UNIVERSIDAD ICESI

Algoritmos y Programación I
Entrega de laboratorio
Sebastián García Acosta

Problem Analysis

Content

Statement

Due to the growing awareness of the effects of human beings on the environment, various entities have begun to worry about helping to reduce the negative impact we have on the environment. Among those entities we find RECYCLES, the recycling staff wants to create an application that helps them train their employees and people in general on the proper management of waste, always trying to apply the 3R, Reuse, Recycle and Reduce.

The first thing you want to save is the information of the waste, of each waste you must indicate an identifier, a name, a source (industrial, residential, municipal, construction, hospital), a color, the time it takes to decompose (in days) and a product that produces it. There are different types of waste, and they are classified into different categories, the main categories are biodegradable, inert and recyclable. In addition to the characteristics of the aforementioned residues, the biodegradable ones want to know if it is suitable for composting or not. Recyclables want to know their type, the types can be: paper, cardboard, glass, plastics, metals. In addition to the type, it is desired to keep a description of what is the most appropriate way to make the disposition of these elements for homes and for industry. Inert residues are those that cannot be recycled and that take a long time to decompose, for example the icopor, of this type of waste you want to keep tips to reduce their use.

For each product associated with a waste, you want to keep an identifier, a name, and a description. It is expected that the system has the ability to search for information associated with a waste either by name or by the identifier of the product that produces it, for which the user must be shown a list of registered products. Keep in mind that a product can produce more than one residue.

All waste has the ability to calculate its harmful effect on the planet taking into account its origin, to make this calculation multiply the time it takes to decompose by a factor determined by its origin according to the following table:

Procedencia	**Factor**
Industrial	10%
Domiciliary	5%
Building	8%
Municipal	12%
Hospitallers	15%

All residues perform this calculation, however, if it is a recyclable waste, 2% is subtracted and if it is a biodegradable waste suitable for composting, 1% is subtracted.

Biodegradable and recyclable waste has an additional behavior regarding whether it is usable or not. To calculate if a biodegradable waste is usable, it must be met that the decomposition time is less than one year and is suitable for composting. For recyclable waste, it must be met that there is a description of the proper way to dispose of the waste.

Functional Requirements

NAME	FR1
DESCRIPTION	Allow to add a residue taking into account the type of waste, since it makes no sense to add a residue of any kind. It must be associated with a product that may or may not exist previously, if it does not exist it must allow the product to be added.
INPUTS	id, name, origin, color, decomposition days, product
OUTPUTS	Operation's result message

NAME	FR2
DESCRIPTION	Generate a report of the waste that is registered, where a title with the type of waste is displayed and the information of the waste of that type is listed.
INPUTS	Wastes array
OUTPUTS	Report string

NAME	FR3
DESCRIPTION	Allow to add a product and the waste it can generate.
INPUTS	id, name, description
OUTPUTS	Operation's result message

NAME	FR4
DESCRIPTION	Find the information of a waste by waste name
INPUTS	waste name
OUTPUTS	Waste's information if found; otherwise, report.

NAME	FR5
DESCRIPTION	Find the information of wastes by product id
INPUTS	Product id
OUTPUTS	Information of wastes (notice the plural reference) if found; otherwise, report.

NAME	FR6
DESCRIPTION	Show a list of registered products
INPUTS	None
OUTPUTS	Information of registered products.

NAME	FR7
DESCRIPTION	Calculate the harmful effect of a waste
INPUTS	Depends on the product
OUTPUTS	None.

