COMP3297 Annotated example of a Fully Detailed Use Case

Taken from Larman's chapter with some modifications. This general format, in turn, follows that recommended by Alistair Cockburn in the book "Writing Effective Use Cases " (Cockburn's work has had a large influence on the way use cases are presented). It deviates only slightly from a template suggested by Wiegers and we'll use a modified form of Wiegers template in the course project. The notes in boxes are for your information and are not part of the use-case.

Sections are ordered such that everything that a reader needs to know to understand the description of flow appears before the Main Success Scenario.

This is a particularly complicated use case because Process Sale is a complicated piece of functionality. Most of the use cases you will write in your career will be much simpler than this.

Items in square brackets are optional.

Use Case POS-UC1: Process Sale
[Created By: Craig Larman **Date Created:** Septemb

Begin by identifying the use case. As a minimum supply an ID and Name. Can also include Author and Date if multiple authors contribute use cases.

[Scope: ParknShop POS application]

[Level: User goal]

Scope: Since use-cases are also used for business modeling, sub-system modeling, etc., it is useful to record the system to which this use-case belongs.
Level: User goal or Sub-task. I.e. is this a full EBP or just a part of an EBP factored out, for example, because it is part of several other EBPs? Can omit if the answer is clear from the Description

Primary Actor: Cashier

Supporting Actors: Payment Authorization Service, Accounting System, Inventory System, ...

[**Trigger,** unless it is simply the first step in the flow]

Description:
Cashier uses the POS system to process the sale of goods selected for purchase by a customer and brought to the checkout. The POS system calculates the total of the sale and, when payment is confirmed, records details of the sale and issues a receipt.

Stakeholders and Interests:
- *Cashier:* Wants accurate, fast entry, and no payment errors, because cash shortages are deducted from his/her salary.
- *Customer:* Wants to make purchase quickly and with minimal effort. Wants proof of purchase. Wants to see display of items and prices as they are entered.
- *Company:* Wants accurate record of transactions. Wants to satisfy customers. Wants to ensure that Payment Authorization Service payment receivables are recorded. Wants fault tolerance such that sales can be captured even if server components (e.g., remote credit validation) are unavailable. Wants fast, automatic and timely update of accounting and inventory data.
- *Manager:* Wants to be able to override easily. Wants to be able to solve cashier problems with minimal effort and interruption.
- *HKSAR Government:* In near future, will want to collect tax from every sale.
- *Payment Authorization Service:* Wants to receive digital authorization requests in the correct format, using correct protocol. Wants accurate accounting of their payables to the branch of the company.

Some templates do not have this section, but it is very useful in practice for checking that you are satisfying all stakeholders' interests. Can be omitted if the only stakeholder is the primary actor and the actor's goals are clear.

Preconditions:
PRE-1: Cashier is identified and authenticated.

Success Guarantee (Postconditions):
POST-1**:** Sale is saved. (Future: Tax is correctly calculated.)
POST-2: Accounting and Inventory are updated.
POST-3: Receipt is generated.
POST-4: Payment authorization approvals are recorded.

Main Success Scenario (also known as Basic Flow or Normal Flow):

1. Customer arrives at POS checkout with items to purchase.
2. Cashier starts a new sale.
3. Cashier enters item identifier.
4. System records sale line item and presents item description, price, and running total. Price calculated from a set of price rules.

Cashier repeats steps 3-4 *until indicates no more items.*

5. System presents total (Future: with taxes calculated).
6. Cashier tells Customer the total, and asks for payment.
7. Customer pays and System handles payment.
8. System logs completed sale and sends sale and payment information to the external Accounting system and/or Inventory system.
9. System presents receipt.
10. Customer leaves with receipt and items (if any).

When using the *essential style* we don't specify how the item identifier is entered

An example of a price rule is a "3 for price of 2" offer.

Notice this common variation in the presentation of a main success scenario. At step 7, no particular form of payment is specified. Each possible form is treated as an extension.

An option is to separate Extensions into Alternative Flows and Exceptions. This can be useful to make the success scenarios clear. Remember, although there is only one main success scenario there can be many other success scenarios (Alternative Flows) in the use case

Extensions:

***a.** At any time, System fails:
To support recovery and accurate accounting, ensure all transaction sensitive state and events can be recovered from any step of the scenario.
1. Cashier restarts System, logs in, and requests recovery of prior state.
2. System reconstructs prior state.
2a. System detects anomalies preventing recovery:
1. System signals error to the Cashier, records the error, and enters a clean state.
2. Cashier starts a new sale.

