# Strategy

## 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 inability of providing an on-premise clerk to sell snack and beverage items has led to a loss in market share. Our solution will create a on demand solution to selling products at all hours in businesses giving a continued growth.

### Related problems

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

## Risks

Information that will increase or decrease the probability of project completion due to choosing a design.

* Weather related operational issues
* Location – traffic not enough
* Location installation – electrical, lighting, space for moving
* Contractual – other vendors are in place

## Resources used

* Stocking staff
* Vendor management for vending machine
* Vendor management for food items
* Transportation for staff and items
* Warehouse space

## 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.

* Space for display is limited to 5 square feet
* Size of items is limited to large sandwiches
* Freshness of items is limited to three days
* American currency or credit only

## 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

* Sales process owner
* Sales manager
* Sales staff
* Suppliers
  + Jim, Canteen – 12/21/2016 interview

### Systems & data sources

* Credit card transaction service
* Inventory listing with prices
* Employee listing

# Analysis – system scope

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.

* Buyer
* Stocker

## 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.

* System sells item for cash
* System sells item for credit
* System sells item on employee credit

### Business use cases

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

* The system is restocked
* The system’s cash is removed

### Use case diagram

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

## Use case groups found

Use cases can be structured to roll up to a group name. Helpful to reduce complexity.

* System sells vending items
* The system is maintained

## 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.

* System selects item to sell

# Project management

## Prioritization

Add other columns of ranking variables to mix in with weightings if useful. Keep the values simple.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Impact | \* Urgency | = Priority |  |
| Use case name | Business usage /  market potential 3=100-67%, 2= 66-34%, 1=0-33% | Business need / perceived value 3=exec/high,  2=mgmt/med  1=staff/low) | Multiplied  Result | In a group, ask for the top 3 and  count the  total. |
| System sells item for cash | 2 | 3 | 6 |  |
| System sells item for credit | 2 | 2 | 4 |  |
| System sells item on employee credit | 1 | 3 | 3 |  |

## Project Iterations

Move the use cases with priorities down to a schedule so that you can see what kind of sprints (Agile) you will have.

### set up

There needs to be a software stage to set up and get all the pieces initially running as a framework. It’s called stubbing out the system or mocking the system depending on what you need. It may be combined with the 1st iteration.

### 1st

The first iteration has to be the most complex use case or one that would likely fail. Stop the project early if this doesn’t work.

* Project set up
* System sells item for cash

### 2nd

* System sells item for credit

### 3rd



# Analysis - detail

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

## Use Cases

### Summary of all use cases

A summary of the use cases in bullets or numbers serving like a table of contents. Summary focuses on inputs and outputs.

|  |  |  |
| --- | --- | --- |
| Priority | ID Name | Description |
| 6 | VM1 Sell item for cash | Buyer inserts bills or coins, selects a product and gets change and product |
| 4 | VM2 Sell item for credit | Buyer inserts credit card, selects and gets a product. |
| 3 | VM3 Sell item for employee credit | Buyer swipes fob, selects and gets a product. |

### VM1 – Sell item for cash

#### Metadata

Information for project tracking

**Author** – Doug Hoff, IPFS students Brenda and Michael

**Date created** – 12/20/2016

**Date revised** –

**Version** – 1.01

#### Use case data

Information about the use case

**Actors** – Buyer

Does not include those participants involved but do not initiate/trigger this flow of events.

**Includes:**

**Extensions:**

**Related systems** – Inventory pricing data system (remote), change system

Sometimes called (supporting actors)

**System** – Vending system

Would not be used if this were a business use case

**Priority** – 6

Get this from the priority chart.

**Iteration** -

**Level** – Goal

Granularity options are: goal | partial goal | group of goals | group of partial goals

**Design constraints** – Cell phone tower access, electrical output within 10 feet of installation

Useful when you don’t use just business terms e.g. web site is required, SQL Server is required, location must be…

**Business goal** (value to sponsor) –

#### Pre-conditions

Rules for beginning this use case: state of system prevents usage, must be testable

* Machine must have sufficient stock
  + RULE – **Stocked machine** – there must be at least one item to sell in machine

#### Course of Events

The sequence of tasks in conversation format between actor and system. Start each number with the system except the trigger and combine actor responses to system events.

