Jhorman Germán Mera Escobar
Mateo Rada Arias
Paula Andrea Trujillo Mejía

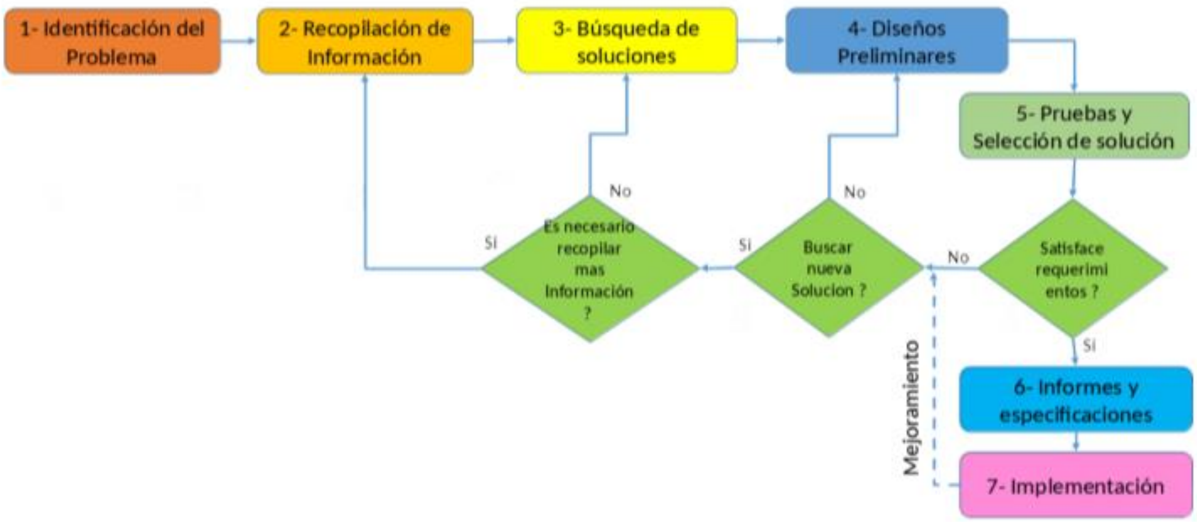
Engineering method
Application for video game store problem

Problematic context

A video game store just created by an eccentric foreign millionaire requires a solution for the entry, processing and exit of customers to the store.

Solution development

To solve the previous situation, the Engineering Method was chosen to develop the solution following a systematic approach and in accordance with the problematic situation raised.
Based on the description of the Engineering Method in the book “Introduction to Engineering” by Paul Wright, the following flow diagram was defined, the steps of which we will follow in the development


1- Identificación del Problema
2- Recopilación de Información
3- Búsqueda de soluciones
4- Diseños Preliminares
5- Pruebas y Selección de solución
No
No
Si
Es necesario recopilar mas Información ?
Si
Buscar nueva Solucion ?
No
Satisface requerimi entos ?
Si
Mejoramiento
6- Informes y especificaciones
7- Implementación

Step 1: Identifying the Problem

Normally, in a typical video game store, the customer usually takes a considerable amount of time to search for the titles of their interest or even to find new titles that they find interesting to venture to try. In addition, for different reasons, the excessive, and in some cases unnecessary, extra interaction with the store personnel can lead to the customer's time spent in the store being extended or the customer's experience not being the best.

<u>Needs and Symptoms</u>

- Customers need, before entering the establishment, to have knowledge of the catalog that the store has at its disposal.