***b.** At any time, Manager requests override:
1. System enters Manager-authorized mode.
2. Manager or Cashier performs one Manager-mode operation (e.g. cash balance change, resume on another register, void a sale,…).
3. System returns to Cashier-mode.

1a. Customer or Manager indicates desire to resume suspended sale:
1. Cashier performs resume operation, and enters the ID to retrieve the sale.
2. System displays the state of the resumed sale and its subtotal.
2a. Sale not found:
1. System signals error to the Cashier.
2. Cashier starts new sale to re-enter all items or cancels request.
3. Cashier continues with sale.

3a. Invalid identifier (item identifier not found in system):
1. System signals error and rejects entry.
2. Cashier responds to the error:
2a. There is a human-readable item identifier (e.g. numeric UPC):
1. Cashier enters the identifier manually.
2. System displays item description and price.
2a. Invalid item identifier: System signals error. Cashier tries alternative method.
2b. No item identifier, but a price is available:
1. Cashier requests override
2. Cashier indicates manual price entry, enters price, (Future: specifies tax class).
2c. Cashier requests *Find Product Help* to obtain item identifier and price.
2d. Cashier rejects item.

3b. There are multiples of same item category and tracking unique item identity is not important (e.g., 3 bottles of Watson's Water, 1lt.):
1. Cashier enters item identifier and the quantity.

3-6a: Customer asks Cashier to remove an item from the purchase:
1. Cashier enters item identifier for removal from sale.
2. System removes item and displays updated running total.

3-6b. Customer tells Cashier to cancel sale:
1. Cashier cancels sale on System.

3-6c. Cashier suspends the sale:
1. System records sale so that it is available for retrieval on any POS terminal.
2. System provides a "suspend" receipt with list of items and a sale ID to retrieve and resume the sale.

4a. The system-generated item price is not wanted (e.g., Customer is offered a lower price for any reason):
1. Cashier requests approval from manager.
2. Manager requests override.
3. Cashier enters new price.
4. System presents new price.

5a. Customer says they are eligible for a discount (e.g. an employee discount):
1. Cashier signals discount request.
2. Cashier enters Customer identification.
3. System presents discount total, based on discount rules.

You'll notice the exact point at which the extensions rejoin the basic flow is often not specified explicitly. It should, however, be clear from the description. There is no need to write your flows as if they were code.

6a. Customer says they intended to pay by cash but don't have enough cash:
1. Cashier asks customer to use an alternate payment method.
   1a. Customer tells Cashier to cancel sale. Cashier cancels sale on System.

7a. Paying by cash:
1. Cashier enters the cash amount tendered.
2. System presents the balance due, and releases the cash drawer.
3. Cashier deposits cash tendered and returns balance in cash to Customer.
4. System records the cash payment.

7b. Paying by EPS:
1. Cashier initiates EPS transaction on customer account.
2. System sends total and store details to external EPS system.
   2a. System detects failure to collaborate with external EPS system:
      1. System signals error to Cashier.
      2. Cashier asks Customer for alternate payment.
3. System receives payment approval and signals approval to Cashier.
   3a. System receives payment denial:
      1. System signals denial to Cashier.
      2. Cashier asks Customer for alternate payment.
4. System records the EPS payment and payment approval.

The entire EPS system, including the card reader, is considered external here.

7c. Paying by cheque...

7d. Paying by credit...

Frequency of Occurrence: Could be almost continuous.

Business Rules: PNS-BR-005, PNS-BR-007.

As we've seen, we catalogue Business Rules in a separate document. Here we reference rules contained in that document.

Special Requirements:

- Touch screen UI on a large flat panel monitor. Text must be visible from 1 meter.
- Robust recovery when access to remote services such as the inventory system fails.
- Language internationalization on the text displayed, Chinese and English.
- Pluggable business rules to be insertable at steps 3 and 7.

Technology and Data Variations List:

*a. Manager override entered by swiping an override card through the card reader, or by entering an authorization code on the keypad, or by unlocking with a physical key.
3a. Item identifier entered by scanning item's bar code with a laser scanner, or via keyboard.
3b. Item identifier may be any UPC, EAN, JAN, or SKU coding scheme.
7a. EPS transactions are conducted via standard external cradle and card reader.

Assumptions:

1. Assume cashier has access to standard names and list of identifiers for items which do not carry a bar-code (e.g. unpackaged vegetables).
2. …

Things we are assuming to be true but might not be. It is outside the scope of the POS system to provide the list of names and identifiers, but we assume it is available. Cashier also needs names in order to use *Find Product Help*.

Notes and Open Issues:

- What flexibility is needed for tax calculation?
- Explore the remote service recovery issue.
- What customization is needed for different businesses?

To act as central location to list issues that have yet to be resolved for this use case.