DESIGN AND SIMULATION
COFFEE VENDING MACHIN

RUBEN MARTINEZ MARTINEZ

DEVELOPMENT ENVIROMENTS

Content

Introduction

The project is based on programming a coffee vending machine with various functionalities. The machine must be prepared to offer the different coffees available. First, enter the number of the coffee to be prepared. Afterwards, it is checked that there are stocks to be able to prepare it, when preparing it, it asks for the amount of sugar and checks if that amount of sugar is still available, if not, it asks the amount again. Finally, the money is introduced, and it checks whether it should return change or not (the maximum amount of money allowed is 20), once finished it returns the change (if it has to) and prepares the coffee, removing the spent stock to it.

Functions

Functions	Use	Type	Values
Main()	Main part of the program where the rest of the functions are called	**VOID**	String[] args
checkIfIngredienrs( )	Check that the ingredients to be used exist, if they are missing, return the first one that is missing	**STRING**	String[] ingredientsNames int[] ingredientsAmmount int[] ingredientsUsed
returnChange()	Calculate from the deposit money how much you have to return in bills and coins	**STRING**	int money
prepareCoffee()	Eliminates the ingredients used for its preparation and returns whether it could have been prepared or not	**BOOLEAN**	int[] ingredientsAmmount int[] ingredientsUsed int sugarAmmount

Software Requirements Specification

A machine to provide coffee without errors and in the most efficient and effective way, knowing the needs of the user.

System requirements

For its functionality, the machine must have the following requirements:

- Memory where to house the created program.
- Money, ingredient and replacement dispensers.
- Enough money to provide change.
- Ingredients necessary for the preparation.
- Electrical supplement for its operation.

Functional requirements

- Select the type of coffee to prepare.
- Select the amount of sugar.
- Insert amount of money.
- Return appropriate change.
- Verification of existence of ingredients.
- Coffee preparation.
- Restarting the machine due to inactivity.

