Т Е Х Н И Ч Е С К И У Н И В Е Р С И Т Е Т – В А Р Н А

инж. Димитричка Желева Николаева

ПОДХОД ЗА РАЗРАБОТКА НА СОФТУЕР, БАЗИРАН НА СОФТУЕРНИ ШАБЛОНИ ЗА ПРОЕКТИРАНЕ

А В Т О Р Е Ф Е Р А Т

на дисертация за получаване на образователната и научна степен „ДОКТОР”

по докторска програма: Системно програмиране

професионално направление: 5.3. Комуникационна и компютърна техника

Научен ръководител: Доц. д-р инж. ВИОЛЕТА БОЖИКОВА

Рецензенти:

1. .....

2. .....

Варна 2019 г.

Дисертационният труд е обсъден на 12.02.2020 г. в катедра „Компютърни науки и технологии“ на катедрен съвет и насочен за защита.

Автор: инж. Димитричка Желева Николаева

Заглавие: Подход за разработка на софтуер, базиран на софтуерни шаблони за проектиране

Т Е Х Н И Ч Е С К И У Н И В Е Р С И Т Е Т – В А Р Н А

инж. Димитричка Желева Николаева

ПОДХОД ЗА РАЗРАБОТКА НА СОФТУЕР, БАЗИРАН НА СОФТУЕРНИ ШАБЛОНИ ЗА ПРОЕКТИРАНЕ

А В Т О Р Е Ф Е Р А Т

на дисертация за получаване на образователната и научна степен „ДОКТОР”

Варна 2019 г.

Дисертационният труд съдържа 246 страници, включително 40 фигури, 27 таблици, 38 формули, оформени в 5 глави, приноси на дисертационния труд, списък с публикациите на автора по темата на дисертационния труд и списък на използваната литература от 240 заглавия, от които 7 на кирилица и 233 на латиница.

Защитата на дисертационния труд ще се състои на ..... г. от ..... ч. в ..... на открито заседание на жури сформирано със заповед на Ректора № ...../..... г.

Материалите по защитата (дисертацията, рецензиите и становищата) са на разположение на интересуващите се в Докторантски център, стая 318 НУК.

ХАРАКТЕРИСТИКА НА ДИСЕРТАЦИОННИЯ ТРУД

1. Актуалност на проблема

С увеличаване сложността на софтуерните системи нараства и необходимостта от подходи, които да автоматизират процеса по създаването им, да намалят времето и средствата за тяхната разработка. Едно иновативно решение на този проблем е използването на софтуерни шаблони за проектиране (software design patters). Тъй като софтуерните шаблони се фокусират върху ранния етап на проектиране, са подходящи за адаптиране в информационните системи, с което подобряват дизайна на приложението, като го правят по-лесен за разбиране, по-лесен за отстраняване на грешки и позволяват кода да бъде по-поддържан и използван многократно. Прилагането на софтуерните шаблони в сложни системи, ще доведе до подобряване на качеството и надеждността на софтуерните системи .

Създаването на нови подходи базирани на софтуерните шаблони, които са готово ефективно решение за създаване на софтуер са важна част в процеса на реинженеринг, където от съществено значение е лесната ориентация в изходния програмен код, а в частност и идентифициране на софтуерните шаблони в него.

2. Цел и задачи на изследването

Целта на дисертацията е да се предложат ефективни подходи за разработка на софтуер, базирани на софтуерни шаблони за проектиране.

За изпълнение на целта са поставени следните **задачи**:

1. Да се предложи обобщена класификация на шаблоните за проектиране.
2. Да се предложи ефективен подход за разработка на софтуер, базиран на комбинация от софтуерни шаблони за проектиране.
3. Да се предложи подход за откриване на софтуерни шаблони за проектиране в изходен програмен код.
4. Да се направи оценка на ефективността на предложените подходи.

3. Обект и предмет на изследване

Обект на изследване са шаблоните за проектиране, които описват модели и алгоритми за решаване на различни проблеми на разработката на софтуер, и възможностите за прилагането им. На тази основа предмет на дисертацията е създаване на подход за разработка на софтуер, базиран на комбинация от софтуерни шаблони за проектиране и подход за идентифициране на софтуерни шаблони за проектиране в изходен програмен код.

4. Методи на изследване

В дисертационния труд, са използвани научно-изследователски методи като системен, сравнителен, исторически, икономически анализи и моделиране. За апробиране на резултатите от научното изследване са използвани техники за моделиране, проектиране и прототипиране на софтуерните системи.

5. Място на изследване

Изследванията са проведени в лабораториите на катедра СИТ и КНТ при ТУ – Варна.

6. Научна новост на изследването

Направена е обобщена класификация на шаблоните за проектиране. Създадени са подход за разработване на софтуер, базиран на комбинация от софтуерни шаблони за проектиране и подход за откриване на софтуерни шаблони в изходен програмен код. Разработена е информационна система за целите на морския транспорт, прилагаща първия подход и приложение за откриване на шаблони от групата на GoF (Сингълтън) в изходен програмен код, прилагащо втория подход. Предимството на първия подход е, че дава възможност за автоматизиране на програмисткия труд, чрез преизползване на програмен код, а вторият подход е обратна инженерна дейност и може да се използва както за прогнозиране качеството на софтуерната система, така и в процесите на поддръжка и еволюция.

7. Практическа ценност на изследването

Към резултатите с приложна насоченост могат да се посочат експерименталните данни и зависимости, получени след измерване на пет софтуерни метрики, направени в две конверсии преди и след прилагане на шаблоните на проектиране, върху разработена за целите на дисертацията тестова, информационна система за морския транспорт, както и оценка на приложение за идентифициране на шаблони за проектиране в изходен програмен код. Получените резултати констатират подобряване на основни качествени характеристики на кода: повишаване експлоатационната характеристика на софтуера, а именно индекса на поддържаемостта, намаляване времето за разработка на софтуера, в резултат от преизползване на програмен код.

8. Апробация на изследването

Основните резултати от изследванията са докладвани и публикувани в следните научни форуми и издания:

Конференции:

- XXIV Международна научна конференция „Електронна техника – ЕТ 2015” 15-17 септември 2015 г. Созопол
- 16th Conference on Electrical Machines, Drives and Power Systems (ELMA), IEEE, 2019
- XXVIII International Scientific Conference Electronics – ET 2019 12 - 14 September 2019, IEEE, Sozopol

Списания:

- УДК 004.4'2. Електронно издание. Секция 3. Информационные технологии в управлении, автоматизации и мехатронике, Международной научно-технической конференции молодых ученых, аспирантов и студентов (Севастополь, 13-15 мая 2015 г.), pp.246-250
- XI Международно научно-техническа конференция „Стратегия на качеството в индустрията и образованието”, Юни 01-05, 2015 г., ТУ-Варна (Украйна, Русия, България), том 1, Направление „Качество в образованието”, pp.470-475
- Годишник на ТУ-Варна, V-та научна конференция с международно участие "Компютърни науки и технологии", 28 - 29 септември 2018 г., Варна, България, том 2, стр.119-126

9. Публикации по дисертационния труд

Основните етапи от разработването на дисертационния труд са отразени в 6 публикации, списък на които е приложен в края на автореферата.

СЪДЪРЖАНИЕ НА ДИСЕРТАЦИОННИЯ ТРУД

ГЛАВА ПЪРВА. АНАЛИЗ НА СОФТУЕРНИ ШАБЛОНИ ЗА ПРОЕКТИРАНЕ

Първа глава изследва и анализира текущото състояние в областта на софтуерните шаблони за проектиране. Представя систематично проучване на софтуерни шаблони за проектиране; дефинира понятието софтуерен шаблон за проектиране; представя стандартен формат за описание на софтуерен шаблон; обобщава техники, подходи и методи за ефективно прилагане на софтуерните шаблони за проектиране; посочва предимства и недостатъци на софтуерните шаблони за проектиране; представя изследване на софтуерните шаблони за проектиране по различни критерии (цел за прилагане на софтуерни шаблони за проектиране, употреба на софтуерни шаблони за проектиране и използвани програмни езици за реализирането им).

ИЗВОДИ КЪМ ГЛАВА ПЪРВА

В настоящата глава са посочени особеностите на СШП и обхвата на приложението им, като са представени проблемите, свързани с прилагането на СШП. Направен е обзор на съществуващите СШП. На тази основа са формулирани следните изводи:

- СШ са актуална проблемна област със засилен научен и практически интерес в последните 25 години. За последните 9 години от 2011 до 2019 година (фиг.1.1.) с 2,6 пъти е увеличен броят на издадената литература в областта на СШ в сравнение с периода 1995-2002 година.
- СШ имат теоретично и практико-приложно значение, доказателство за това са предимствата и разнообразието от методи, подходи и техники, използвани за създаване на качествен, надежден и лесен за поддръжка софтуер. В този контекст използването на СШП и тяхната ефективна комбинация са от съществено значение.
- Най-често целите от прилагане на СШП са свързани или с идентифициране на софтуерните шаблони за проектиране в програмен код или с постигане на надеждно функциониране. Най-използвани са GoF софтуерни шаблони за проектиране от поведенческата група - 45%, след това са GoF шаблони от структурната група - 31% и GoF шаблони от създаващата група - 24% (фиг.1.23.). Най-използвани в софтуерните разработки са: шаблон Наблюдател от поведенческата група, шаблон Адаптер от структурната група и шаблон Абстрактна Фабрика от създаващата група.
- Софтуерните шаблони са инструмент, който предоставя възможност за многократно използване на софтуер, който намалява времето, цената за разработка и поддръжка на софтуера и автоматизира програмисткия труд.
- Анализът в областта на софтуерните шаблонни, показа наличие на голям брой шаблони за проектиране и липса на единна класификация, която да ги обобщи.
- Създаването на нови подходи, които ще използват комбинация от софтуерни шаблони за проектиране и ще откриват софтуерни шаблони за проектиране в изходен програмен код е от важно значение за повишаване на качеството на софтуера и ефективността на програмисткия труд.

