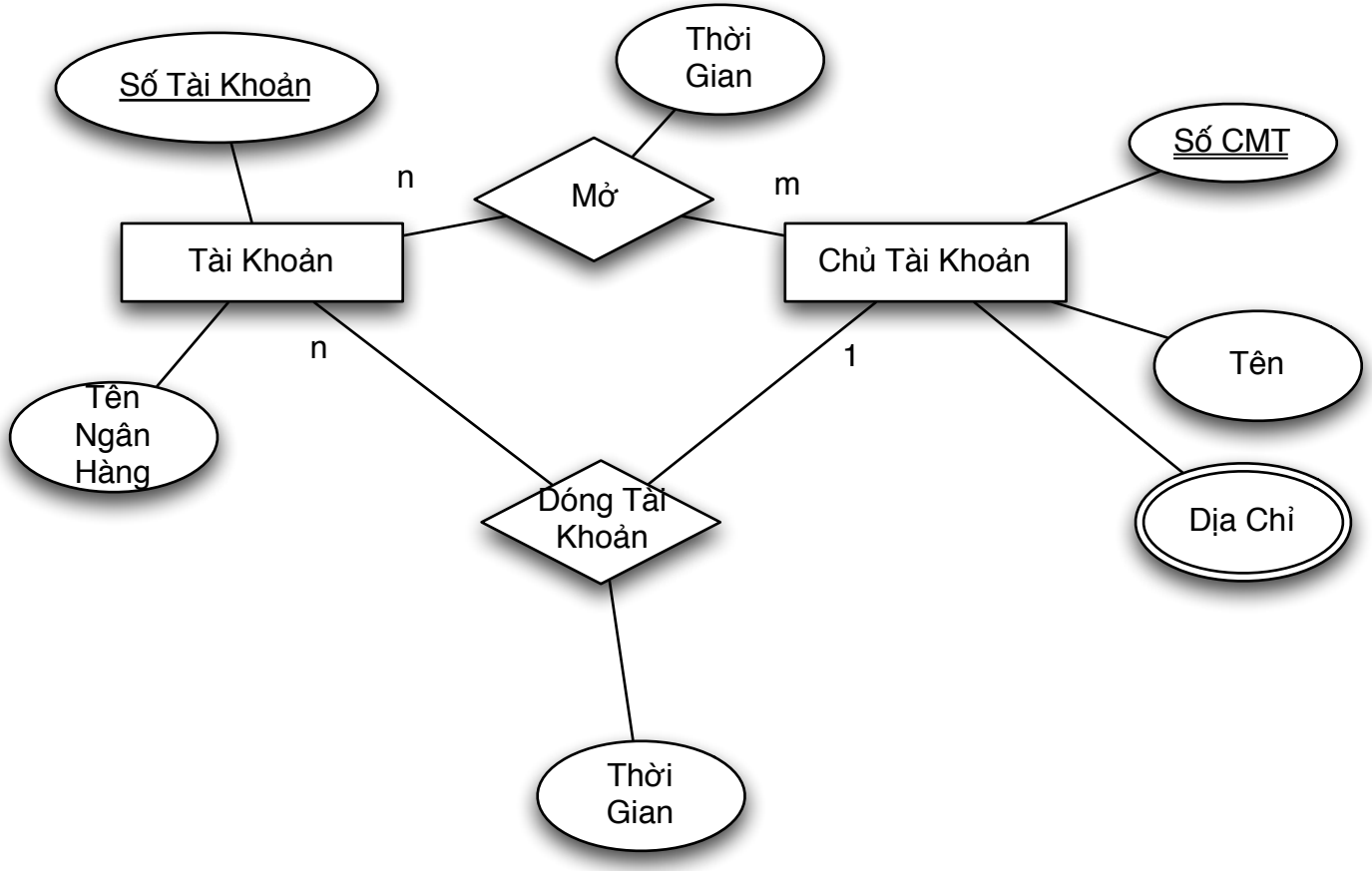
- Cardinality
  - the number of associations that occur among specific things, such as a customer places many orders and an employee works in one department



Attributes of things

- attribute
  - one piece of specific information about a thing
- identifier (key)
  - an attribute that uniquely identifies a thing
- Compound attribute
  - an attribute that contains a collection of related attributes

Entity-relationship (ER) diagram



Thank you for your attention!

Q&A