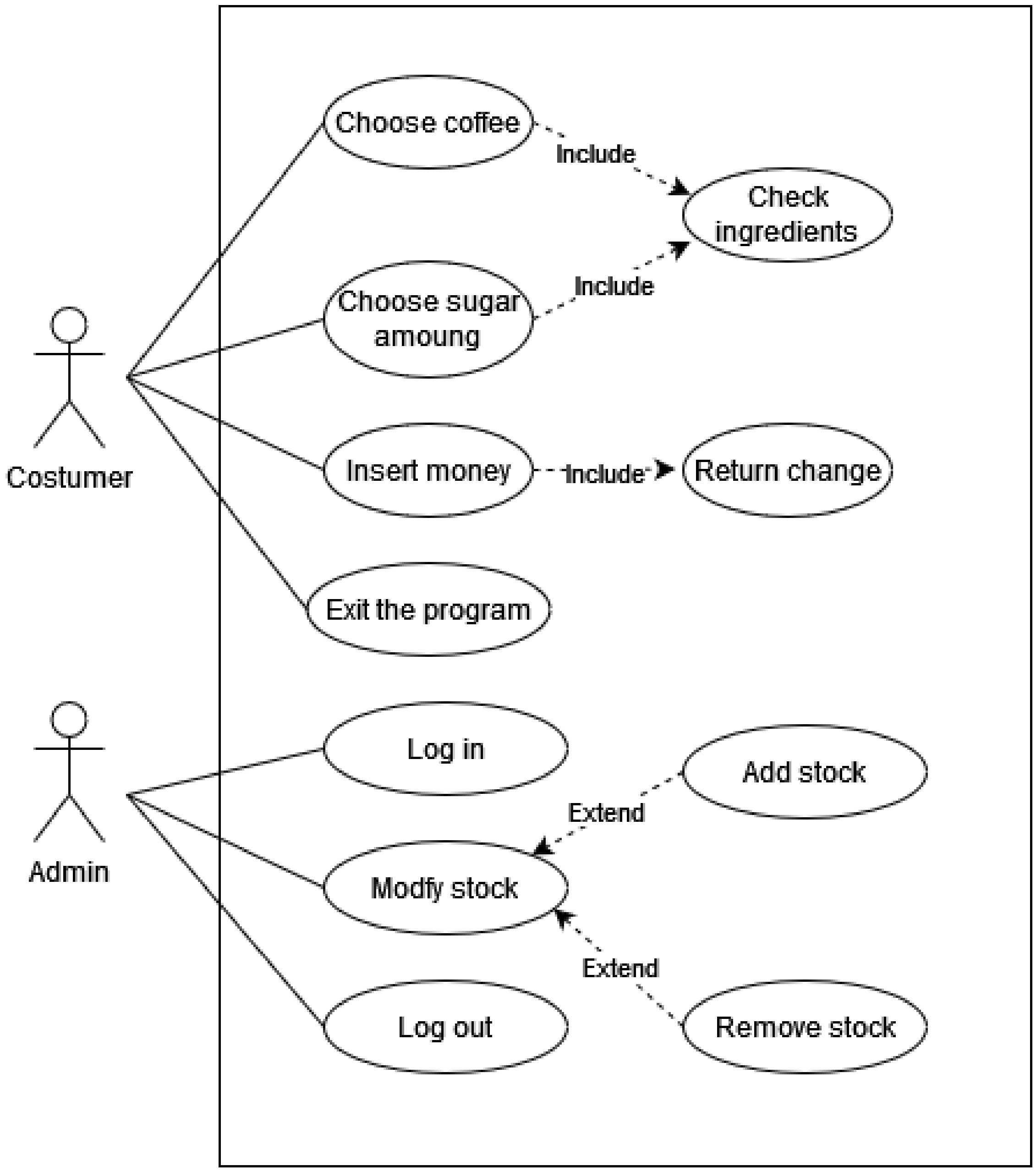
Non-functional requirements

- Preparation and reaction speed.
- Guarantee the minimum of errors and totally null in the return of the change.
- Security and vulnerability restrictions.
- Understandable user interface.
- Variety of possibilities.

Modelling and Diagrams

Use case model

In the use case model we can see the options that both type of users can do. The customer can select all kind of possible things for the coffee to be prepared, if any of them fails, is the customers choice to continue, change the option or leave it. The admin can only check the existances of the stock and add or remove stock from the machine.