ГЛАВА ВТОРА. КЛАСИФИКАЦИЯ НА ШАБЛОНИТЕ ЗА ПРОЕКТИРАНЕ

Във втора глава е направена класификация на шаблоните за проектиране според различни признаци: предназначение, начин на използване, техники за прилагане и подходи за идентифициране на СШП.

Разгледаните класификационни признаци, могат да се обобщят в три нива на детайлизация (фиг. 2.20., Таблица 2.6. и Таблица 2.7.). Таблица 2.6. представя второ ниво на детайлизация:

Таблица 2.6. Класификация на ШП (описание по групи и подгрупи) – второ ниво на детайлизация

Ниво на детайлизация	Класификационен признак
К.1.	**Шаблони за проектиране според предназначението**
К.1.1.	Софтуерни Шаблони (Design Patterns)
К.1.2.	Core J2EE шаблони (Core J2EE Patterrns)
К.1.3.	Шаблони за корпоративно приложение (Patterns of Enterprise Application Architecture)
К.1.4.	Облачни шаблони за проектиране (Cloud Design Patterns)
К.1.5.	Анти-шаблони (Antipatterns)
К.1.6.	Архитектурни шаблони (Architectural Design Patterns)
К.1.7.	Шаблони за обектно-ориентиран софтуерен дизайн (Patterns for Object Oriented Software Design)
К.1.8.	Други шаблони за проектиране (Other Design Patterns)
К.1.9.	Шаблонно-ориентирана софтуерна архитектура, Шаблони за конкурентни и мрежови обекти (Pattern-Oriented Software Architecture, Patterns for Concurrent and Networked Objects)
К.1.10.	Хардуерни шаблони, Софтуерни шаблони, Комбинация от хардуерни и софтуерни шаблони (Hardware Patterns, Software Patterns, Combination of Hardware and Software Patterns)
К.1.11.	Каталог на микро шаблони в Java (Micro Pattern Catalog в Java)
К.1.12.	Шаблони за сигурност (Security patterns)
К.1.13.	Софтуерни шаблони за услуги, Основни софтуерни решения за SOAP / WSDL и RESTful Web Services (Service Design Patterns Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services)
К.1.14.	Шаблонни езици (Pattern Languages of Program Design 5)
К.2.	**Софтуерни шаблони според начина на използване**
К.2.1.	***Самостоятелно използвани СШ***
К.2.2.	***Комбинация от СШ***
К.2.2.1.	***Според релациите между СШ***
К.2.2.1.1.	Шаблон X използва Шаблон Y в решението си
К.2.2.1.2.	Шаблон Y използва Шаблон X в решението си
К.2.2.1.3.	Шаблона X може да се комбинира с Шаблона Y
К.2.2.2.	***Според сходството между СШ***
К.2.2.2.1.	Метод Фабрика (Factory Method), Абстрактна Фабрика (Abstract Factory), Строител (Builder)
К.2.2.2.2.	Сингълтън (Singleton), статичен клас (static class)
К.2.2.2.3.	Абстрактна фабрика (Abstract Factory), Сингълтън (Singleton), Метод Фабрика (Factory Mehod)
К.2.2.2.4.	Прототип (Prototype), Абстрактна Фабрика (Abstract Factory), Обектен пул (Pool Obje
К.2.2.2.5.	Мултитонен (обобщаващ) Сингълтън шаблон/ Сингълтън регистър (Miltition/ Registry of Singleton), Сингълтън пул (Pool Singleton), Сингълтън (Singleton) и Обектен пул (Pool Object)
К.2.2.2.6.	Адаптер (Adapter), Фасада (Façade)

Ниво на детайлизация	Класификационен признак
К.2.2.3.	***Според припокриването между СШ***
К.2.2.3.1.	Комбинация от припокриващи се СШ
К.2.2.3.2.	Консервативна комбинация от СШ
К.2.2.3.	***Според взаимодействието със СШ или влагане на СШ***
К.3.	**Софтуерни шаблони според използваните техники**
К.3.1.	***Програмиране, осигуряващо надеждно функциониране на програмните системи (ПОНФ)***
К.3.1.1.	Софтуер с приемливо ниво на грешки (Fault tolerant)
К.3.1.2.	Софтуер с минимизиране на програмните грешки (Fault avoidance)
К.3.1.3.	Защитно програмиране (Defensive programming)
К.3.2.	***Повторно използване в софтуерното производство (ПИ)***
К.3.2.1.	Проектиране на библиотеки за преизползване
К.3.2.2.	Създаване на компоненти за преизползване
К.3.2.3.	Създаване на стандарти за интегрирано използване на библиотеки
К.3.2.4.	Създаване на модели за процеса на преизползване и на инструментални средства за поддръжката им
К.4.	**Софтуерни шаблони според подходите за идентифицирането им**
К.4.1.	***Според методологията и аналитичния стил***
К.4.1.1.	Подходи чрез заявки към базата данни (Database-Query Approaches)
К.4.1.2.	Подходи базирани на метрики (Metrics-Based Approaches)
К.4.1.3.	UML структурни, графични и матрични базирани подходи (UML Structure, Graph and Matrix-Based Approaches)
К.4.1.4.	Смесени подходи (Miscellaneous Approaches)
К.4.2.	***Според методите за възстановяване на СШ***
К.4.2.1.	Според вида анализ за възстановяване на модели от изходния код на различни стари приложения
К.4.2.2.	Методи за откриване според вида на изходния код
К.4.3.	***Според идентифицираните СШ***

ИЗВОДИ КЪМ ГЛАВА ВТОРА

- Класификационните признаци са от особенно важно значение, както за описание на предназначението на конкретния шаблон, принадлежността му към определена шаблонна група, но също така и за определяне на неговите зависимости (сходства, взаимодействие с други шаблони и възможност за комбинирано използване) спрямо други шаблони от същата или друга група.
- Класифицирането на софтуерните шаблони според подходите за идентифицирането им ще доведе до тяхното ефективно прилагане в процеса на обратна инженерна дейност.
- Направена е обобщена класификация на шаблоните за проектиране, които са групирани по следните основни признаци според: предназначение, начин на използване, използвани техники и подходи за идентифициране на софтуерните шаблони. Това е начин за структуриране на многообразието от шаблони за проектиране и своеобразен каталог, който би улеснил тяхното изучаване, разбиране и прилагане.

ГЛАВА ТРЕТА. ПОДХОД ЗА РАЗРАБОТКА НА СОФТУЕР, БАЗИРАН НА КОМБИНАЦИЯ ОТ СОФТУЕРНИ ШАБЛОНИ ЗА ПРОЕКТИРАНЕ И ПОДХОД ЗА ОТКРИВАНЕ НА СОФТУЕРНИ ШАБЛОНИ В ИЗХОДЕН ПРОГРАМЕН КОД

Глава трета представя нов подход за разработване на софтуер, базиран на комбинация от софтуерни шаблони за проектиране (ПБКСШП) и подход за откриване на софтуерни шаблони в изходен програмен код (ПОСШП). Направен е избор на комбинация от софтуерни шаблони за създаване на ПБКСШП. Представен и описан е алгоритъм на ПБКСШП; представена е диаграма с комбинация от СШП използвани за реализиране на подхода. Представен и описан е подход за откриване на СШ за проектиране в изходен програмен код (ПОСШП). Направени са изводи относно прилагането на ПБКСШП и ПОСШП.

3.1. Подход за решаване на един клас софтуерни системи

В настоящият дисертационен труд се предлага подход за разработка на софтуер базиран на комбинация от СШП. Софтуерът е от тип Информационна система (ИС) със следната функционалност:

- работа с меню;
- изпращане на мейли;
- криптиране и декриптиране на данни;
- достъп до Релационна база данни (РБД);
- въвеждане и извършване на изчислителни операции;
- въвеждане и модифициране на данни в РБД;
- генериране на отчети.

За всяка една от тези **функционалности** се предлагат следните СШП:

- За работа с меню – шаблон Команда (A, Фиг.3.3.);
- За изпращане на мейли – шаблон Фасада (B, Фиг.3.4.);
- За криптиране и декриптиране на данни – шаблон Фабричен Метод (C, Фиг.3.9.);
- За достъп до РБД – шаблон Сингълтън (D, Фиг.3.7.);
- За въвеждане и извършване на изчислителни операции – шаблон Фасада (E, Фиг.3.6.);
- За въвеждане и модифициране на данни в РБД – шаблон Фабричен Метод (F, Фиг.3.8.);
- За генериране на отчети – шаблон Фасада (G, Фиг.3.5.)

За реализиране на конкретните функционалности на практика съществуват и други шаблонни решения (Таблица 3.1.)