● In the catalog, it must contain reviews and critiques of the games at the time of being consulted by the client.
● The customer must be able to add the games of their interest to a shopping list.
● The shopping list of each client, after completion, must generate an identification code that will be used when entering the establishment.
● Each game must have its own identification code.
● On the premises, there must be a set of stations that allow quick consultations.
● When entering the code from the list, the names of the video games must be uploaded to the first station, which must indicate the blocks or shelves where it must search for each of the games of interest.
● The video game list should be organized depending on the location of each of the shelves, in such a way that the buyer follows the best route.
● If the game is out of stock, your code will not appear in the list sorted by the location of its respective shelf.
● The customer must be able to choose between 2 ordering algorithms to fulfill the task of organizing his list depending on the location of each shelf.
● The customer must be assigned an automated basket that follows him to locate the games he finds, one on top of the other, following the order provided in the previous stage.
● Each customer must have a time depending on their entrance to the store, to which is added the time the customer spent looking for the video games.
● The order in which customers enter the checkout is given by the time the customer entered the store plus the time it took to collect the games.
● All customers must make a single queue even if there are several ATMs or service points.
● If there are available service points, as many clients will be attended simultaneously as there are available service points.
● The output order may vary, as it depends on how long each customer takes at the checkout, which is given by the number of items they are going to buy.
● Video games, at the time of being billed, the order will be in the way that the last game added to the basket must be the first one billed and packed.
● After completing the billing, the total cost of the games listed by each customer must be indicated.

Definition of the problem
The video game store requires the development of a shopping simulator that allows Caleños to know a little about how this new store will work in the city.

Step 2: Gathering Information
In order to understand the operation of the application a little more, it is necessary to know those algorithms that will make it possible to execute it.

Generics: Programming with generics allows us to create reusable code in multiple entities. With this type of generic code, it will automatically work with the type of data passed to its type parameter. Also, they allow the compiler to report many compilation errors that until now will only be discovered at runtime, at the same time, they allow to eliminate casts by simplifying, reducing repetition and increasing the readability of the code.

Stacks: The Stack class is a class of the LIFO type calls (Last In - First Out in Spanish, Last in Enter - First in Exit). It is a stack, which has a sequence of elements of the same type in which access to it is done through a single place called the top: The operations that characterize the stack are to introduce a new element on top (push) and that of extracting the element located at the top (pop).

Queue: A Queue or Queue is a data structure that follows the FIFO Philosophy (First In - First Out in Spanish First in Enter - First in Exit). Items can be inserted at any time, but only the item that has been in the queue the longest can be removed at any time.

TAD: An Abstract Data Type (TAD) is a set of data or objects to which operations are associated. The TAD provides an interface with which it is possible to perform the permitted operations, abstracting from the way in which these operations are implemented.

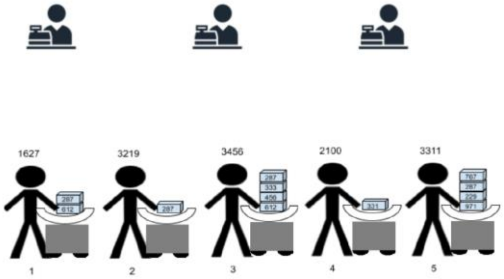
Hash Tables: A Java Hashtable is a data structure that uses a hash function to identify data using a key or key. In other words, it would be quite convenient to be able to maintain the ability to easily access any description from its key, but at the same time use a reasonable amount of memory. These are two of the properties that a Hash Table has.

On the other hand, it is important to mention and explain a little where all the information of the application will be found.

Github: Github is a portal created to host the code of any developer's applications, the platform is created for developers to upload the code of their applications and tools, and that, as a user, you can not only download the application, but also enter to your profile to read about it or collaborate with its development. Git is one of these control systems, which allows you to compare the code in a file to see the differences between versions, restore old versions if something goes wrong, and merge changes from different versions. It also allows you to work with different branches of a project, such as development to add new functions to the program or production to debug bugs.
Thus, Github is a portal to manage the applications that use the Git system. In addition to allowing you to look at the code and download the different versions of an application, the platform also acts as a social network connecting developers with users so that they can collaborate by improving the application.