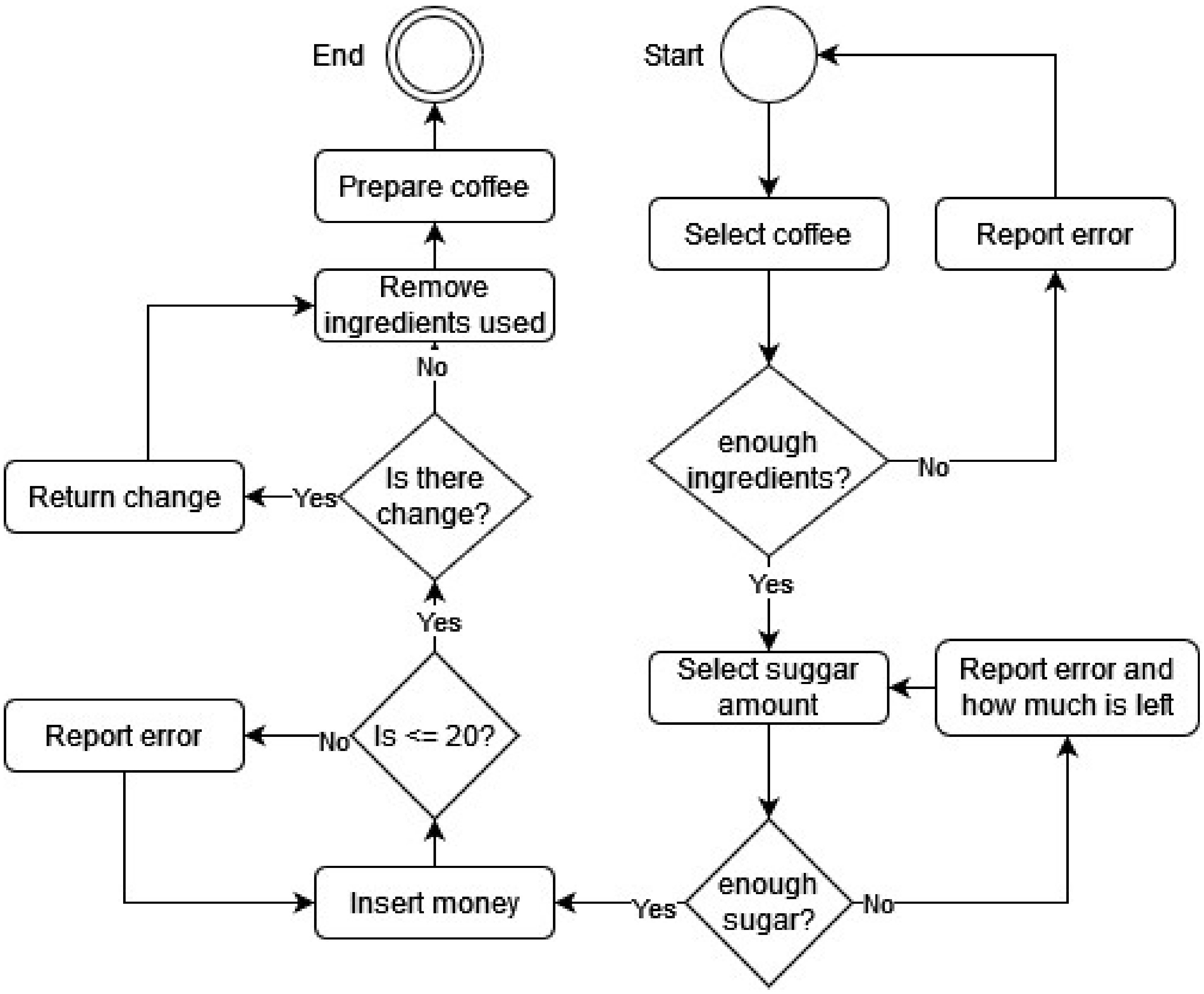


In this case we assume that the machine always has enough money for change, but its a good point to solve that.