Таблица 3.1. Софтуерни шаблонни решения за 7 типа функционалности (A-G)

Функционалност/ СШП	За работа с менюта	За изпращане на мейли	За кодиране и декодиране на данни	За достъп до РБД	За въвеждане и изчисляване	За въвеждане и модифициране на данни в РБД	За генериране на отчети
Команда	1						
	2						
Верига отговорности		2					
Абстрактна Фабрика							
				2			
Метод Фабрика			1			1	
				2		2	
Прокси							
							2
Фасада		1			1		1
						2	
Сингълтън				1			
Наблюдател							
						2	

шаблони прилагани в ПБКСШП

шаблони прилагани в практиката за реализиране на функционалности от A-G

Практиката показва, че има и други подобни решения, за разработка на такива ИС за този клас задачи. Извадка от информационни системи прилагащи софтуерни шаблони е показана в Таблица 3.2., като е съпоставена с ПБКСШП (Z1). За разликата от останалите ИС, които използват комбинация от СШ за решаване само на конкретен проблем, възникнал в процеса на реализиране на ИС, то ПБКСШП обхваща реализацията на цялата ИС, като тя се базира единствено и само на тази комбинация от СШП, за изпълнение на посочените по-горе функционалности.

Таблица 3.2. Сравнителен анализ между ИС, с използване на комбинации от СШ

Статия №/ СШ	*1*	*2*	*3*	*4*	*5*	*5*	*5*	*5*	*5*	*5*	*6*	**Z1**
1	X								X		X	
2											X	**X**
3	X										X	
4									X		X	**X**
5						X						
6						X						
7												**X**
8						X				X		
9	X						X					**X**
10				X								

Статия №/ СШ	*1*	*2*	*3*	*4*	*5*	*5*	*5*	*5*	*5*	*5*	*6*	**Z1**
11	X											
12		X			X			X				
13							X	X				
14							X			X		
15	X			X							X	
16	X											
17			X									
Z2	***1***	***2***	***3***	***4***	***5***	***5***	***5***	***5***	***5***	***5***	***6***	***7***

Легенда: Z1 – ПБКСШП, Z2 – Прилагане на СШП в ИС

Абстрактна Фабрика[1], Фактори Метод[2], Строител[3], Сингълтън[4], Адаптер[5], Композиция[6], Фасада[7], Прокси[8], Команда[9], Итератор[10], Медиатор[11], Посетител[12], Стратегия[13], Шаблонен Метод[14], MVC[15], (Шаблон на състоянието) Statechart pattern[16], Модели на сигурност - модели за контрол на достъпа (Security Patterns - access control patterns)[17]

Медицинска Информационна Система[1], Болнична Информационна Система[2], Защитени бази данни[3], Доброволно предоставени географски информационни карти[4], Система за сигурност[5], Предлага модели за одит, контрол и мониторинг (ACMDP) за изграждане на автономни и стабилни системи (ARS), като и мобилни роботизирани системи (MRS)[6], ИС обслужваща Морския транспорт[7]

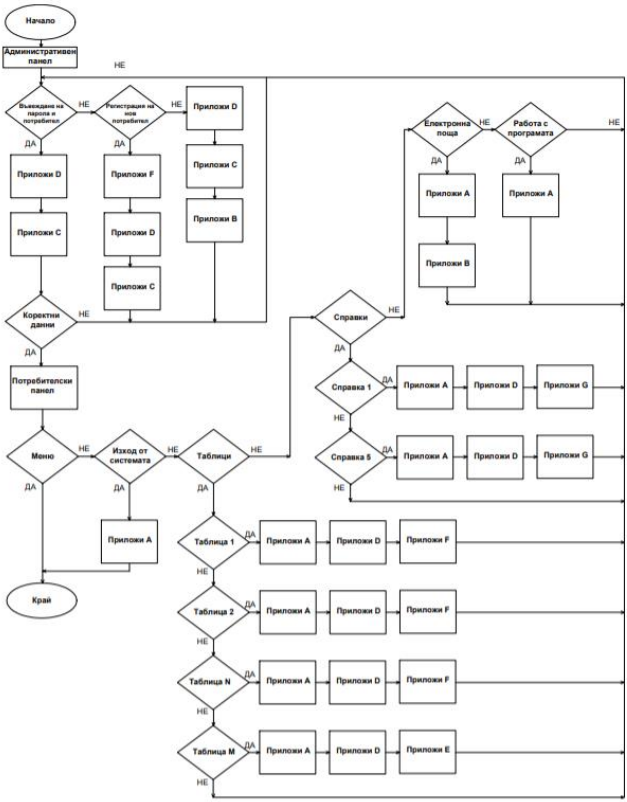
3.2. Избор на комбинация от софтуерни шаблони, за реализиране на ПБКСШП

За реализиране на ПБКСШП са използвани четири софтуерни шаблона от групата на GoF: Команда, Фасада, Сингълтън и Фабричен метод. Подбора им е направен след като подробно са разгледани определени техни характеристики: предназначение на СШ, случаи на употреба и сходство с други СШ. Доказано, е че изборът от софтуерната шаблонна комбинация е подходяща за изпълнение на посочените в т.3.1. функционалности за създаване на ПБКСШП.

3.3. Подход базиран на комбинация от софтуерни шаблони за проектиране (ПБКСШП) – алгоритъм и описание

В конкретния случай за ПБКСШП алгоритмичните указания се свеждат до 3 основни стъпки: вход в системата (стартиране), достъп до административен и потребителски панел. Достъпът до административния панел става непосредствено след стартиране на системата (вход в системата). На потребителя се предоставя три вида избор: Регистрация на нов потребител, Въвеждане на потребителско име и парола за достъп или Генериране на нова парола. При коректно зададена парола и потребител се осигурява достъп до потребителския панел, който предлага списък с меню: Табици, Справки (генериране на справки по зададен период), Изпращане на справки (изпращане на справки по електронна поща до конкретен, предварително зададен потребител), Работа с програмата (описание на функционалността на система) и Изход от системата.

- **Алгоритъм на ПБКСШП**



Фиг.3.1. Алгоритъм на ПБКСШП

- **Описание на алгоритъм ПБКСШП**

1. Вход в системата.

2. Административен панел.

2.1. Ако изборът е Регистрация на потребител, тогава приложи СШ F, D и C

2.1.1. В случай на успех към т. 2.2.

2.1.2. В случай на неуспех към т. 2.1.

2.2. В случай, че изборът е Въвеждане на потребител и парола за достъп, тогава приложи СШ D и C

2.2.1. В случай на успех към т. 3.

2.2.2. В случай на неуспех към т. 2.2.

2.3. В случай, че изборът е Генериране на нова парола, тогава приложи СШ D, B и C

2.3.1. В случай на успех към т. 2.2.

2.3.2. В случай на неуспех към т.2.3.

3. Потребителски панел. Избор на подменю.

3.1. Ако изборът е подменю Таблици, тогава:

3.1.1. Ако изборът е Таблица N, тогава приложи СШ A, F и D

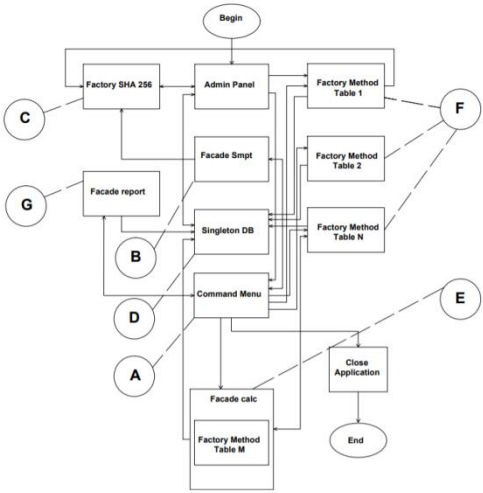
Повтори

3.1.1.1. Добави запис.
3.1.1.2. Изтрий запис.
3.1.1.3. Запази запис.
3.1.1.4. Промени запис.
3.1.1.5. Изход.
Докато N <10
3.1.2. Ако изборът е Таблица M, тогава приложи СШ A, D и E
Повтори
3.1.2.1. Добави запис.
3.1.2.2. Изтрий запис.
3.1.2.3. Запази запис.
3.1.2.4. Промени запис.
3.1.2.5. Изход.
Докато M <2
3.2. В случай, че изборът е подменю Справки, тогава:
3.2.1. Ако изборът е Справки R, тогава приложи СШ A
Повтори
3.2.1.1. Ако е избран период, тогава приложи СШ G, D
3.2.1.1.1. Ако зададения период е коректен се генерира отчет R.
3.2.1.1.2. Ако периода е некоректен към т. 3.2.
Докато R <6
3.3. В случай, че изборът е подменю Електронна поща с отчети, тогава приложи СШ A, B
3.3.1. В случай на успех към т.3.
3.3.2. В случай на неуспех към т. 3.3.
3.4. В случай, че изборът е подменю Работа с програмата, тогава приложи СШ A
3.5. В случай, че изборът е подменю Изход от системата, тогава приложи СШ A
Забележка: N и M– пореден номер на таблица, където (N ϵ [1, 9], M ϵ{1}],), R – пореден номер на справка, където (R ϵ [1, 5])

В реализацията на ПБКСШП са използвани четири шаблона, връзките между тях са представени на фиг.3.2, където инициалите от A до G са както следва:

A. За работа с меню – шаблон Команда;
B. За изпращане на мейли – шаблон Фасада;
C. За криптиране и декриптиране на данни – шаблон Фабричен Метод;;
D. За достъп до РБД – шаблон Сингълтън;
E. За въвеждане и извършване на изчислителни операции – шаблон Фасада;
F. За въвеждане и модифициране на данни в РБД – шаблон Фабричен Метод;
G. За генериране на отчети – шаблон Фасада.

