# Strategy

If you need to validate what the organization is there to do.

Vision:

## Problem statement

Use standard format of <this problem> affects <these people> with these <specific bad results>. Our solution will help by providing <general good results>.

The access to banking services is restricted to bank hours affecting the majority of our bank customers by having to wait to the next business day, adding more staff or hours, or finding another solution. Our solution will allow the bank to provide services at any hour and in locations where a bank branch is not located reducing overhead costs. Replacing teller and not having bank open with automation of transaction external to bank reduces overhead and improves customer experience.

### Related problems

During problem definition, other problems can easily come up and should be captured.

## Constraints

The rules that will constrict either the scope, the budget, or the timeframe of the project. Risk is a probability that these will be in force. Assumptions for design will be to take the risk or not and trace back to these.

Infrastructure, technology, laws, etc.

## Stakeholders

A list of the people and systems that have an impact on the project requirements. Actors can be extracted and grouped from this list.

### People

|  |  |  |
| --- | --- | --- |
| Name | Position / Dept | Notes |
| I. Hafmunee | Bank president |  |
| Gee Kee Tekhed | CIO of bank |  |

### Systems & data sources

# Analysis – system level

Post-elicitation stage done with all analysts after the initial interviews and document reviews. Do this first.

## Actors

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

* Customer
* Operator
* System

## 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.

Grouped use cases are either system or business use cases and can be groups of value/goal level use cases or groups of partial use cases like used in this example. 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.

* <<group>> Do Transaction
  + Withdraw Cash
  + Deposit checks/cash envelope
  + Check balance
  + Transfer funds
* Communicate with bank computer
* Display greeting/prompt/ad

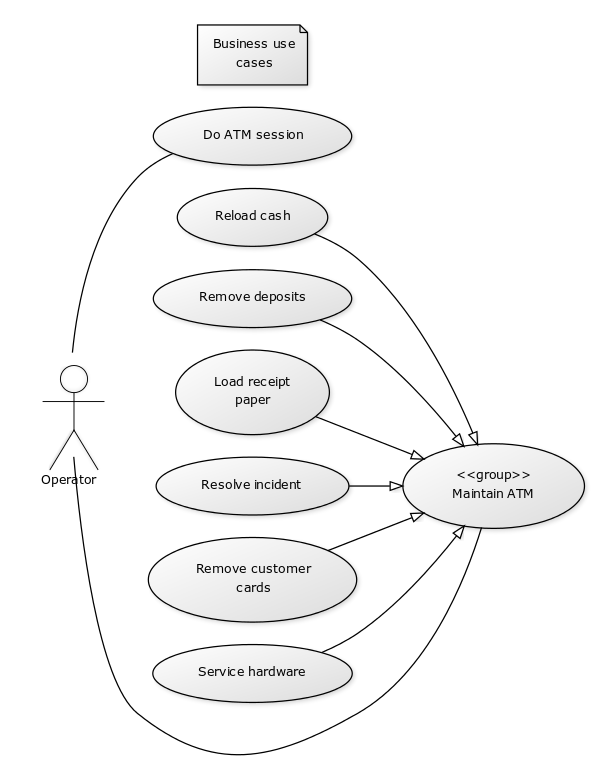
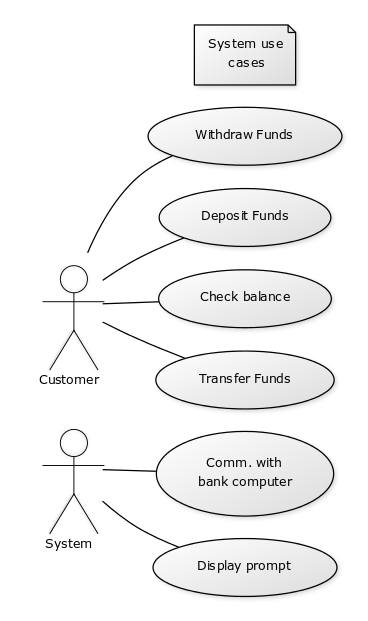
### Business use cases

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

* Do ATM session (switch on/off)
* <<group>> Maintain ATM
  + Remove deposits
  + Reload cash
  + Load receipt paper
  + Resolve incident/hardware failure
  + Remove customer cards
  + Service hardware

### Use case diagram

Here would be a good place to put a Visio diagram(s) showing the relationships of all the use cases.



