Miguel Andrés Sarasti – A00364978

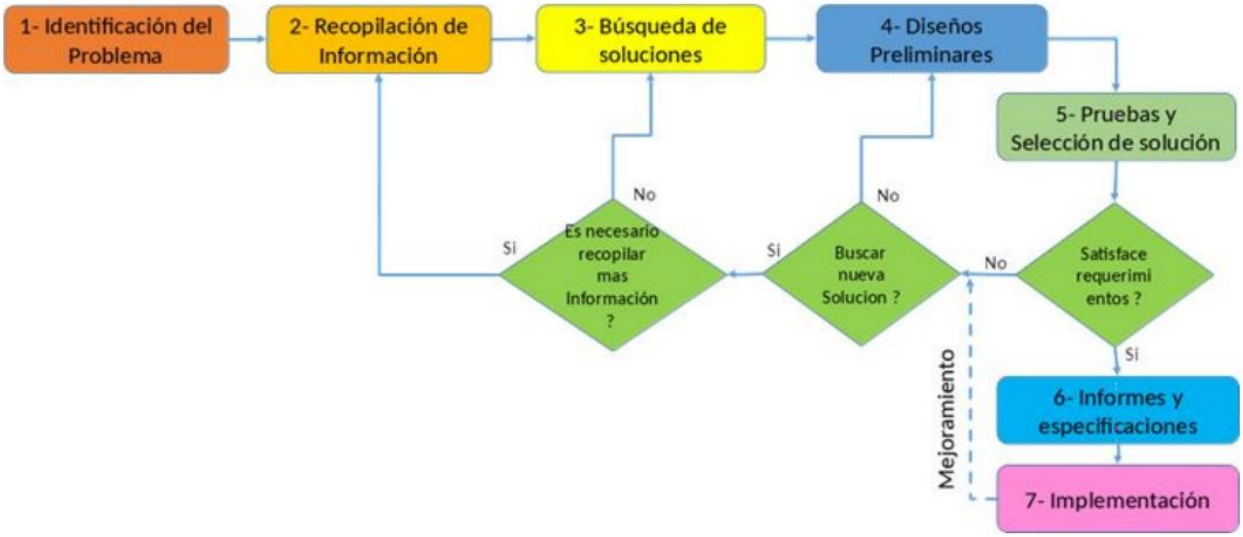
Método de la Ingeniería

Contexto problemático

El Equipo VIP de Simulación de la Universidad Icesi asignó un subproyecto que consiste en el desarrollo de un prototipo de software que permita gestionar eficientemente las operaciones CRUD (Crear, Leer, Modificar/Actualizar, Borrar) sobre una base de datos de personas del continente americano.

Desarrollo de la Solución

Seguiremos estos pasos mostrados en el siguiente diagrama de flujo para llegar al desarrollo de la solución:



Paso 1: Identificación del problema

Identificación de necesidades y síntomas

1. El proyecto necesita un software capaz de realizar las operaciones CRUD, con capacidad máxima de mil millones de personas.
2. Se busca que la generación de nombres completos sea por medio de dos dataset.
3. La fecha de nacimiento debe seguir una distribución de edad para toda América.
4. La estatura debe ser generada aleatoriamente en un intervalo lógico.
5. La nacionalidad debe mantener unos porcentajes relativos de la población de cada país respecto al continente.
6. El programa debe tener una barra de progreso e indicar cuanto tiempo se demoró la operación.
7. El programa debe guardar los datos, los datos deben ser persistentes.

Definición del Problema

El Equipo VIP requiere un programa capaz de realizar las operaciones CRUD sobre una base de datos de personas del continente americano, con una capacidad máxima de mil millones de personas.

Paso 2: Recopilación de la información

Definiciones

CRUD: son las cuatro funciones básicas del almacenamiento persistente, el acrónimo hace referencia a las funciones *crear*, *leer*, *modificar/actualizar* y *eliminar/borrar*.