- **Връзки между комбинация от софтуерните шаблони използвани в ПБКСШП**



Фиг.3.2. Комбинация от СШП използвани за реализиране на ПБКСШП

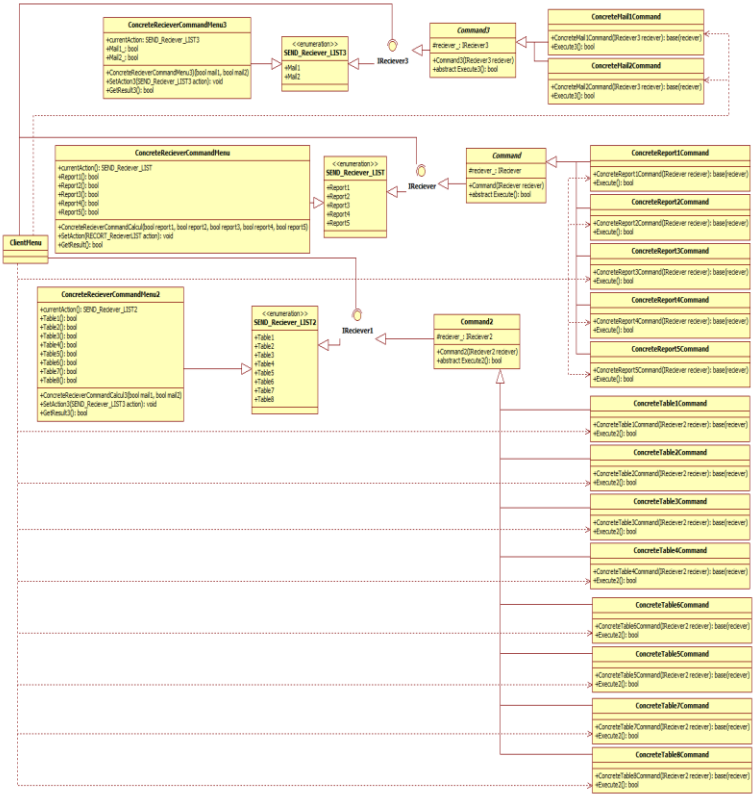
3.4. Подход базиран на комбинация от софтуерни шаблони за проектиране (ПБКСШП) – клас диаграми и описание

За реализиране на ПБКСШП са създадени 15 СШ, базирани на СШ от групата на GoF: Сингълтън, Фасада, Фабричен Метод и Команда за изпълнение на 7 типа функционалности:

- За работа с меню - DPCommandMenu.cs;
- За изпращане на мейли - DPFacadeMail.cs;
- За генериране на отчети – DPFacadeReports.cs;
- За въвеждане и извършване на изчислителни операции - DPFacadeCalcTableM.cs;
- За достъп до РБД - DPSingletonDataBase.cs;
- За криптиране и декриптиране на данни - DPFactoryMethodSHA256.cs;
- За въвеждане и модифициране на данни в РБД - DPFactoryMTable1.cs;
- За въвеждане и модифициране на данни в РБД – DPFactoryMTable2.cs;
- За въвеждане и модифициране на данни в РБД – DPFactoryMTable3.cs;
- За въвеждане и модифициране на данни в РБД – DPFactoryMTable4.cs;
- За въвеждане и модифициране на данни в РБД – DPFactoryMTable5.cs;
- За въвеждане и модифициране на данни в РБД – DPFactoryMTable6.cs;
- За въвеждане и модифициране на данни в РБД – DPFactoryMTable7.cs;
- За въвеждане и модифициране на данни в РБД – DPFactoryMTable8.cs;

- За въвеждане и модифициране на данни в РБД – DPFactoryMTable9.cs

Реализирано е меню с подменюта чрез софтуерен шаблон Команда - **DPCommandMenu** (A) (Фиг. 3.3.):



Фиг.3.3. Клас диаграма на СШ Команда, предназначен за работа с меню (A)

- Подменю Reports: enum RECORT_RecieverLIST, IReciever interface, abstract class Command, ConcreteReport1Command, ConcreteReport2Command, ConcreteReport3Command, ConcreteReport4Command (O), ConcreteReport5Command и ConcreteRecieverCommandCalcul;
- Подменю Tables: enum TABLE_Reciever_LIST2, interface IReciever2, abstract class Command2, ConcreteTable1Command, ConcreteTable2Command,

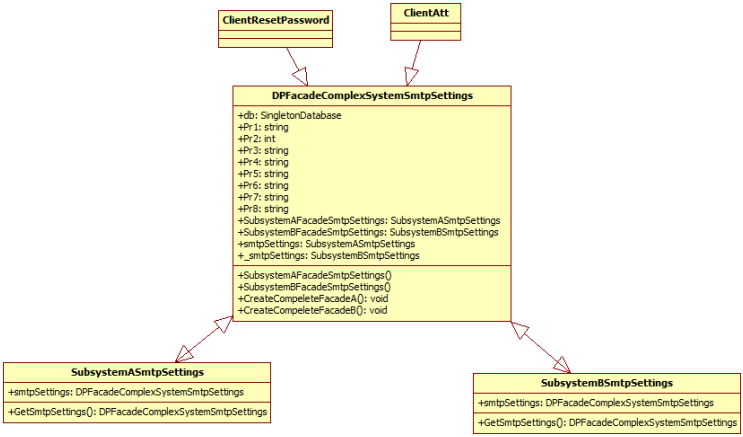
ConcreteTable3Command, ConcreteTable4Command, и т.н. до ConcreteTableMCommand (K) за всяка една от таблиците в ПБКСШП и ConcreteRecieverCommandCalcul2;

- Подменю Send_mails: enum SEND_Reciever_LIST3, IReciever3 interface, abstract class Command3, ConcreteMail1Command, ConcreteMail2Command и ConcreteRecieverCommandCalcul3.

Фасадния шаблон се използва за изпълнение на три типа функционалности:

- За изпращане на мейли;
- За генериране на отчети;
- За въвеждане и извършване на изчислителни операции.

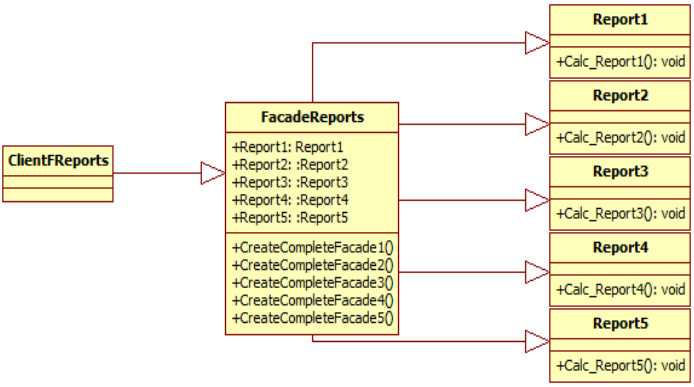
Конкретната реализация на Фасаден шаблон за изпращане на електронни писма - DPFacadeMail е представена на фиг.3.4:



Фиг.3.4. Клас диаграма на СШ Фасада, предназначен за изпращане на мейли (B)

- Class Façade - DPFacadeComplexSystemSmtpSettings;
- Sybsystem - SubsystemASmtpSettingsFacade и SubsystemBSmtpSettingsFacade;
- Client - това е потребителският интерфейс ClientResetPassword и ClientAtt.

Втората реализация на Фасадния СШ в ПБКСШП е за генериране на отчети – DPFacadeRepots, диаграма на Фиг.3.5. представлява връзките между елементите:

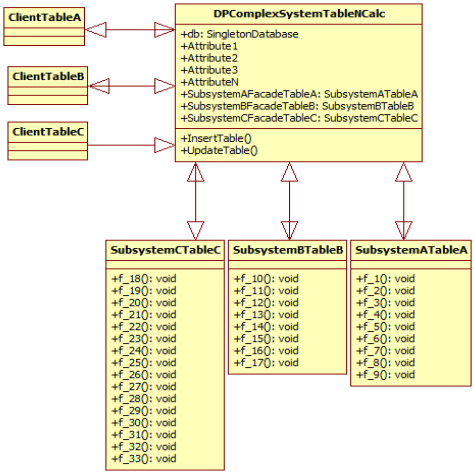


Фиг.3.5. Клас диаграма на СШ Фасада, предназначен за генериране на отчети (G)

- Class Façade - FacadeReports;
- Sybsystems - classes: class Report1, class Report2, class Report3, class Report4 и class Report5 и methods: void Calc_ Report1(), void Calc_ Report2(), void Calc_ Report3(), void Calc_ Report4() и void Calc_ Report5().
- Client - това е потребителският интерфейс: ClientFReports

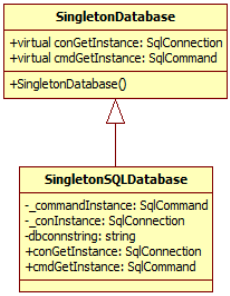
Третата Фасадна реализация е предназначен за въвеждане и извършване на изчислителни операции - DPFacadeCalcTableM (Фиг. 3.6.):

- Class Façade: DPComplexSystemTableNCalc;
- Sybsystem1: class SubsystemATableA, който наследява DPComplexSystemTableMCalc, class SubsystemAFacadeTableA, който наследява SubsystemATableA;
- Sybsystem2: class SubsystemBTableB, който наследява DPComplexSystemTableNCalc, SubsystemBFacadeTableB, който наследява SubsystemBTableB;
- Sybsystem3: class SubsystemCTableC, който наследява DPComplexSystemTableNCalc, SubsystemCFacadeTableC който наследява SubsystemCTableC;
- Client: това е потребителският интерфейс: ClientTableA, ClientTableB и ClientTableC



Фиг.3.6. Клас диаграма на СШ Фасада, предназначен за въвеждане и извършване на изчислителни операции (E)

В ПБКСШП, шаблона Сингълтън се използва за създаване на връзка към база данни (фиг.3.7.):



Фиг.3.7. Клас диаграма на СШ Сингълтън, предназначен за достъп с РБД (D)

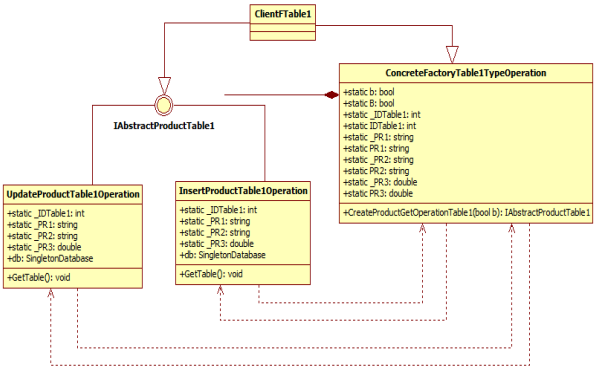
- class SingletonDatabase;
- virtual SqlConnection conGetInstance;
- virtual SqlCommand cmdGetInstance;
- class SingletonSqlServerDatabase, който наследява SingletonDatabase;
- dbconnstring string, която се използва за връзка с РБД.