1. The use case starts when the actor inserts **cash** (DD). The change system validates cash.
2. The system prompts actor with amount entered (ET#1). The actor selects **inventory item** (DD) to purchase.
3. The system validates item can be purchased.
   1. RULE – **Item in stock** - Item is available to be purchased
   2. RULE – **No jam detected** – Item has fallen from previous purchase for same item.
4. The system validates it has sufficient cash to perform transaction.
   1. RULE – **Sufficient cash** - Cash must be equal or greater than item cost.
5. The system dispenses item. The actor retrieves item.

#### Extension points – optional

Extension points where use case can optionally go and then come back to the same point.

* **Cash is greater than purchase price** (4) – The vending system requests the change system to return **correct change**. The change system updates the amount of change in vending system. The vending system validates whether to show exact change message. The actor retrieves change. Use case continues on #5.
  + RULE – **exact change required** – less than any of 3 quarters, 4 dimes, or 1 nickel are available in the system.
* **Cash return is selected** (2) System returns cash and use case ends.

#### Extension points – errors, exceptions

Errors occur at any point where there is a validation of a rule.

Errors occur at communication to other systems.

* **Invalid cash** – (1) – System prompts user with bad cash message and change system reject bill or coin.
* **Item out of stock** (4a) – System prompts actor to select another item (ET#2)
* **Item has jammed** (4a) – System prompts actor to select another item. (ET#3)
* **More money needed** (3a) – System prompts actor to enter more money (ET#4)

#### Post-conditions

Is it really important to review the necessary outcomes of this use case? If so, summarize here. Also use for minimal conditions of satisfaction.’

#### Notes/ Special Requirements

Any kind of quality, capacity, security, availability, disaster recovery information.

* **System** – refrigeration when necessary for freshness or product enjoyment
* **System security** - protect products and money from theft.
* **Sales** – buyers need to see real products being purchased as incentive
* **Capacity** – must be able to sell no less than 30 items

## Data Dictionary

A separate file to describe in business terms the data entities and data groupings of the entire project.

* Cash
  + Quantity
  + Country of origin

|  |  |  |
| --- | --- | --- |
|  | US? | Result |
|  | No | Invalid |
|  | Yes | Valid |

* + Denomination
  + Type – paper or coin
  + RULES

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | US? | Type | Denomination | Result |
|  | Yes | Paper | 1 | Valid |
|  | Yes | Paper | 2 | Invalid |
|  | Yes | Paper | 5 | Valid |
|  | Yes | Paper | 10 and higher | Invalid |
|  | Yes | Coin | .01 | Invalid |
|  | Yes | Coin | .05 | Valid |
|  | Yes | Coin | .10 | Valid |
|  | Yes | Coin | .25 | Valid |
|  | Yes | Coin | .50 | Invalid |
|  | Yes | Coin | 1.00 | Valid |

* Inventory item
  + Product
  + Manufacturer
  + Price
    - RULE – Items must be more than MINIMUM\_VEND\_PRICE and less than MAXIMUM\_VEND\_PRICE
  + Cost

## Business Rules

A separate file to describe in business terms the constraints of the processes used within the use case sequences of tasks. If a rule is only used once, include it in the use case.

* Credit card accepted – MasterCard, VISA, AmEx.
* MINIMUM\_VEND\_PRICE = .50
* MAXIMUM\_VEND\_PRICE = 10.00

## Glossary

A separate file to define in business terms the less understood terms used in the requirements documents.

**term -**  definition

# Design documents

## Assumptions

Those constraints that are risks which have to be taken in order for a design process to begin.

* Risk – weather
  + Temperature is no lower than 0 degrees.
  + Temperature is no higher than 100 degrees.

## Text messages

A list of IDs and their associated text messages used for internationalizing words on a screen, menu, or other small pieces of text. This is technically called externalized text.

|  |  |
| --- | --- |
| ET#1 |  |
| ET#2 |  |
| ET#3 |  |
| ET#4 |  |

## Screens

A separate file to collect design recommendations from users and analysts for either suggestions on final design or for ideas while thinking through the requirements.