## Partial goal names found

Use cases can be structured to drill down to a named sequence of tasks not ending in a goal or repeatable. Helpful to identify reusable parts.

* Authorize user

# S1 Withdraw Cash

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 inserts card, selects amount to withdraw, takes cash 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.

Bank

### 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.

Rule – **Cash in machine** - must be greater or equal to Max Withdrawal Amount

## 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

### Includes:

The use cases that are extracted out of this use case and given a special name, so they can be reused. They are required to be a part of this use case. This use case can be considered a grouped use case if it includes one of a group of partial goal use cases.

### Included in:

The use case(s) that uses this one as a necessary part of it.

### Use cases grouped by this ID:

If it doesn’t have an included group above, then it will be a category for several use cases.

### Grouped by:

The group that has others like this one.

G1 Do Transaction

## Tracking info

### Author: BA class of 5/10/18

### Date created: 5/10/18

### Date revised: 5/11/18

## 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…

* ATM chassis UDYWE8448484H9

### Priority: 4 (high)

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.

### Value to sponsor: Customer satisfaction and overhead efficiency.

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

### Sponsor: Doug

Who is accountable for this use case being delivered successfully?

## 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 inserts card.
2. The system gets info from card reader. The system validates the card.
   1. RULE: - **Valid banks**: list here…
3. The system prompts for PIN (D1). The actor enters PIN.
4. The system requests the bank to authorize actor with PIN and card data and return account info. The bank responds with an authorization and account info. The system logs the bank communication.
5. The system prompts the actor with a transaction menu (D2). The actor selects Withdraw Cash option.
6. The system prompts the actor for account to use (D3). The actor selects account.
7. The system prompts for withdrawal amount (D4). The actor enters withdrawal amount.
8. The system validates the withdrawal amount.
   1. RULE – **Withdrawal increment**: withdrawal amount must be evenly divisible by 20.
   2. RULE – **Available funds**: withdrawal amount must be equal to or less than available funds in account.
   3. RULE – **Maximum withdrawal amount** - $500 per transaction per day starting at midnight.
9. The system requests bank to debit account for actor. The bank confirms. The system logs the bank communication.
10. The system dispenses the requested amount of cash. The system logs the cash dispensing. The system prompts actor to take cash (D5). The actor takes the cash.
11. The system prompts for another transaction (D6). The actor declines.
12. The system prints receipt (R1). The system prompts to have actor take receipt (D7). The actor takes the receipt.
13. The system ejects the card. The system prompts for the actor to take the card (D8). The actors takes the card.
14. The system prompts with a thank-you screen (D9). The system starts timer of inactivity.

## 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.

* **Invalid bank card** (2a) – System prompts actor that they have an invalid bank card (D?). The system returns card. The actor takes the card. The system returns to idle state. The use case continues at #1.
* **Bank card not readable** (2) – System prompts actor that they have a bad bank card (D?). The …
* **Incorrect PIN** (4) – The system prompts of incorrect PIN. The actors confirms. The use case continues at #3.
  + **RULE – 3 bad PINs** – card is retained.
* **Increment incorrect** (8a) -
* **Funds not available** (8b) -
* **Maximum withdrawal amount exceeded** (8c) -

## 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.

* <Name> (<number(s) in course of events where this could occur>) – <description of what to do and where to return in the course of events>
* <Name> (<number(s) in course of events where this could occur>) – <description of what to do and where to return in the course of events>

## Post-conditions / success criteria

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.