Фабричен Метод се използва за изпълнение на два вида функционалности:

- За криптиране и декриптиране на данни;
- За въвеждане и промяна на данни в РБД.

Първата реализация на Фабричен Метод за криптиране и декриптиране на данни използва Статичен Фабричен Метод. Поведението на фабричния клас във фабричния метод, както и наследените класове Update и Insert, е подобно на поведението на Медиатора и Колегите в явната реализация на Медиатора. Класът на Фабриката действа като Посредник, а двата подкласа за Актуализиране и Въвеждане са както Колега 1 и Колега 2.

От диаграмата на Фиг.3.8. може да се проследи конкретната шаблонна реализация за Insert и Update в таблица Table1 от ИСМТ:

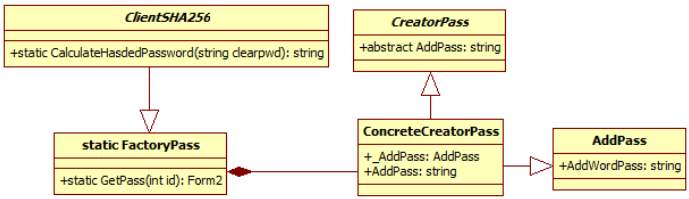


Фиг.3.8. Клас диаграма на СШ Фабричен Метод, предназначен за въвеждане и модифициране на данни в РБД (F)

- IAbstractProductTable1 интерфейс;
- Class ConcreteFactoryTable1TypeOperation;
- Class InsertProductTable1Operation и Class UpdateProductTable1Operation, които наследяват интерфейса на IAbstractProductTable1.

По същия начин шаблоните се прилагат към останалите таблици от базата данни.

Втората реализация на Фабричен Метод в ПБСШП за криптиране и декриптиране на данни (Фиг.3.9.)



Фиг.3.9. Клас диаграма на СШ Фабричен Метод, предназначен за криптиране и декриптиране на данни (С)

- abstract class CreatorPass;
- class ConcreteCreatorPass;
- Static class FactoryPass;
- abstract class ClientSHA256 (Client);
- AddPass (допълнителен низ, който се добавя при формиране на паролата).

Следвайки този подход, могат да се разработят много други ИС.

3.5. Подход за откриване на софтуерни шаблони (ПОСШ) – алгоритъм и описание

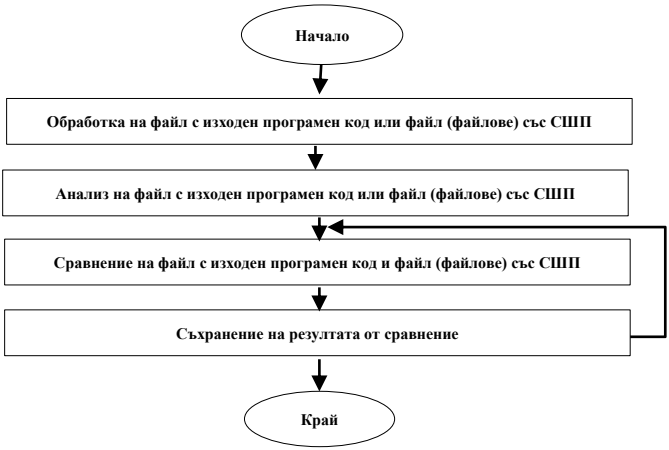
Идентифициране на шаблони за проектиране в програмен код може да подпомогне процесите на поддръжка и еволюция на програмите: това е начин за разбиране на дизайна и изпълнението на програмата; това е начин за информирано реструктуриране, подобряване на структурата и внасяне на промени в кода. Специализираната литература представя не малко подходи за идентифициране, които в повечето случаи са доста сложни и много често - неефективни.

Вторият подход, който се предлага в настоящата глава е практичен подход за откриване на СШ в програмен код. Този подход е използван за учебни цели, в рамките на дисциплината „Софтуерни Шаблони за проектиране“ в Технически университет - Варна. Макар и тестван за идентификация на конкретен шаблон от групата на GoF - Сингълтън, подходът лесно може да се адаптира за идентификация на всеки друг GoF шаблон. Използването на този подход в учебната практика доведе до положителни резултати, както и до повишаване интереса на студентите към процесите, свързани с поддръжка на софтуера и до подобряване ефективността на учебния процес.

Идентифициране на СШ в програмен код е обратна инженерна дейност и може да се използва както за прогнозиране качеството на софтуерната система, така и в процесите на

поддръжка и еволюция: за по-добро разбиране на дизайна и изпълнението на програмата и за извършване на промени в програмния код.

- **Алгоритъм на ПОСШ**



Фиг.3.10. Алгоритъм на ПОСШП (общобщен вариант)

- **Описание на алгоритъма на ПОСШ**

1. Вход в Приложението.
2. Обработка на файл*.
   - 2.1. Избор на директория.
   - 2.2. Избор на проект.
   - 2.3. Селектиране и обединяване на текущите в проекта файлове с разширение .cs.
   - 2.4. Съхраняване на обединения файл под ново разширение .txt.
3. Анализ на файл.
   - 3.1. Зареждане на файл*.
   - 3.2. Форматиране на файла.
   - 3.3. Визуализиране на форматирания файл.
   - 3.4. Обработка на форматирания файл:
     - 3.4.1. Разделяне на думи;
     - 3.4.2. Запис във файл ResultDesignPattern/ ResultOutputCode.
4. Сравняване на файлове.
   - 4.1. Зареждане на файла с изходен програмен код ResultCode и файла или файлове ResultDesignPatternN.
   - 4.2. Сравняване на ResultCode и ResultDesignPatternN.
5. Съхраняване на резултата в OutputCompareResult.

Забележка: * файл с изходен програмен код или със софтуерен шаблон

ИЗВОДИ КЪМ ГЛАВА ТРЕТА

- Направен е сравнителен анализ (Таблица 3.1.) между СШП с функционалността, която се постига в ПБКСШП и същите СШП и функционалност, но в други подходи. Резултатът посочва, че в практиката за постигне на един и същ тип функционалност могат да се приложат различни СШП.
- Направен е сравнителен анализ на съществуващи подходи, базирани на комбинация от шаблони за проектиране, за разработка на софтуер от тип информационна система. Резултатът от него показва, че при сравнение на използваната комбинация от СШП в ПБКСШП със съществуващи в практиката се наблюдава от 0 до 50 % съвпадение (Таблица 3.2.), от което може да се заключи, че няма реализирана подобна комбинация от СШП.
- За реализиране на ПБКСШП са използвани четири софтуерни шаблона от групата на GoF: Команда, Фасада, Сингълтън и Фабричен метод. Подбора им е направен след като подробно са разгледани определени техни характеристики: предназначение на СШ, случаи на употреба и сходство с други СШ. Изборът от софтуерната шаблонна комбинация е подходяща за изпълнение на посочените в т.3.1. функционалности за създаване на ПБКСШП:
  - Шаблонът Команда е подходящ за изграждане на силно преизползваеми компоненти. Той премахва обвързаността между функционалността и нейния инициатор. Компонентите, използващи команди, са изключително податливи на надграждане, тъй като могат да използват всеки един обект, имплементиращ нужния интерфейс. Това е първият избор в комбинацията от СШП за реализиране на ПБКСШП.
  - Използването на шаблона Фасада опростява използването на сложна система. В същото време това намалява броя на извикваните методи. Резултатът е намаляване на броя на зависимостите в приложението. Важно е да се отбележи, че фасадата не изисква да са скрити използваните от системата обекти. Но има едно важно условие: клиентската част, за която е създаден фасадния интерфейс трябва само да го използва и да не осъществява пряк достъп до системата. Освен това системата може да осигурява няколко фасади за решаване на различни задачи. Това позволява създаване на няколко нива на абстракция, които могат да бъдат едно от друго или да са в една и съща равнина. Клиентите могат или да използват конкретна фасада за своята работа или да работят директно с конкретни обекти.