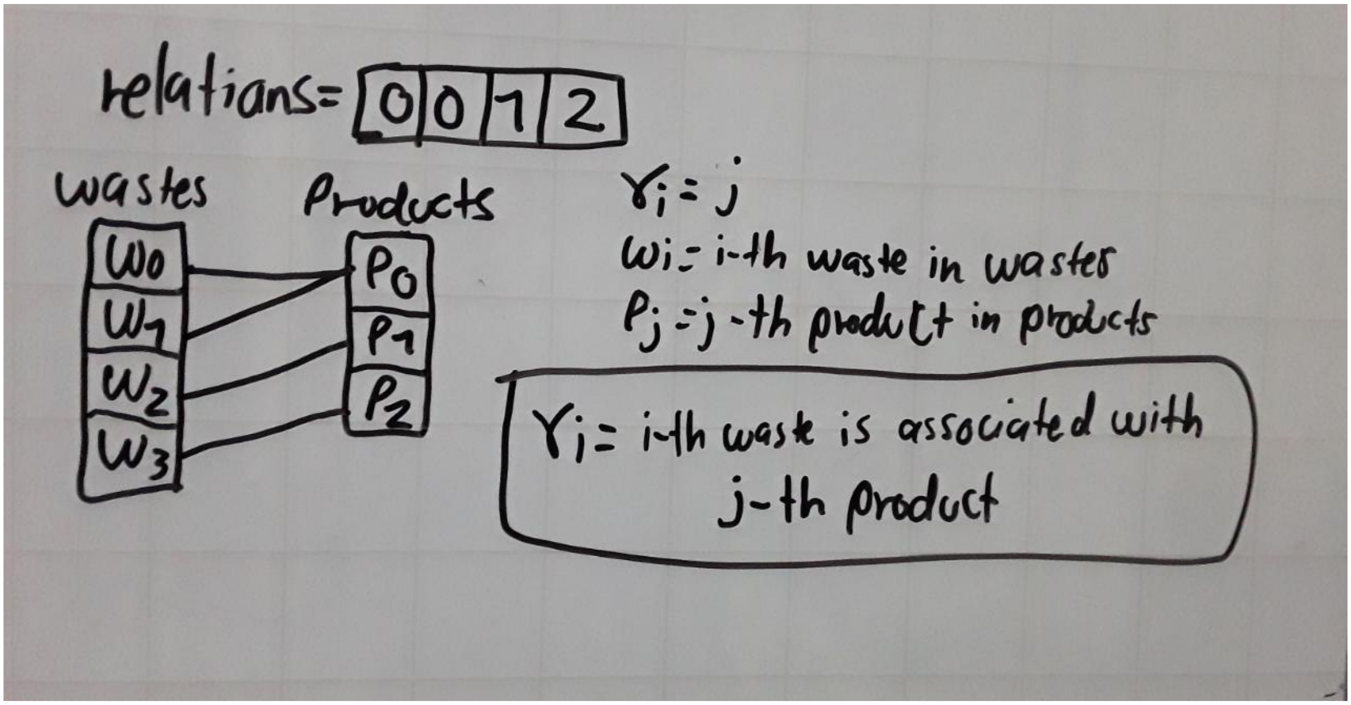
NAME	FR8
DESCRIPTION	Determine if a biodegradable or recyclable waste is usable.
INPUTS	Depends on the type of waste
OUTPUTS	None.

NAME	FR9
DESCRIPTION	List the waste of a product taking into account its harmful effects, showing first the most harmful.
INPUTS	Depends on the type of the wastes of the product
OUTPUTS	None.