TRIE:

Paso 3: Búsqueda de soluciones creativas

Paso 4: Transición de las Ideas a los Diseños Preliminares

Paso 5: Evaluación y Selección de la Mejor Solución

Paso 6: Preparación de informes y especificaciones

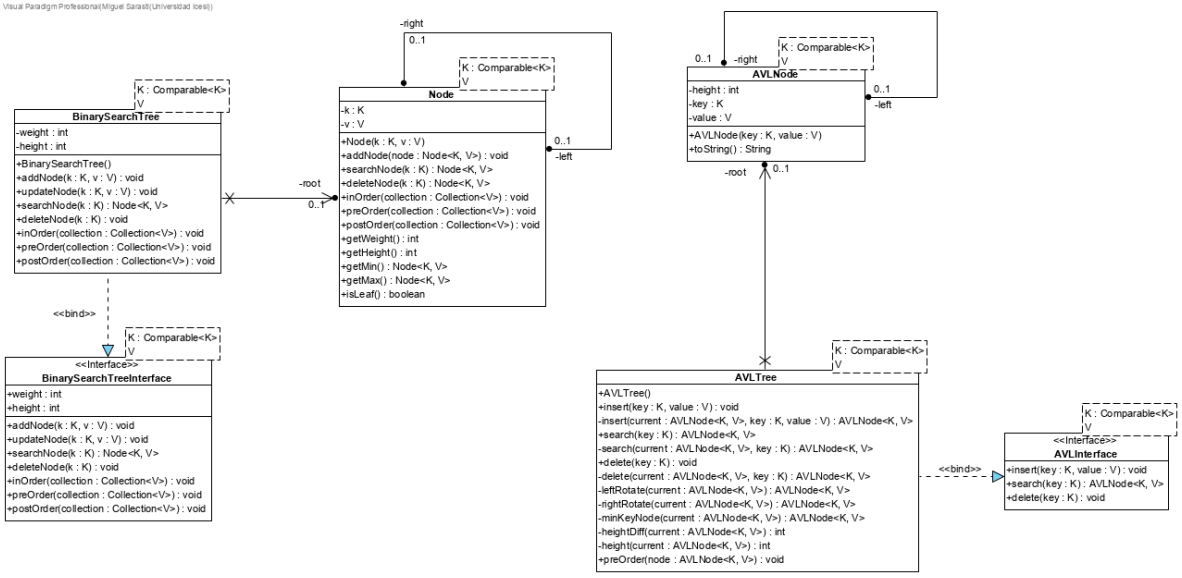


Ilustración 1 Diagrama de clases de BST y Árbol AVL

Paso 7: Implementación del diseño

Implementando en el lenguaje de programación Java.

Tareas por implementar:

Pruebas Unitarias

Árbol binario de búsqueda

Configuración de los Escenarios de BinarySearchTree:

Nombre	Clase	Escenario
setUpStage1	BSTTest	b = new BinarySearchTree<Integer, String>()

Nombre	Clase	Escenario
setUpStage2	BSTTest	b = new BinarySearchTree<Integer, String>() b.addNode(5, "Michael") b.addNode(10, "Leonard") b.addNode(7, "Raphael") b.addNode(2, "Donatello") b.addNode(1, "Splinter")

Diseño de Casos de Prueba:

Objetivo de la Prueba: Comprobar que el programa agregue nuevos elementos al BST				
Clase	Método	Escenario	Valores de Entrada	Resultado
BinarySearchTree	b.addNode()	setUpStage1	b.addNode(2, “Kirito”)	Kirito

Objetivo de la Prueba: Comprobar que el programa no agregue elementos al BST con una llave ya existente				
Clase	Método	Escenario	Valores de Entrada	Resultado
BinarySearchTree	b.addNode()	setUpStage1	b.addNode(2, "Kirito") b.addNode(1, "Asuna") b.addNode(6, "Sinon") b.addNode(1, “Alice”)	Exception

Objetivo de la Prueba: Comprobar que el programa permita modificar elementos existentes del BST				
Clase	Método	Escenario	Valores de Entrada	Resultado