- Съществува връзка между Шаблон Фасада и Шаблон Сингълтън, която може да се определи като двустранна (фиг.2.15. В, според критерии взаимодействие с или с влагане на СШ).
- Шаблонът Сингълтън се използва, ако системата има нужда само от обект в един и същ случай.
- Сингълтън за класове се използва за управляващи ресурси, които са единични по същество – фокусът на селектиране, историята на навигация или дълбочината на даден прозорец. Възможни приложения са мениджър на принтер или мениджър за свързване на база данни. Полезен е, когато трябва да се контролира достъпът до ограничен ресурс.
- Съществува връзка между Шаблон Сингълтън и Фабричен Метод, която може да се определи, като двустранна (фиг.2.15. В, според критерии взаимодействие с или с влагане на СШ).
- Шаблонът Фабричен Метод е подходящ, когато класът не знае предварително какви обекти ще бъдат създадени, възможни варианти за внедряване; (или) класът е програмиран, така, че спецификацията на генерирания обект да се определят само от наследниците; (или) класът разпределя и делегира част от функциите си в спомагателен клас. В същото време е необходимо да се скрие изпълнението му, за постигане на по-голяма гъвкавост или да се разшири функционалността.
- Шаблонът Фабричен Метод се реализира в три форми *класически* - това е специален случай на "Шаблонния метод"; *полиморфен* - представя стратегия за създаване на екземпляр от семейство от типове, позволяващи използването на фабрика в различни контексти; *статичен* - най-простата форма на фабрика. Методът за създаване на статично устройство позволява да се заобиколят ограниченията на конструктора. За конкретните цели се прилага статичен метод.

- Предложен е и е реализиран подход за разработка на софтуер от тип ИС, базиран на предложената комбинация от шаблони за проектиране (ПБКСШП).
- Предложен е и е реализиран подход за търсене на шаблон от тип GoF в програмен код (ПОСШ).

ГЛАВА ЧЕТИРИ: ЕКСПЕРИМЕНТАЛНИ РЕЗУЛТАТИ И ОЦЕНКА НА РАЗРАБОТЕНИТЕ ПОДХОДИ

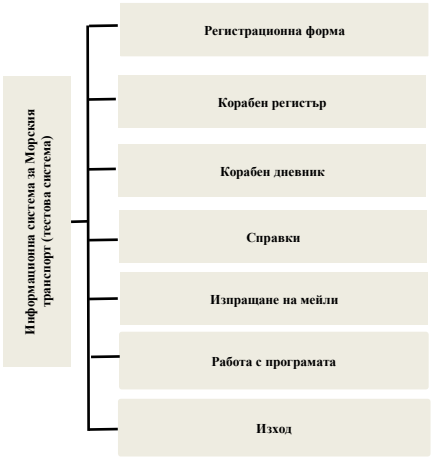
В четвърта глава са представени експериментални резултати от прилага-не на двата подхода. Представени са резултати от използване на първия подход (ПБКСШП) за създаване на примерна информационна система, в случая – ИС за целите на морския транспорт (ИСМТ). Направена е оценка на ПБКСШП, на база на 5 метрични показатели, замерени в две конверсии (случая) - преди и след прилагане на комбинацията от СШП при разработка на примерна информационна система (ИСМТ). По аналогичен начин е направена и оценка на ПОСШ. Резултатите от експерименталните изследвания и оценки на двата подхода са предс-тавени таблично. Направени са изводи относно ефективността на приложените подходи.

4.1. Създаване на ИСМТ, базирана на ПБКСШП

ИСМТ, базирана на ПБКСШП се състои от 7 структурни елемента:

Регистрационна форма, Корабен регистър, Корабен дневник, Справки за период, Изпращане на мейли, Работа с програмата, Изход от програмата (Фиг.4.2.).

4.1.1. Основни структурни елементи на ИСМТ



Фиг.4.2. Основни структурни елементи на ИСМТ

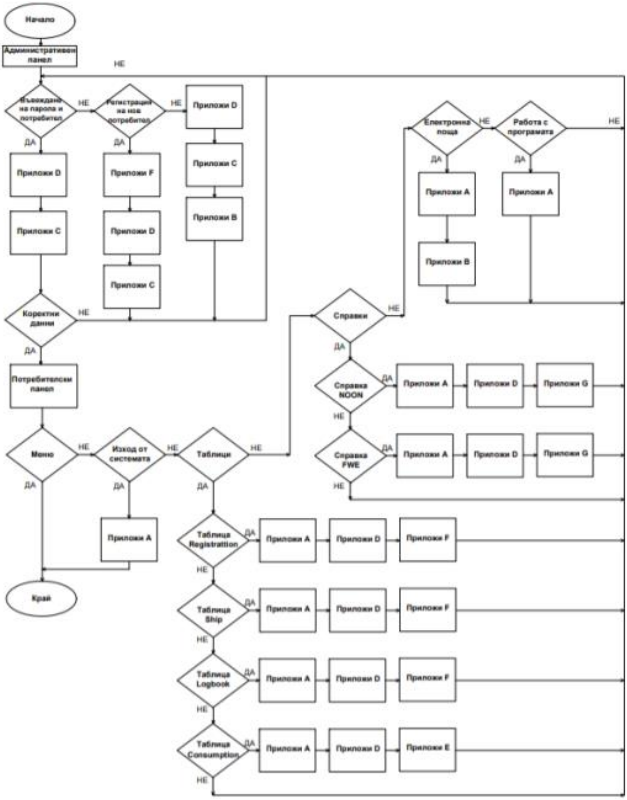
4.3.2. Функционалност на ИСМТ

- Реализираната ИСМТ има следната обща функционалност:

- Администриране;
- Изчисляване консумацията на горива, газьол и масла (бункероване) и производство на питейна вода за нуждите на екипажа;
- Въвеждане и обработка на информацията в РБД;
- Автоматично генериране на отчети;
- Автоматично изпращане на мейли.

4.3.3. ИСМТ – алгоритъм, описание, клас диаграми и реализиране на ИСМТ чрез ПБКСШП

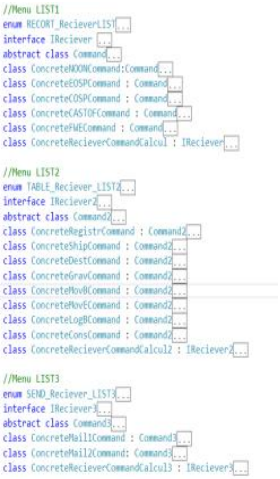
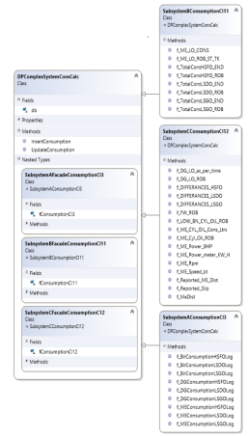
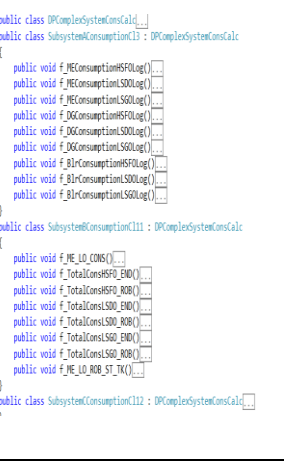
- **Алгоритъм**



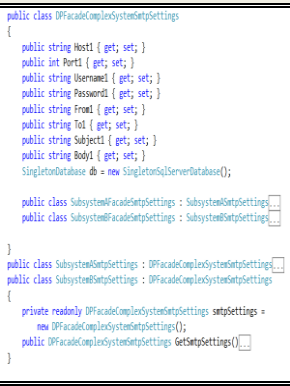
Фиг.4.3. Алгоритъм на ИСМТ (обобщен вариант)

- **Клас диаграми и реализиране на ИСМТ чрез ПБКСШ**

Таблица 4.3. Клас диаграми и реализиране на ИСМТ чрез ПБКССШП

DPCommandMenu.cs	
Клас диаграма	***Реализиране***
	
DPFacadeConsumptionCalc.cs	
Клас диаграма	***Реализиране***
	

DPFacadaMail.cs

Клас диаграма	*Реализиране*
	

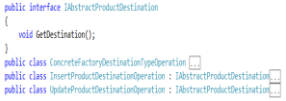
```
public class DPFacadeComplexSystemSmtpSettings
{
    public string Host1 { get; set; }
    public int Port1 { get; set; }
    public string Username1 { get; set; }
    public string Password1 { get; set; }
    public string From1 { get; set; }
    public string To1 { get; set; }
    public string Subject1 { get; set; }
    public string Body1 { get; set; }
    SingletonDatabase db = new SingletonSqlServerDatabase();

    public class SubsystemAFacadeSmtpSettings : SubsystemASmtpSettings...
    public class SubsystemBFacadeSmtpSettings : SubsystemBSmtpSettings...

}
public class SubsystemASmtpSettings : DPFacadeComplexSystemSmtpSettings...
public class SubsystemBSmtpSettings : DPFacadeComplexSystemSmtpSettings
{
    private readonly DPFacadeComplexSystemSmtpSettings smtpSettings =
        new DPFacadeComplexSystemSmtpSettings();
    public DPFacadeComplexSystemSmtpSettings GetSmtpSettings()...
}
```

DPFacadeReports.cs

Клас диаграма	*Реализиране*
	

```
public class NOON
{
    public void Calc_NOON()
    {
        FREPORTS v1 = new FREPORTS();
        if (MenuLogBook.NOON == true)
        {
            v1.Calc_NOON();
        }
        SingletonDatabase db = new SingletonSqlServerDatabase();
    }
}
public class EOSP...

public class COSP...

public class CASTOFF...

public class FWE...

public class FacadeReport...
```

DPFactoryMetodDestination.cs

Клас диаграма	*Реализиране*
	

```
public interface IAbstractProductDestination
{
    void GetDestination();
}
public class ConcreteFactoryDestinationTypeOperation ...
public class InsertProductDestinationOperation : IAbstractProductDestination...
public class UpdateProductDestinationOperation : IAbstractProductDestination...
```

