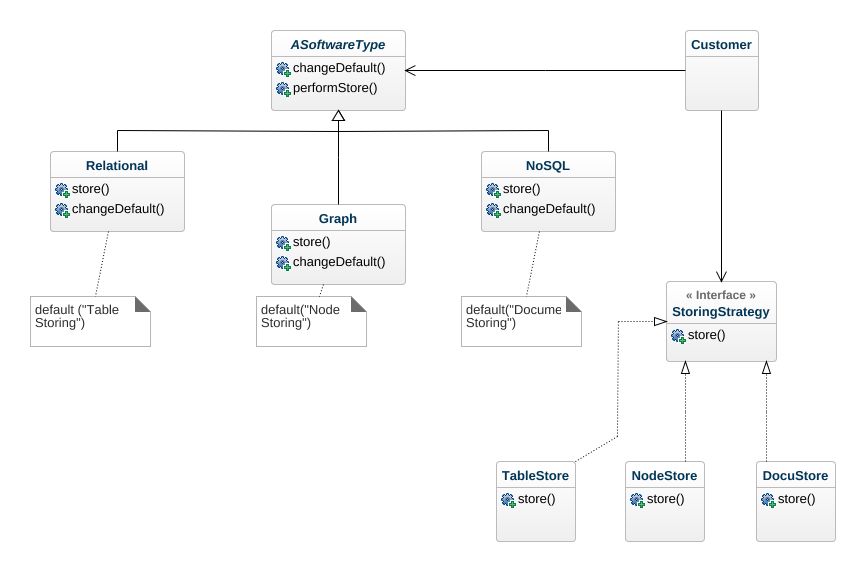
**Homework 2**

Software Engineering

Anna Jinneman and William Roberts

**UML Diagram:**

**Java Program Code:**

package esof322hw2strategymethod;

import java.util.Scanner;

/\*\*

\*

\* @author William Roberts & Anna Jinneman

\*

\*/

public class Esof322Hw2StrategyMethod {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

SoftwareType type;

StoringStrategy storeType;

Scanner scanner = new Scanner(System.in);

//We start this program off by asking the client what software type they would like to utilize:

type = useSoftwareType(scanner);

type.performStore("hi"); //Then we perform the store() with the default method of their softwareType

//Now we want to ask the client what storing method they would like to change the default to:

storeType = useStrategyMethod(scanner);

type.changeDefault(storeType); //Then change the Default to the perscribed method!

type.performStore("Hi"); //Final storing of data.

}

public static SoftwareType useSoftwareType(Scanner scanner)

{

SoftwareType type;

System.out.println("What software type would you like to use (NoSQL, Relational, or Graph)?");

String softwareType = scanner.nextLine();

if ("Relational".equals(softwareType)) {

type = new Relational();

System.out.println("-You've chosen Relational.");

}

else if("NoSQL".equals(softwareType))

{

type = new NoSQL();

System.out.println("-You've chosen NoSQL.");

}

else if("Graph".equals(softwareType))

{

type = new Graph();

System.out.println("-You've chosen Graph.");

}

else

{

type = new NoSQL();

System.out.println("-Okay, we've chosen NoSQL for you.");

}

return type;

}

public static StoringStrategy useStrategyMethod(Scanner scanner)

{

System.out.println("What storing method would you like to change to (TableStore, DocumentStore, or NodeStore)?");

String storingType = scanner.nextLine();

StoringStrategy storeType;

if ("TableStore".equals(storingType)) {

storeType = new TableStore();

System.out.println("-You've chosen TableStore.");

}

else if("DocumentStore".equals(storingType))

{

storeType = new DocuStore();

System.out.println("-You've chosen DocuStore.");

}

else if("NodeStore".equals(storingType))

{

storeType = new NodeStore();

System.out.println("-You've chosen NodeStore.");

}

else

{

storeType = new DocuStore();

System.out.println("-Okay, we've chosen docuStore for you.");

}

return storeType;

}

public interface StoringStrategy{ //The interface that controls all of our storing methods (like Node, Docu, or Table store

public void store(String data);

}

public abstract class SoftwareType //The abstract class of software types, ensures that each software type has these things

{

public StoringStrategy storeMethod;

public void performStore(String data){};

public void changeDefault(StoringStrategy newOne){};

}

// The first Storage Strategy, it uses "Document Store Method" to store data.

public static class NoSQL extends SoftwareType{

public StoringStrategy storeMethod;

NoSQL()

{

storeMethod = new DocuStore();

}

public void performStore(String data)

{

// Default to Document Store

//System.out.println("Document Store Method");

storeMethod.store(data);

}

public void changeDefault(StoringStrategy newOne){

storeMethod = newOne;

}

}

// Relational Storage Strategy uses "Table Store Method" to store the data.

public static class Relational extends SoftwareType{

public StoringStrategy storeMethod;

Relational()

{

storeMethod = new TableStore();

}

public void performStore(String data)

{

//Table Store

//System.out.println("Table Store Method");

storeMethod.store(data);

}

public void changeDefault(StoringStrategy newOne){

storeMethod = newOne;

}

}

//The last of the storage strategies, it uses "Node Store Method" to store the data.

public static class Graph extends SoftwareType{

public StoringStrategy storeMethod;

Graph()

{

storeMethod = new NodeStore();

}

public void performStore(String data)

{

//Node Store

//System.out.println("Node Store Method");

storeMethod.store(data);

}

public void changeDefault(StoringStrategy newOne){

storeMethod = newOne;

}

}

//Below are the different storing methods, they each override the interfaces store with their own procedure. (THEY ARE DUMMY METHODS)

public static class DocuStore implements StoringStrategy

{

public void store(String data){System.out.println("Document Store Method Call");}

}

public static class TableStore implements StoringStrategy

{

public void store(String data){System.out.println("Table Store Method Call");}

}

public static class NodeStore implements StoringStrategy

{

public void store(String data){System.out.println("Node Store Method Call");}

}

}

**Java Code Output:** (Only one example shown in output)

Output:

What software type would you like to use (NoSQL, Relational, or Graph)?

Graph

-You've chosen Graph.

Node Store Method Call

What storing method would you like to change to (TableStore, DocumentStore, or NodeStore)?

TableStore

-You've chosen TableStore.

Table Store Method Call

**UML Sequence Diagram:**