# Analysis – use cases summary

This is mostly a work document for the post-elicitation stage done with all analysts after the initial interviews and document reviews. Do this first. It’s a summary of the use cases in bullets or numbers serving like a table of contents. This is an optional part of a project document but when there are many use cases, it is very helpful.

## Actors

First step in analysis is to brainstorm and validate these roles and/or systems.

* Customer
* System
* Manager
* InvControl

## Use case names

Second step in analysis is to brainstorm and validate scope at the goal level.

### System use cases

Only those use cases which start after the software is running and an actor directly interacts with the system.

* Check inventory remotely
* Buy product with credit card
  + Customer walks up to kiosk, select product, uses credit card, gets product and receipt.
* Buy product with ApplePay
* Maintain inventory internally
* View sales report
  + Manager requests a report any number of machines remotely and gets data.

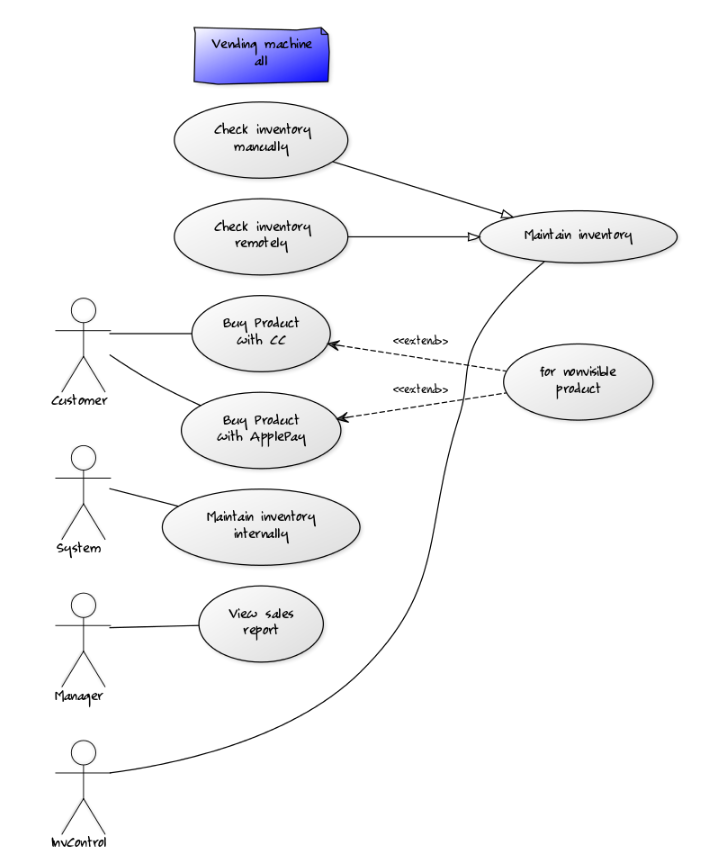
### Business use cases

Any use case that does not completely exist as an interaction between software and the actor.

* Check inventory manually
* Stock machine

### Use case diagram

Here would be a good place to put a diagram(s) showing the relationships of all the use cases. A good online tool is https://yuml.me/diagram/usecase/draw



## Grouped use cases

Use cases can be structured to roll up to a group name but are useful when complexity is high. Place bigger use cases that don’t all have the same flow of events here but have a common theme.

Grouped use cases are either system or business use cases and can be groups of goal level use cases or groups of partial use cases. Also common in this section are the Manage <Entity> type of use case which expands into the CRUD use cases. e.g. Manage Account = Create Account, Read Account (Display details of account), Update Account, Delete Account.

* Maintain inventory
  + Check inventory manually
  + Check inventory remotely

## Partial goal names found

Use cases can be structured to drill down to a sequence of tasks not ending in a goal with value or repeatable. That sequence may be assigned a name to help out understanding e.g. Log In or Search for Product. These are helpful to identify reusable parts and cut down the amount of repeated text in a set of use cases, but not necessary.

* View non-visible products

# Use Case Details - Structured

Final versions of the requirements documents used for walk-throughs with the main stakeholders.

IDs can be anything from consecutive numbers to meaningful acronyms about what kind of use case this is and what system it is a part of.

## C1-Buy product with credit card

Repeat this template for each use case to be documented.

### General info

#### Description:

Short summary that can be extracted and used in the use case summary page.

Customer walks up to kiosk, select product, uses credit card, gets product and receipt.

#### Actors:

The roles that can initiate this use case. It does not include participants.

Customer

#### Supporting roles/systems:

Sometimes called (supporting actors) and are other “actors” that are involved during the course of events. These can also be called interfaces.