DPFactoryMetodGravity.cs

Клас диаграма	*Реализиране*

	`public interface IAbstractProductGravity`
	`{`
	`  void GetGravity();`
	`}`
	`public class ConcreteFactoryGravityTypeOperation ...`
	`public class InsertProductGravityOperation : IAbstractProductGravity...`
	`public class UpdateProductGravityOperation : IAbstractProductGravity...`

DPFactoryMetodLogbook.cs

Клас диаграма	*Реализиране*

	`public interface IAbstractProductLogbook`
	`{`
	`  void GetLogbook();`
	`}`
	`public class ConcreteFactoryLogbookTypeOperation...`
	`public class InsertProductLogbookOperation : IAbstractProductLogbook...`
	`public class UpdateProductLogbookOperation : IAbstractProductLogbook...`

DPFactoryMetodMovementBegin.cs

Клас диаграма	*Реализиране*

	`public interface IAbstactProductMovBegin`
	`{`
	`  void GetMovBegin();`
	`}`
	`public class ConcreteFactoryMovBeginTypeOperation...`
	`public class InsertProductMovementBeginOperation : IAbstactProductMovBegin...`
	`public class UpdateProductMovementBeginOperation : IAbstactProductMovBegin...`

DPFactoryMetodMovementEnd

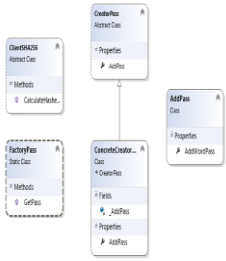
Клас диаграма	*Реализиране*

	`public interface IAbstactProductMovEnd`
	`{`
	`void GetMovEnd();`
	`}`
	`public class ConcreteFactoryMovEndTypeOperation ...`
	`public class InsertProductMovementEndOperation : IAbstactProductMovEnd...`
	`public class UpdateProductMovementEndOperation : IAbstactProductMovEnd...`

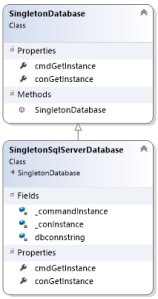
DPFactoryMetodShip.cs

Клас диаграма	*Реализиране*
	`public interface IAbstractProductShip { void GetShip(); } public class ConcreteFactoryShipTypeOperation ... public class InsertProductShipOperation : IAbstractProductShip... public class UpdateProductShipOperation : IAbstractProductShip...`

DPFactoryMetodSHA256.cs

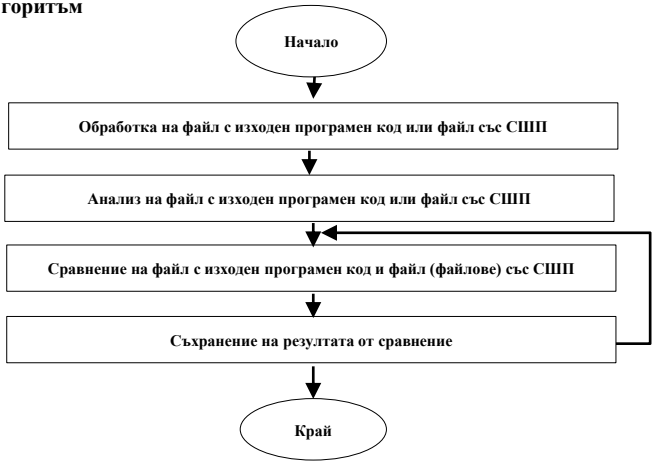
Клас диаграма	*Реализиране*
	`public abstract class CreatorPass { public abstract string AddPass { get;} } public class ConcreteCreatorPass : CreatorPass { AddPass _AddPass = new AddPass(); public override string AddPass { get... } } public static class FactoryPass { public static WindowsFormsApplication2.ConcreteCreatorPass GetPass(int id)... } public abstract class ClientSHA256 { public static string CalculateHashedPassword(string clearpwd)... }`

DPSingletonDatabase.cs

Клас диаграма	*Реализиране*
	`public class SingletonDatabase { public SingletonDatabase() { } public virtual SqlConnection conGetInstance { get; set; } public virtual SqlCommand cmdGetInstance { get; set; } } public class SingletonSqlServerDatabase : SingletonDatabase { private SqlConnection _conInstance = null; private SqlCommand _commandInstance = null; private string dbconnstring = @"Data Source=(localdb)\Projects;Initial Catalog=AD;Integrated Security=True;" + "Connect Timeout=15;Encrypt=False;TrustServerCertificate=False"; public override SqlConnection conGetInstance... public override SqlCommand cmdGetInstance { get... set... } }`

4.4. Приложение за идентифициране на СШП в изходен програмен код - алгоритъм, описание, реализация чрез ПОСШ

- **Алгоритъм**



Фиг.4.4. Алгоритъм на ПИСШ (обобщен вариант)

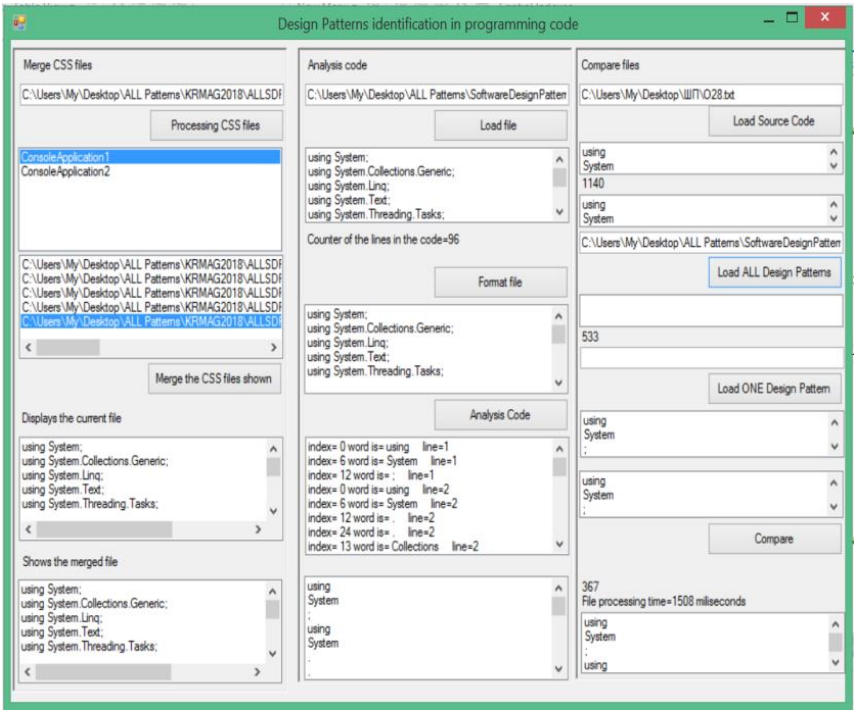
- **Описание на алгоритъма**

1. Вход в Приложението.
2. Обработка на файл.
   - 2.1. Избор на директория.
   - 2.2. Избор на проект.
   - 2.3. Селектиране и обединяване на текущите в проекта файлове с разширение .cs.
   - 2.4. Съхраняване на обединения файл под ново разширение .txt.
3. Анализ на файл.
   - 3.1. Зареждане на файл.
   - 3.2. Форматиране на файла.
   - 3.3. Визуализиране на форматирания файл.
   - 3.4. Обработка на форматирания файл:
     - 3.4.1. Разделяне на думи;
     - 3.4.2. Запис във файл ResultDesignPattern/ ResultOutputCode.
4. Сравняване на файловете.
   - 4.1. Зареждане на файла с изходен програмен код ResultCode и файла/ файлове ResultDesignPatternN.
   - 4.2. Сравняване на ResultCode и ResultDesignPatternN.
5. Запис на резултата в OutputCompareResult.

(Забележка: файлът може да бъде файл с изходен програмен код или код със софтуерен шаблон)

Подхода за откриване на софтуерни шаблони в изходен програмен код (**ПОСШ**) е реализиран като WindowsForms Applcation на езика C# във Visual Studio.NET.

- **Реализация на ПОСШ**



Фиг.4.5. Приложение за идентифициране на СШ в изходен програмен код (основна екранна форма)

- **Резултати**

ПОСШ е приложен в 24 тестови изходни кода, написани на C#, за идентифициране на Шаблона Сингълтън. Резултатите са:

- 54% от всички проведени тестове за идентифициране на шаблона Сингълтън в изходния програмен код са с процент съвпадение по-голям от 80%, което е показател за ефективност.
- Средната стойност за идентифициране на шаблона Сингълтън в изходен програмен код, в направените 24 теста е 82%, което е показател за ефикасност.

Резултатите за ефективност и ефикасност на подхода са оптимистични, което може да се види от Таблица 4.4.