Table 1 Functional Requirements

Problem Analysis and solution purpose

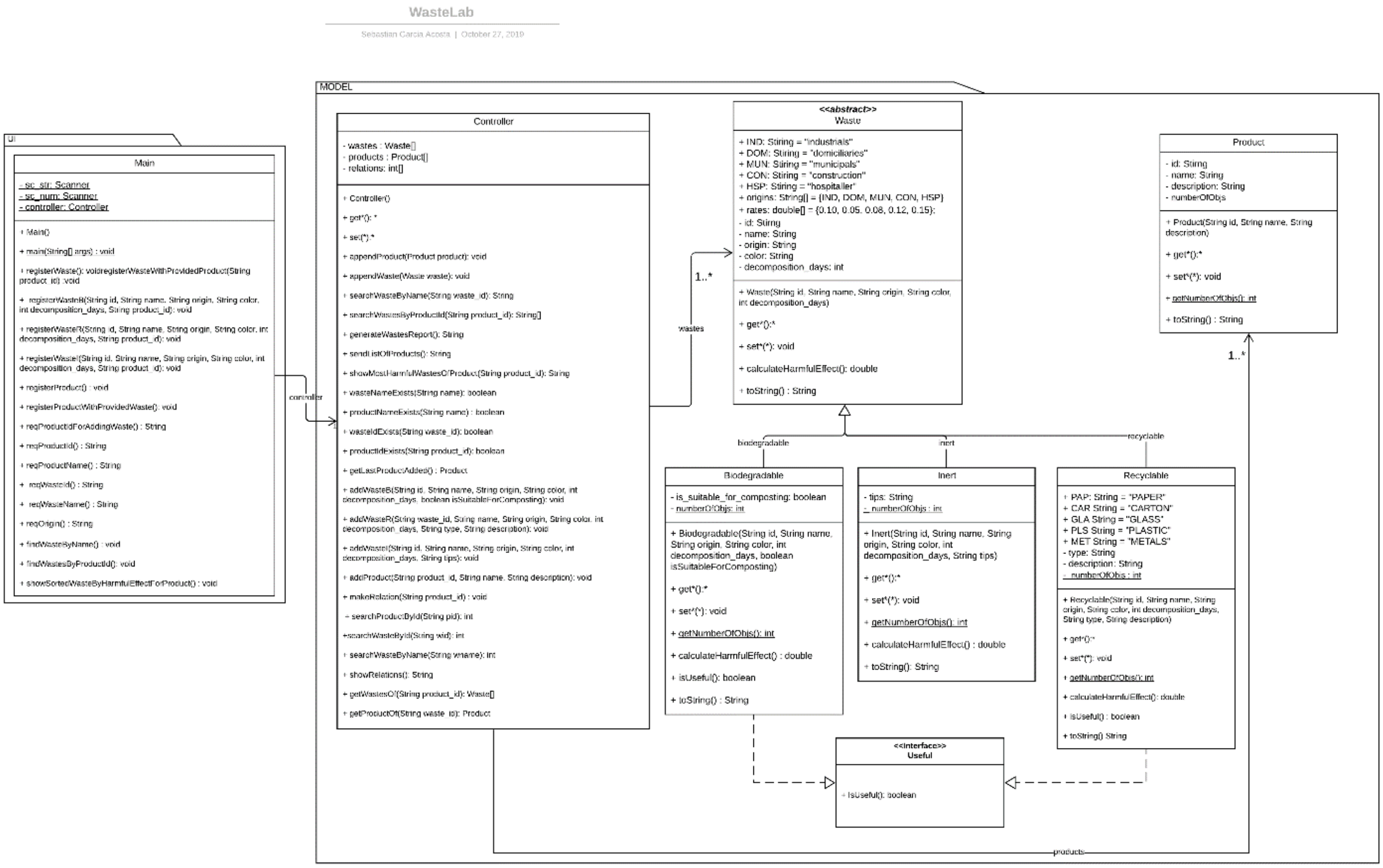
In this statement, the key problem is: how to stablish a relationship between product and wastes so that all products be associated with, at least, one waste; and all waste has only one product. The solution of this problem entails the solution of more than 50% of the work. So, the scheme used is as follows:


relations= 0 0 1 2
wastes
Products
W0
W1
W2
W3
P0
P1
P2
ri = j
wi := i-th waste in wastes
Pj := j-th product in products
ri = i-th waste is associated with j-th product

Let $r_i = j$; $w\_i = i_{th}\, waste$; $p_j = j_{th}\ \ product$. So that the i_{th} position in relations (r) means that $i_{th}\, waste$ is associated with the $j_{th}\ \ product$. This simple scheme makes that the rest of the architecture guarantees that:

a. I won’t be a paradox when trying to add a product or waste.
b. There won’t be direct associations between Product and Waste classes; hence, these relations will be handled in the Controller class.
c. A waste only must have one product and a product could have at least one waste associated.