BinarySearchTree	b.updateNode()	setUpStage2	b.updateNode(1, “Kirito”)	Kirito

Objetivo de la Prueba: Comprobar que el programa busque un elemento por medio de una llave en el BST.				
Clase	Método	Escenario	Valores de Entrada	Resultado
BinarySearchTree	b.searchNode()	setUpStage2		Michael

Objetivo de la Prueba: Comprobar que el programa elimine elementos correctamente del BST				
Clase	Método	Escenario	Valores de Entrada	Resultado
BinarySearchTree	b.deleteNode()	setUpStage2	b.deleteNode(2)	True

Objetivo de la Prueba: Comprobar que el programa calcule el peso del BST				
Clase	Método	Escenario	Valores de Entrada	Resultado
BinarySearchTree	b.getWeight()	setUpStage2		5

Objetivo de la Prueba: Comprobar que el programa calcule la altura del BST				
Clase	Método	Escenario	Valores de Entrada	Resultado
BinarySearchTree	b.getHeight()	setUpStage2		3

Árbol AVL

Configuración de los Escenarios de AVL Tree:

Nombre	Clase	Escenario
setUpStage1	AVLTTest	tree = new AVLTree<Integer, String>()

Nombre	Clase	Escenario
setUpStage2	AVLTTest	tree = new AVLTree<Integer, String>() tree.insert(10, "Hi") tree.insert(20, "Hello") tree.insert(30, "Nein") tree.insert(40, "Luck") tree.insert(50, "Be") tree.insert(25, "Die")

Diseño de Casos de Prueba:

Objetivo de la Prueba: Comprobar que el programa agregue nuevos elementos al árbol AVL				
Clase	Método	Escenario	Valores de Entrada	Resultado
AVLTree	tree.insert()	setUpStage1	tree.insert(2, "Kirito")	Kirito

Objetivo de la Prueba: Comprobar que el programa agregue elementos a un árbol AVL existente				
Clase	Método	Escenario	Valores de Entrada	Resultado
AVLTree	tree.insert()	setUpStage2	b.addNode(2, "Kirito") b.addNode(1, "Asuna") b.addNode(6, "Sinon")	Nein

Objetivo de la Prueba: Comprobar que el programa busque un elemento por medio de una llave en un árbol AVL existente				
Clase	Método	Escenario	Valores de Entrada	Resultado
AVLTree	tree.search()	setUpStage2		Hello

Objetivo de la Prueba: Comprobar que el programa busque un elemento agregado por medio de una llave en el BST.				
Clase	Método	Escenario	Valores de Entrada	Resultado
AVLTree	tree.insert() y tree.search()	setUpStage2	tree.insert(15, “Kirito”)	Kirito

Objetivo de la Prueba: Comprobar que el programa elimine elementos correctamente del árbol AVL				
Clase	Método	Escenario	Valores de Entrada	Resultado
AVLTree	tree.delete()	setUpStage2	tree.delete(25)	True

Objetivo de la Prueba: Comprobar que el programa elimine elementos correctamente del árbol AVL				
Clase	Método	Escenario	Valores de Entrada	Resultado
AVLTree	tree.delete()	setUpStage2	tree.delete(50)	True

Objetivo de la Prueba: Comprobar que el programa autobalancee correctamente al árbol AVL				
Clase	Método	Escenario	Valores de Entrada	Resultado
AVLTree	tree.insert() y tree.getRoot()	setUpStage1	tree.insert(1, "Kirito") tree.insert(2, "Asuna") tree.insert(3, "Sinon") tree.insert(10, "Eugeo");	Asuna

Objetivo de la Prueba: Comprobar que el programa autobalancee correctamente al árbol AVL				
Clase	Método	Escenario	Valores de Entrada	Resultado
AVLTree	tree.insert() y tree.getRoot()	setUpStage2	tree.insert(1, "Kirito") tree.insert(2, "Asuna") tree.insert(3, "Sinon") tree.insert(15, "Eugeo");	Nein