Activity diagrams

Prepare coffee

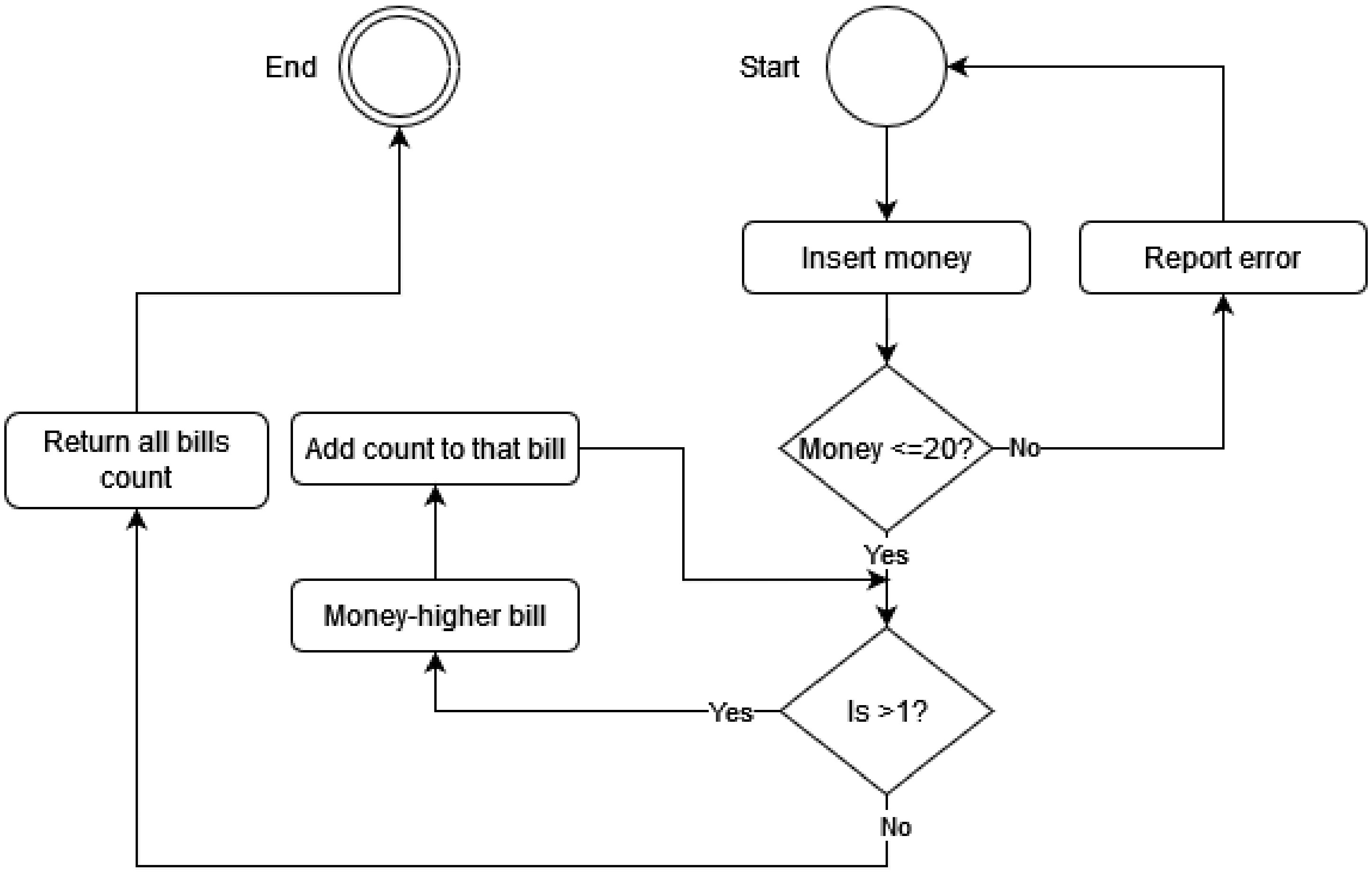
First of all we have the activity diagram for coffee preparation from when the user enters what they want until it is prepared. To cancel at any time during your preparation, you would have to wait 30 seconds for the machine to restart itself.



As you can see, the maximum amount of money that can be inserted is 20, and the change is returned in the largest possible bills and coins. As a detail to improve, the machine does not warn when there is not a ingredient, you must select a coffee that uses that specific ingredient.

Return change

As a second diagram we have the change return function, basically when the money is introduced it is checked that it does not exceed 20, for security reasons, if so it checks if the figure is greater than 1 (that is, the price of the coffee), in If it is, it subtracts the largest amount of change to be returned and adds a counter to later return that change. If it is still greater than 1, it will continue doing the same until it is equal to 1, then it returns all the counters as change.



If a single coin of 1 is entered, it only returns zeros, so there is no change.

Program

The program begins by asking the user to choose between the coffee options available, or they can exit the program. When you choose a coffee, it checks that all the ingredients exist, otherwise it asks for a different coffee again. If the coffee has all the ingredients, this time it asks for the amount of sugar, if there is not that amount of sugar, it returns an error and the amount of sugar available so you can enter the amount again. Finally, ask for the amount of money to be inserted, this money is checked and calculated to return the exact change, this amount cannot exceed 20. After all this, the used ingredients are eliminated and the coffee is prepared.

If you log in as an administrator, you can select the ingredient you want to modify it, either adding or removing it. The way you log as admin is typing 999, and then inserting the password (8426).

//As a note to improve, the change should be able to be modified so that the administrator can add or remove as appropriate.