* Credit card processor
* HQ server

#### Type:

Options are: system | business | distributed (business tasks interspersed with system tasks). Include the system name if several are used.

System

#### Pre-conditions:

Rules for beginning this use case: state of system prevents usage, must be testable. Or in a business use case, this must be the current state that has met a goal through another use case that this use case can now follow.

* There must be at least one product in machine.
* Communication has been made with processor in last five minutes.

### Scope info

#### Level:

Options are: goal | partial goal | group of goals | group of partial goals. Goal level will comprise 90% of the use cases.

Goal

### Tracking info

#### Author: Agile Use Case class

#### Date created: 5/15/18

#### Date revised: 5/15/18

### Project info

#### Design constraints:

Pure business term descriptions are hard to write. Constraints describe Any kind of policy, infrastructure, time, location, budget, hardware, or software that must be accommodated by this process e.g. web site is required, SQL Server is required, location must be…, hardware must be…

* Credit card reader is swipe/insert type (customer inserts chip card)
* Selection mechanism Vendormatic XYZ282 (customer rotates view)
  + Jam sensors
* Keypad Buttonmatic TYR7777 (customer select Enter key)
* 3G cell connection

#### Priority:

Priority will be by goal level or higher. Partial goal use cases will take their priority from the highest level that it is included in.

High (9).

#### Value to sponsor:

Value must be specified by the requirement that it is supporting for the business.

The growth of sales from external credit card customers will improve revenue.

#### Sponsor:

Who is accountable for this use case being delivered successfully?

Doug

### Course of Events

The sequence of tasks in conversation format between actor and system. For best linking to other steps, start each number with a system task except for the trigger. Combine actor responses to system events when well. Rules are placed under the task unless they can be reused and then they are referenced and placed in a separate file.

The number of tasks per number is usually small and starts with the system or the role. Tasks are individually stated so the system/role can do multiple things but in separate sentences. The last task will prepare the state of the system so that this use case can be performed again. There will be no condition statements to branch into two separate use cases. There may be a section that is removed to a named partial use case and called an <<include>> to shorten the detailed use case.

References that can be used here to document anything other than a functional requirement are:

* **T#** - Text file item number – used for error messages and small prompts
* **D#** - Design file item number – used for web pages, full screen menus, etc.
* **R#** - Report file item number – used for printed or on-screen report formats
* **\* -**  a Data Dictionary item – used to refer to data description and validation so that the detail doesn’t have to be specified here
* **Rule#** - Rule file item number – used to refer to process rules. Generally, this will follow one path only and another use case will pick up any other options. Some data validation rules find their way here but should be collected under the Data Dictionary. Unnumbered rules are not reusable and will just be defined below their functional requirement.

1. The use case starts when the actor finds a product they want to buy. The actor uses keypad to select the product.
2. The system checks inventory and displays number available. The system prompts actor for quantity to purchase (S2). The actor enters quantity to purchase.
3. The system displays total and prompts actor for payment (S4). The actor inserts credit card.
4. The system requests transaction approval from credit card processor. The processor approves.
5. The system dispenses product. The system prints receipt (R1).
6. The system updates inventory. The system transmits transaction to HQ server.
7. The system shows main screen (S1).

### Alternate flows (errors, exceptions)

The error flows are where a rule is broken, or something interrupts the normal “happy path” of the course of events. This often is during communication or other type of I/O.

* **Credit card denied** (4) – Prompt actor for another credit card? Return at #4.
* **Time out communicating with processor** (4) – Retry two more times. Prompt user with status message. If still fails, then cancel transaction. Use case ends.
* **Jam occurs** (5) - ?

### Alternate flows (extension points)

An exception to branching is when there is an optional <<extends>> of a partial use case. But the use case returns to where the option was taken.

* **Select non-visible product** (1) – Actor selects option to browse non-visible products. System shows products with product details.
* **Gift wrap product (3)** – System talks with actor about options and prices. Actor select options. Use case starts over at #3.

### Post-conditions

What are your tests that tell you that this is a successful completion of a use case? It may be a repetition of one of the tasks or a file or document that has been completed. But there are minimal ways to complete the goal and there are very excellent ways to complete it. Put both down.

* Credit card transaction was processed.
* Transaction was transmitted to HQ server.
* Customer received product.

### Notes/ Special Requirements

Any kind of quality, capacity, security, availability, disaster recovery information that is because of this use case. Maybe you also have ideas about design, or people who need to be checked with, etc.

* Temperature
* Tax tables must be loaded for zip codes….