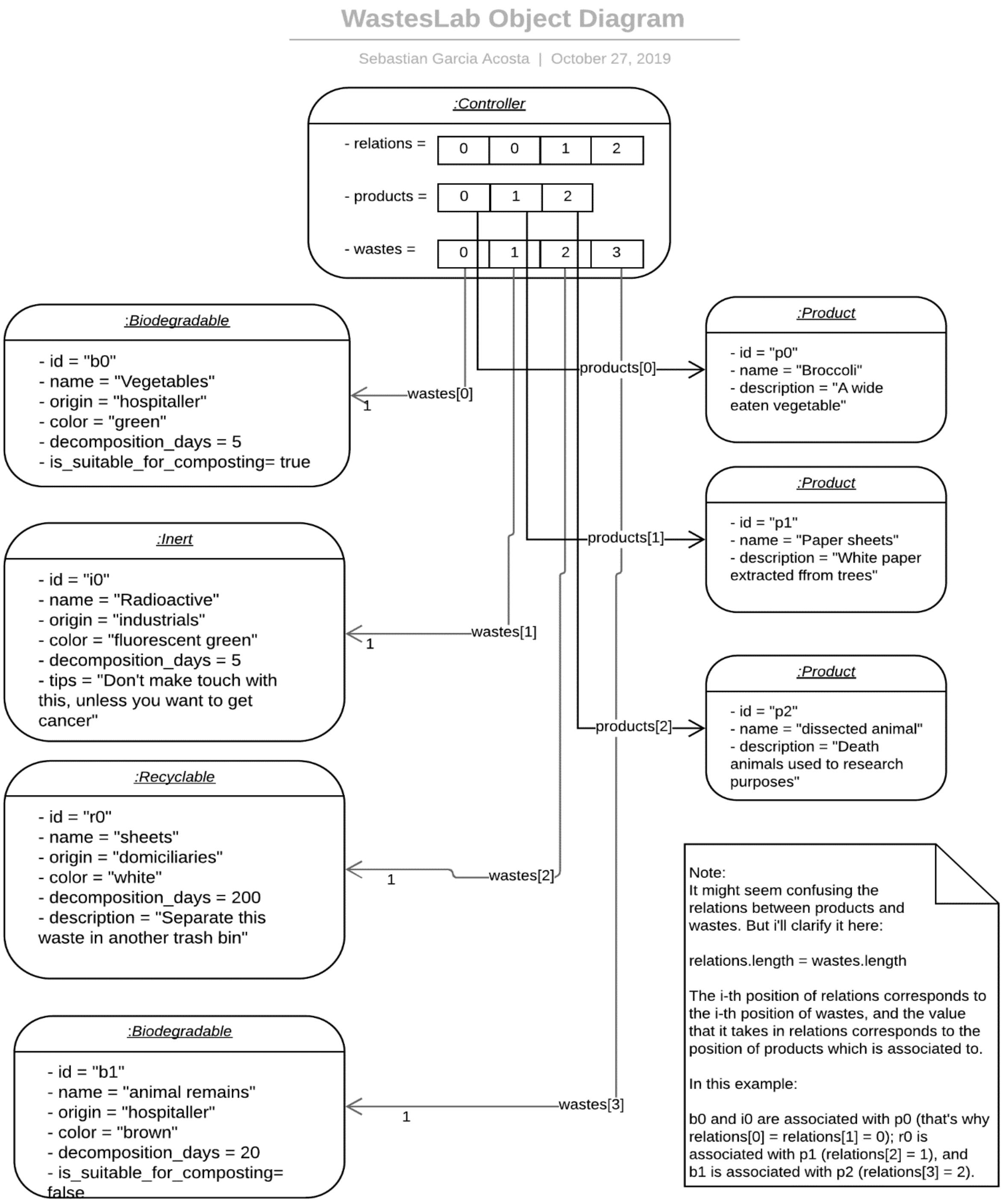
UML Class Diagram


WasteLab
Sebastian Garcia Acosta | October 27, 2019
UI
Main
- sc_str: Scanner
- sc_num: Scanner
- controller: Controller
+ Main()
+ main(String[] args) : void
+ registerWaste(): voidregisterWasteWithProvidedProduct(String product_id) :void
+ registerWasteB(String id, String name, String origin, String color, int decomposition_days, String product_id): void
+ registerWasteR(String id, String name, String origin, String color, int decomposition_days, String product_id): void
+ registerWasteI(String id, String name, String origin, String color, int decomposition_days, String product_id): void
+ registerProduct() : void
+ registerProductWithProvidedWaste(): void
+ reqProductIdForAddingWaste() : String
+ reqProductId() : String
+ reqProductName() : String
+ reqWasteId() : String
+ reqWasteName() : String
+ reqOrigin() : String
+ findWasteByName() : void
+ findWastesByProductId(): void
+ showSortedWasteByHarmfulEffectForProduct() : void
controller
MODEL
Controller
- wastes : Waste[]
- products : Product[]
- relations: int[]
+ Controller()
+ get*(): *
+ set(*):*
+ appendProduct(Product product): void
+ appendWaste(Waste waste): void
+ searchWasteByName(String waste_id): String
+ searchWastesByProductId(String product_id): String[]
+ generateWastesReport(): String
+ sendListOfProducts(): String
+ showMostHarmfulWastesOfProduct(String product_id): String
+ wasteNameExists(String name): boolean
+ productNameExists(String name) : boolean
+ wasteIdExists(String waste_id): boolean
+ productIdExists(String product_id): boolean
+ getLastProductAdded() : Product
+ addWasteB(String id, String name, String origin, String color, int decomposition_days, boolean isSuitableForComposting): void
+ addWasteR(String waste_id, String name, String origin, String color, int decomposition_days, String type, String description): void
+ addWasteI(String id, String name, String origin, String color, int decomposition_days, String tips): void
+ addProduct(String product_id, String name, String description): void
+ makeRelation(String product_id) : void
+ searchProductById(String pid): int
+searchWasteById(String wid): int
+ searchWasteByName(String wname): int
+ showRelations(): String
+ getWastesOf(String product_id): Waste[]
+ getProductOf(String waste_id): Product
wastes
1..*
<<abstract>>
Waste
+ IND: Stiring = "industrials"
+ DOM: Stiring = "domiciliaries"
+ MUN: Stiring = "municipals"
+ CON: Stiring = "construction"
+ HSP: Stiring = "hospitaller"
+ origins: String[] = {IND, DOM, MUN, CON, HSP}
+ rates: double[] = {0.10, 0.05. 0.08, 0.12, 0.15};
- id: Stirng
- name: String
- origin: String
- color: String
- decomposition_days: int
+ Waste(String id, String name, String origin, String color, int decomposition_days)
+ get*():*
+ set*(*): void
+ calculateHarmfulEffect(): double
+ toString() : String
Product
- id: Stirng
- name: String
- description: String
- numberOfObjs
+ Product(String id, String name, String description)
+ get*():*
+ set*(*): void
+ getNumberOfObjs(): int
+ toString() : String
1..*
biodegradable
inert
recyclable
Biodegradable
- is_suitable_for_composting: boolean
- numberOfObjs: int
+ Biodegradable(String id, String name, String origin, String color, int decomposition_days, boolean isSuitableForComposting)
+ get*():*
+ set*(*): void
+ getNumberOfObjs(): int
+ calculateHarmfulEffect() : double
+ isUseful(): boolean
+ toString() : String
Inert
- tips: String
- _numberOfObjs : int
+ Inert(String id, String name, String origin, String color, int decomposition_days, String tips)
+ get*():*
+ set*(*): void
+ getNumberOfObjs(): int
+ calculateHarmfulEffect() : double
+ toString(): String
Recyclable
+ PAP: String = "PAPER"
+ CAR String = "CARTON"
+ GLA String = "GLASS"
+ PLS String = "PLASTIC"
+ MET String = "METALS"
- type: String
- description: String
- _numberOfObjs : int
+ Recyclable(String id, String name, String origin, String color, int decomposition_days, String type, String description)
+ get*():*
+ set*(*): void
+ getNumberOfObjs(): int
+ calculateHarmfulEffect(): double
+ isUseful() : boolean
+ toString() String
<<Interface>>
Useful
+ IsUseful(): boolean
products