* <Name of goal>
  + <minimum set of qualifications>
  + <maximum set of what could be>

## 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.

* Shelter
* Illumination

# S2 Transfer Funds

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.

### 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.

Bank

### 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.

## 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

### Includes:

The use cases that are extracted out of this use case and given a special name, so they can be reused. They are required to be a part of this use case. This use case can be considered a grouped use case if it includes one of a group of partial goal use cases.

### Included in:

The use case(s) that uses this one as a necessary part of it.

### Use cases grouped by this ID:

If it doesn’t have an included group above, then it will be a category for several use cases.

### Grouped by:

The group that has others like this one.

G1 Do Transaction

## Tracking info

### Author: BA class of 5/10/18

### Date created: 5/10/18

### Date revised: 5/11/18

## 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…

* ATM chassis UDYWE8448484H9

### Priority: 0 (low)

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.

### Value to sponsor: Customer satisfaction and overhead efficiency.

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

### Sponsor: Doug

Who is accountable for this use case being delivered successfully?

## 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 inserts card.
2. The system gets info from card reader. The system validates the card.
   1. RULE: - **Valid banks**: list here…
3. The system prompts for PIN (D1). The actor enters PIN.
4. The system requests the bank to authorize actor with PIN and card data and return account info. The bank responds with an authorization and account info. The system logs the bank communication.
5. The system prompts the actor with a transaction menu (D2). The actor selects Withdraw Cash option.
6. The system prompts the actor for account to use (D3). The actor selects account.
7. The system prompts for withdrawal amount (D4). The actor enters withdrawal amount.
8. The system validates the withdrawal amount.
   1. RULE – **Withdrawal increment**: withdrawal amount must be evenly divisible by 20.
   2. RULE – **Available funds**: withdrawal amount must be equal to or less than available funds in account.
   3. RULE – **Maximum withdrawal amount** - $500 per transaction per day starting at midnight.
9. The system requests bank to debit account for actor. The bank confirms. The system logs the bank communication.
10. The system dispenses the requested amount of cash. The system logs the cash dispensing. The system prompts actor to take cash (D5). The actor takes the cash.
11. The system prompts for another transaction (D6). The actor declines.
12. The system prints receipt (R1). The system prompts to have actor take receipt (D7). The actor takes the receipt.
13. The system ejects the card. The system prompts for the actor to take the card (D8). The actors takes the card.
14. The system prompts with a thank-you screen (D9). The system starts timer of inactivity.

## 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.

* **Invalid bank card** (2a) – System prompts actor that they have an invalid bank card (D?). The system returns card. The actor takes the card. The system returns to idle state. The use case continues at #1.
* **Bank card not readable** (2) – System prompts actor that they have a bad bank card (D?). The …
* **Incorrect PIN** (4) – The system prompts of incorrect PIN. The actors confirms. The use case continues at #3.
  + **RULE – 3 bad PINs** – card is retained.
* **Increment incorrect** (8a) -
* **Funds not available** (8b) -
* **Maximum withdrawal amount exceeded** (8c) -

## 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.

* <Name> (<number(s) in course of events where this could occur>) – <description of what to do and where to return in the course of events>
* <Name> (<number(s) in course of events where this could occur>) – <description of what to do and where to return in the course of events>

## Post-conditions / success criteria

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.

* <Name of goal>
  + <minimum set of qualifications>
  + <maximum set of what could be>

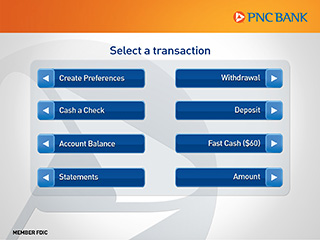
## Notes/ Special Requirements

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.

* Shelter
* Illumination

# Designs

## D1 – transaction menu



Nicole – fewer options are better….

# Reports

## R1 - Receipt

