# 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.

## UC1 Do secure transaction

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.

Actor goes to machine, inserts card, enters PIN, selects Withdraw Cash, enters amount. Bank verifies amount and actor gets cash and receipt.

#### Actors: Customer, ?

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

#### Supporting roles/systems (other stakeholders): Bank

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

#### Type: System

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

#### 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.

When the ATM has decided the Withdraw Cash use case cannot be used.

### Scope info

#### Level: Goal

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

#### 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: GUC1 Do Transaction

The group that has others like this one.

### Tracking info

#### Author: ATM class of 10/16/2020

#### Date created: 10/16/2020

#### Person and Date revised: ATM class of 10/16/2020 on 10/16/2020

### Project info

#### Design constraints: ATM chassis Bazinga 278385495

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…

#### Priority: 9 - 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 actor: Provides convenience by having 24/7 availability.

#### Value to sponsor: Provides service outside of bank business hours saves overhead and labor costs.

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

#### Sponsor: Mr. Big

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 mostly on forms.
* **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. Also bolded and colored is good.
* **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 their **ATM card\***.
2. The system reads card. The system prompts for PIN (D#1). The actor enters PIN.
3. The system requests the bank to validate the PIN (UC1.3.a). The system logs the communication (UC1.3.b). The bank returns **Accounts\*** for the actor.
4. The system prompts for the type of **Transaction\*** (D#2).
5. **<<includes>> PUC1 - Withdraw Cash (really includes any one of the secure transactions)**
6. The system prompts a next transaction (D#5). The actor declines.
7. The system prompts actor for a printed receipt (D#8). The actor declines.
8. The system returns the card. The system prompts actor to take card (D#7). The actor takes their card.
9. The system returns to idle screen (D#0).

### 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.

### 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.

**Print receipt** (#12) The system prints receipt. The actor takes the receipt. The use case continues at #13. (almost a separate use case)

### 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. Some people use MoSCow – must have, should have, could have instead of min and max conditions.

* The bank updated the actor's account.

### 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.

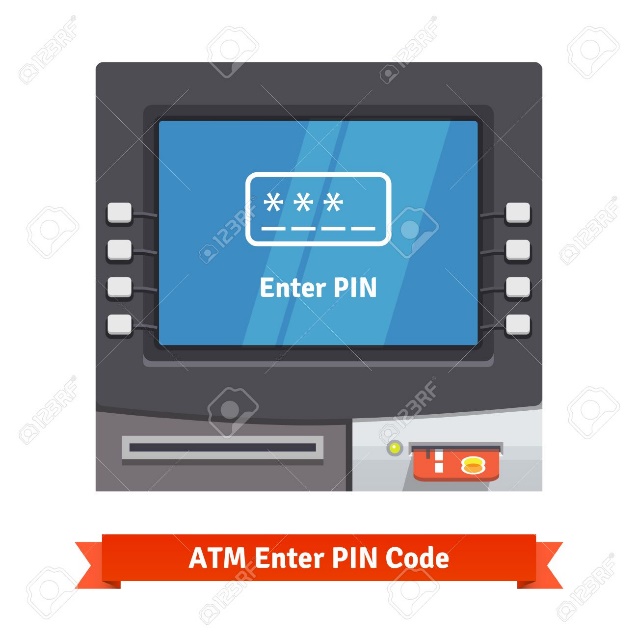
limited hours of operation?

limited geographic area?

security?

# Screen and report designs

D#1 - enter PIN



D#5 Next transaction

Yes / No

# PUC1 - Withdraw Cash

### General info

#### Description:

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

Actor goes to machine, inserts card, enters PIN, selects Withdraw Cash, enters amount. Bank verifies amount and actor gets cash and receipt.

#### Actors: Customer, ?

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

#### Supporting roles/systems (other stakeholders): Bank

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

#### Type: System

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

#### 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.

When the ATM has decided the Withdraw Cash use case cannot be used.

### Scope info

#### Level: Goal

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

#### 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: UC1 - Do secure transaction

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.

### Tracking info

#### Author: ATM class of 10/16/2020

#### Date created: 10/16/2020

#### Person and Date revised: ATM class of 10/16/2020 on 10/16/2020

### Project info

#### Design constraints: ATM chassis Bazinga 278385495

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…

#### Priority: 9 - 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 actor: Provides convenience by having 24/7 availability.

#### Value to sponsor: Provides service outside of bank business hours saves overhead and labor costs.

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

#### Sponsor: Mr. Big

Who is accountable for this use case being delivered successfully?

## Course of events

1. The actor selects Withdraw Cash.
2. The system prompts for account to select (D#3). The actor selects account.
3. The system prompts for amount to withdraw (D#4). The actor enters the amount.
4. The system validates the amount.
   1. RULE - Prevent overdrawn account - The amount to be withdrawn is equal to or less than balance in the Account.
   2. RULE - Maximum amount per transaction - Amount withdrawn is equal to or less than $200 for this transaction.
   3. RULE - Maximum amount per day - Amount withdrawn is equal to or less than $200 per day starting at midnight.
   4. RULE - Withdrawal increment amount - Amount withdrawn is in $20 increments.
5. The system transmits **Message\*** to debit account to the bank. The system logs the communication.
6. The system dispenses the cash. The system logs the dispensing of cash. The system updates cash on hand.
   1. RULE - Call for cash replenishment - If amount is less than $5000, the system sends message to the bank to replenish the cash.
   2. RULE - Not enough cash for withdrawal - If amount is less than $200, then system does not allow Withdraw Cash transactions.
7. The system prompts actor to take cash (D#6). The actor takes cash.

### 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.

**Prevent overdrawn account** fails (4) - The system prompts user that they don't have enough money in the account. The actor confirms. The use case continues at #7.

**Cancel button pressed** (2-3…) The system prompts user for cancelling transaction. The user agrees. The use case continues at #14.

### Designs

D#4 Withdrawal amount