For more visibility, you can find this UML Class Diagram on lucidchart:

https://www.lucidchart.com/invitations/accept/56e0446a-683e-4b6c-a6b7-3ea93fe760de

Object diagram



WastesLab Object Diagram

Sebastian Garcia Acosta | October 27, 2019

:Controller

- relations = 0 | 0 | 1 | 2
- products = 0 | 1 | 2
- wastes = 0 | 1 | 2 | 3

:Biodegradable

- id = "b0"
- name = "Vegetables"
- origin = "hospitaller"
- color = "green"
- decomposition_days = 5
- is_suitable_for_composting= true

:Inert

- id = "i0"
- name = "Radioactive"
- origin = "industrials"
- color = "fluorescent green"
- decomposition_days = 5
- tips = "Don't make touch with this, unless you want to get cancer"

:Recyclable

- id = "r0"
- name = "sheets"
- origin = "domiciliaries"
- color = "white"
- decomposition_days = 200
- description = "Separate this waste in another trash bin"

:Biodegradable

- id = "b1"
- name = "animal remains"
- origin = "hospitaller"
- color = "brown"
- decomposition_days = 20
- is_suitable_for_composting= false

:Product

- id = "p0"
- name = "Broccoli"
- description = "A wide eaten vegetable"

:Product

- id = "p1"
- name = "Paper sheets"
- description = "White paper extracted ffrom trees"

:Product

- id = "p2"
- name = "dissected animal"
- description = "Death animals used to research purposes"

Note:
It might seem confusing the relations between products and wastes. But i'll clarify it here:

relations.length = wastes.length

The i-th position of relations corresponds to the i-th position of wastes, and the value that it takes in relations corresponds to the position of products which is associated to.

In this example:

b0 and i0 are associated with p0 (that's why relations[0] = relations[1] = 0); r0 is associated with p1 (relations[2] = 1), and b1 is associated with p2 (relations[3] = 2).

Algoritmos y Programación I
Entrega de laboratorio
Sebastián García Acosta

You can find this UML Object Diagram on lucidchart:

https://www.lucidchart.com/invitations/accept/c030f6c4-983d-49e2-8429-f72419288b96

Traceability

Functional requirement	Class	Method
FR1	Main	registerWaste()
FR2	Controller	showWastes()
FR3	Main	registerProductWithProvidedWaste()
FR4	Main	registerProductWithProvidedWaste()
FR5	Main	findWastesByProductId()
FR6	Main	showProducts()
FR7	Controller	calculateHarmfulEffect()
FR8	Controller	isUseful()
FR9	Controller	showSortedWasteByHarmfulEffectForProduct