**SD#1** – panel interface layout

**Type** – User elicited

User elicited - analysis | analyst approved – design



## Reports

A separate file to collect design recommendations from users and analysts for either suggestions on final design or for ideas while thinking through the requirements.

R#1 – Report layout - **receipt**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Header info** | | | | | | | |
| Sort 1  Header with page break | Sort 2  Header | Sort 3  Ascending | Sort 4 |  |  |  |  |
| **Field 1** | **Field 2** | **Field 3** | **Field 4** | **Field 5** | **Field 6** | **Field 7** | **Field 8** |
| Format - | Link to  detail - screen |  |  |  |  |  |  |
| Total on  Sort 1 |  | Total on  Sort2, Sort 3 |  |  |  | Total on All | Total on All Average All |
| **Footer info** | | | | | | | |

# Test documents

## Test cases

### System scenario tests

1. VM1 (happy path)

#### Pre-conditions

1. VM1 + **Stocked machine**

#### Optional paths

1. VM1 + **Sufficient cash** #4
2. VM1 + **Cash is greater than purchase price** #4
3. VM1 + **Cash is greater than purchase price** #4 + **exact change required**
4. VM1 + **Cash return is selected** #2

#### Errors

1. VM1 + **Invalid cash**
2. VM1 + **Item out of stock** #4a
3. VM1 + **Item has jammed** #4a
4. VM1 + **More money needed** #3a

## Test data

Equivalence set best candidate boundary – exact amount

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| VM1 Data - ID | Money Type | Amount | Product | Set | Best candidates |
| VM1.1a, b | Coin | 1.00 – 1.75 | Any one product | Valid | 1.00 |
| 1.75 |
| VM1.2a, b | Coin | .95 or less, 1.80 or more | Any one product | Invalid | .95 |
| 1.80 |
| VM1.3 | Bill | 1.00 | Any one product | Valid | 1.00 |
| VM1.4 | Bill | 2.00 | Any one product | Invalid | 2.00 |
| VM1.5a, b | Mixed coin and bill | 1.00 – 1.75 | Any one product | Valid | 1.00 |
| 1.75 |
| VM1.6 | Mixed coin and bill | 1.80 or more | Any one product | Invalid | 1.80 |

## Test cases

### Test case VM1.001

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Buy an item with cash – happy path** | | | | | |
| ID: **VM1.001** | | | | | |
| Type: scenario system | | | | | |
| Test suite(s): **VM1** | | | | | |
| Test data: **VM1.1a** | | | | | |
| Priority: 6 | | | | | |
| Hardware required: Canteen-wire-snack1 | | | | | |
| Subsystems required: Vend1CC2Change3Price2 | | | | | |
| **Duration**: 1 min. | **Start**: | **End**: | | | |
| **Effort**: 1 min. | **Time spans**: | | | | |
| **Setup**: none | | | | | |
| **Tear down**: clear transaction log | | | | | |
| **Time started:** | | | | | |
| **Test step** | **Test data** | | **Expected Result** | **Actual Result** | **Bug ID** |
| 1. insert **cash** | 1.00 in bill | | prompts with amount entered |  |  |
| 1. select **inventory item** to purchase | D3- Bugels | | system dispenses item |  |  |
| 1. retrieve item |  | |  |  |  |
| **Time completed:** | | | | | |
| **Comments by tester:** | | | | | |

|  |  |
| --- | --- |
| Test execution summary | |
| Tester |  |
| Date completed |  |
| Duration, actual |  |
| Effort, actual |  |
| Result |  |

### Test case VM1.Ex

|  |  |  |
| --- | --- | --- |
| **Buy an item with cash – exploratory** | | |
| **Type: Exploratory** | | |
| **ID: VM1.Ex** | | |
| **Test suite(s): VM1** | | |
| **Test data: any** | | |
| **Priority: 6** | | |
| **Hardware required: Canteen-wire-snack1** | | |
| **Subsystems required: Vend1CC2Change3Price2** | | |
| **Duration**: 1 hour | **Start**: | **End**: |
| **Effort**: 1 hour | **Time spans**: | |
| **Setup**: none | | |
| **Tear down**: clear transaction log | | |
| **Time started:** | | |
| **Description:**  Try to find any issues with change usage, item selection or…. And record results below: | | |