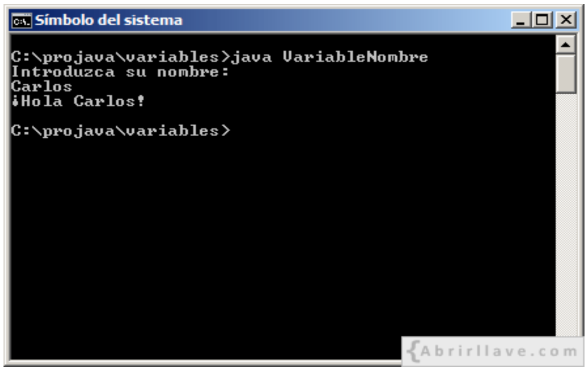
Step 3: Search for Creative solutions

Alternative 1. Simulation program (With graphical interface)



It is a Java program that simulates the entry and exit of customers to the video game store. Using data structures such as Queues (simulating the queue of customers), Stacks (simulating the basket and the users cart) and Hash Tables (simulating the shelves and customer lists), a rough simulation of the order of Exit of customers to the store based on the number of games, order of arrival and estimated times in the store of each of the customers, with an interface that allows an approximate view of the movements that the customer would make for each section graphically exemplifying the actual experience.

Alternative 2. Simulation program (By Console)



```
C:\projava\variables>java VariableNombre
Introduzca su nombre:
Carlos
¡Hola Carlos!

C:\projava\variables>
```

It is a program made in Java, with console input (plain text), which simulates the entry and exit of customers to the video game store. Using data structures such as Queues (simulating the queue of customers), Stacks (simulating the basket and the users cart) and Hash Tables (simulating the shelves and customer lists), a rough simulation of the order of customer departure to the store based on the number of games, order of arrival and estimated times in the store of each of the customers.

Alternative 3. Simulation program with a 3D view of the store.

It is a program made in Java that simulates, with a 3D visualization (in augmented reality), the entry and exit of customers to the video game store. Using data structures such as Queues (simulating the queue of customers), Stacks (simulating the basket and the users cart) and Hash Tables (simulating the shelves and customer lists), a rough simulation of the order of departure of customers to the store based on the number of games, order of arrival and estimated times in the store of each of the customers.With an interface that allows the user to see in an approximate way and in first person, the movements that would be carried out for each section graphically exemplifying the real experience of going to the store with the new and innovative service system.

Step 4: Transition from Ideas to Preliminary Designs

In this step, we discard the alternatives that are less feasible. Therefore, we discarded alternative 3. Simulation program with a 3D view of the store, since due to the difficulty of 3D simulation, its implementation may be imprecise and not provide an optimal solution to the problem.
The review of the other alternatives leads us to the following:

Alternative 1. Simulation program (With graphical interface)

Alternative 2. Simulation program (By Console)

Step 5: Evaluation and Selection of the Best Solution

- Criterion A: La alternativa es amigable con el usuario. La alternativa entrega una solución:
  - [2] Friendly
  - [1] Unfriendly
  - Criterio B: The alternative graphs the simulation in an understandable way. Deliver a solution:
  - [3] Completely understandable
  - [2] Fairly understandable
  - [1] Little understandable

Functionality criteria are left out since both solutions implement the same methods to simulate

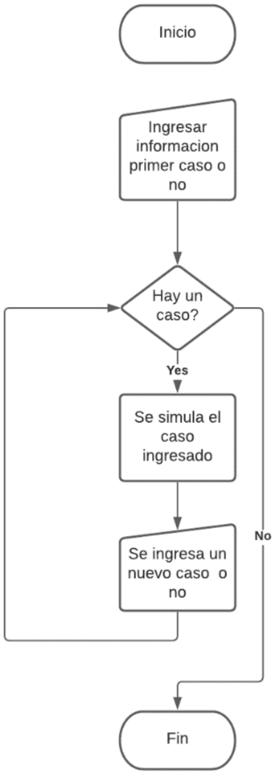
	Criterion A	Criterion B	Total
Alternative 1. Graphical interface	Friendly 2	Completely understandable 3	5
Alternative 2. No graphical interface	Not very friendly 1	Little understandable 1	2

Selection

According to the previous criteria evaluation, the simulation with a graphical interface should be implemented to make the simulation more understandable for the user.

Step 6: Preparation of Reports and Specifications

The specification, design and requirements documents that the program to implement has can be found in the docs folder of the github repository where the project is stored. (Make up a pseudocode with that diagram)


Inicio
Ingresar informacion primer caso o no
Hay un caso?
Yes
Se simula el caso ingresado
Se ingresa un nuevo caso o no
No
Fin

Step 7: Design Implementation

The implementation is in the github repository, it is made in Java code with Javafx.