Таблица 4.4. Експериментални резултати от идентифициране на шаблона Сингълтън в изходен програмен код в C#

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
R	641	123	282	297	220	475	113	104	404	126	325	322	234	171	644	215	243	280	258	259	337	107	97	155
S	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44	44
CDS	43	38	43	34	33	43	30	36	44	39	32	34	39	25	34	36	38	33	36	38	38	35	35	35
Time	685	162	326	332	254	519	144	141	449	166	358	357	274	197	679	252	282	314	295	298	376	143	133	191
%Compare	98%	86%	98%	77%	75%	98%	68%	82%	100%	89%	73%	77%	89%	57%	77%	82%	86%	75%	82%	86%	86%	80%	80%	80%

Средна стойност от откриване на шаблон: 82%

Легенда: **R**: изходен програмен код [думи] ; **S**: шаблон Сингълтън [думи];
CDS: кандидати за идентифициране на шаблон Сингълтън в изходен програмен код [думи];
Time: време за изпълнение [милисекунди]
% Compare: идентифициран шаблон Сингълтън в изходен програмен код [%]
DSDP: идентифициран шаблон Сингълтън в изходен програмен код
№ от 1 до 24: C# приложение

4.5.2. Резултати

За оценка на индикаторите за качество на кода при прилагане на ПБКСШП е извършен автоматизиран анализ на 5 софтуерни показатели върху тестовата система (ИСМТ) в 2 конверсии, а именно, приложения: без използване на шаблони и след добавяне на КСШ. Това са индекс на поддръжка, цикломатична сложност, дълбочина на наследяване, отношение на класа и кодови линии. Анализът показва, че след използване на КСШ, индексът за поддръжка се е подобрил значително. Съществуват стандарти като ISO / IEC FCD 25010, които определят списък на атрибутите за качество на софтуера. "Поддържане" е един от тези атрибути. Атрибутите на качеството обаче не се разглеждат изолирано. Подобряването на някои атрибути може да има положително или отрицателно въздействие върху другите. По този начин, възможността за поддържане има положително въздействие върху следните качествени атрибути:

- Reliability (надеждност)
- Testability (проверимост)
- Flexibility (гъвкавост)

Таблица 4.11. Взаимодействие между атрибутите на качеството

	Availability	Efficiency	Flexibility	Integrity	Interoperability	Maintainability	Portability	Reliability	Reusability	Robustness	Testability	Usability
Availability								+		+		
Efficiency			-		-	-	-	-		-	-	-
Flexibility		-		-		+	+	+		+		
Integrity		-			-				-		-	-
Interoperability		-	+	-		-						
Maintainability	+	-	+					+			+	
Portability		-	+		+	-			+		+	-
Reliability	+	-	+			+				+	+	+
Reusability		-	+	-				-			+	
Robustness	+	-						+				+
Testability	+	-	+			+		+				+
Usability		-								+	-	

От друга страна, атрибутът за повторно използване зависи пряко от гъвкавостта и проверимостта, така че този фактор също ще се подобри. Подобряването на възможностите за повторна употреба, гъвкавост и надеждност също ще повлияят положително на преносимостта. Освен това подобряването на възможността за повторно използване, гъвкавост и възможност за тестване се отразява положително на надеждността и т.н. Следователно трябва да се заключи, че прилагането на КСШ за този клас ИС ще доведе до подобряване на редица качествени показатели и следователно до цялостно подобряване на качеството на софтуера.

Резултатите от сравнителния анализ за двата случая са представени в Таблица 4.12:

Таблица 4.12. Резултат от оценка на ИСМТ използващ ПБКСШП

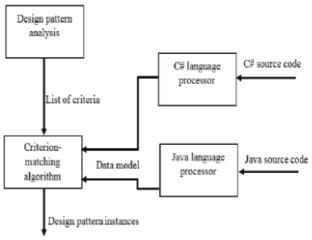
СШ	Код без СШ	Код със КСШ	индекс на поддържаемост		цикломатична сложност		дълбочина на наследяване		свързване на класа		линии на кода	
			без СШ	със СШ	без СШ	със СШ	без СШ	със СШ	без СШ	със СШ	без СШ	със СШ
1) DPFMD	Update() : void	UpdateDestination	61	81	1	**10**	-	1	9	11	11	25
	Insert() : void	InsertDestination	67	89	1	**10**	-	1	5	8	7	19
2) DPFML	Update() : void	InsertLoogbook	55	89	1	**62**	-	1	11	9	14	83
	Insert() : void	UpdateLoogbook	62	89	1	**62**	-	1	7	12	7	88
3) DPFMMB	Update() : void	UpdateMovementBegin	61	81	1	**10**	-	1	9	11	11	25
	Insert() : void	InsertMovementBegin	67	89	1	**10**	-	1	5	8	7	19
4) DPFMME	Update() : void	UpdateMovementEnd	61	81	1	**10**	-	1	9	11	11	25
	Insert() : void	InsertMovementEnd	69	89	1	**10**	-	1	5	8	6	19
5) DPFMS	Update() : void	UpdateShip	60	88	1	**18**	-	1	9	11	11	33
	Insert() : void	InsertShip	65	89	1	**18**	-	1	5	8	7	28
6) DPFMSH*	CalculateHashedPassword(string) : string	ClientFactory SHA256	72	81	2	**3**	-	1	4	5	4	5
7) DPCM*	shipToolStripMenuItem_Click(object, EventArgs) : void	ShipCommand	85	85	1	**2**	-	2	3	2	2	4
	gravityToolStripMenuItem_Click(object, EventArgs) : void	GravCommand	85	85	1	**2**	-	2	3	2	2	4
	destinationToolStripMenuItem_Click(object, EventArgs) : void	DestCommand	85	85	1	**2**	-	2	3	2	2	4
	consumptionToolStripMenuItem_Click(object, EventArgs) : void	ConsCommand	85	85	1	**2**	-	2	3	2	2	4
8) DPFCC*	f_TotalConsHSFO_END() : void	f_TotalConsHSFO_END() : void	70	90	1	**1**	-	-	1	1	5	1
	f_TotalConsHSFO_ROB() : void	f_TotalConsHSFO_ROB() : void	70	90	1	**1**	-	-	1	1	5	1
	f_TotalConsLSDO_END() : void	f_TotalConsLSDO_END() : void	70	90	1	**1**	-	-	1	1	5	1
	f_TotalConsLSDO_ROB() : void	f_TotalConsLSDO_ROB() : void	70	90	1	**1**	-	-	1	1	5	1
	f_TotalConsLSGO_END() : void	f_TotalConsLSGO_END() : void	70	90	1	**1**	-	-	1	1	5	1
	f_TotalConsLSGO_ROB() : void	f_TotalConsLSGO_ROB() : void	70	90	1	**1**	-	-	1	1	5	1
9) DPFM*	smtp() : void	SmtpSettings.SmtpSettingsFacade	58	53	1	**7**	-	2	5	22	13	54
10) DPSDB*	sqlcon_() : void	AbstractDatabase	92	93	1	**5**	-	1	1	2	1	5

4.1. Оценка на ПОСШП. Резултати

Както вече бе споменато качеството на софтуерният продукт може да се оцени чрез изчисляване на софтуерните метрични показатели, на базата на които всеки един продукт се счита за „добър“ или „лош“. Обикновено в края на проекта или в края на даден цикъл в методологията за управление на проекти се прави изчисляване на тези показатели, ако показателят не е достатъчно добър, изходният код може да бъде изменен. Този подход е известен като ретроактивен подход (retroactive approach), той има по-висока цена от всяко проактивно решение (колкото по-рано се открие грешка по време на разработката, толкова по-евтино е да се отстрани дефектът). Софтуерните шаблони за проектиране влияят върху

показателите за качество на софтуера. Така още по време на разработка има възможност да се оценят качествените показатели, което е и същността на проактивния подход.

Таблица 4.13. Съпоставка между ПОСШП и [137]

Подход	ПОСШП	[137]
СШП	11 (1)	5, 6, 7, 11, 17, 19
Схема		

Забележка:
СШ: Всички СШ[1], Абстрактна Фабрика[2], Адаптер[3], Мост[4], Композиция[5], Декоратор[6], Фактори Метод[7], Наблюдател[8], Прототип[9], Прокси[10], Сингълтън[11], Състояние[12], Шаблонен Метод[13], Посетител14, Стратегия[15], Команда[16], Медиатор[17], Строител[18], Верига отговорности[19], Фасада[20]

За постигане на възможно най-голяма гъвкавост [137] се създава рамка за съвпадение на критерии, при което СШП се описват чрез този набор от критерии, към който могат да бъдат съпоставени кодовите елементи.

- **Резултати**

Таблица 3.14. Резултат от оценка на Приложение за идентифициране на GoF шаблона Сингълтън, базирано на ПОСШП

Подход	индекс на поддържаемост	цикломатич на сложност	дълбочина на наследяване	свързване на класа	линии на кода
ПОСШП	**61**	88	7	61	704

ИЗВОДИ КЪМ ГЛАВА ЧЕТИРИ

- Направен е автоматизиран анализ на базата на 5 софтуерни показа-тели на ПБКСШП за тестовата система в 2 конверсии, а именно, при-ложения без използване на шаблони и след добавяне на комбинация от софтуерни шаблони. Изследваните показатели са: индекс на подд-ръжка, цикломатична сложност, дълбочина на наследяване, отно-шение на класа и кодови линии. Проучванията показват, че след из-ползване на комбинация от софтуерни шаблони, индексът за подд-ръжка се е подобрил значително. От друга страна, атрибутът за пов-торно използване зависи пряко от гъвкавостта и проверимостта, така че този фактор също ще се подобри. Подобряването на въз-можностите за повторна употреба, гъвкавост и надеждност също ще повлияят положително на преносимостта. Освен това подобряването на

възможността за повторно използване, гъвкавост и възможност за тестване положително се отразява на надеждността и т.н. Следова-телно трябва да се заключи, че прилагането на комбинация от соф-туерни шаблони за този клас информационни системи ще доведе до подобряване на редица качествени показатели и следователно до ця-лостно подобряване на качеството на софтуера.

- ПОСШП е оценен на базата на използвани 5 софтуерни метрики за оценка на кода: индекс на поддържаемост, цикломатична сложност, дълбочина на наследяване, свързване на класа и линии на кода. Индекса на поддържаемост е 61%, което е в границите на номинал-ната стойност за този софтуерен показател.
- Направените изследвания показват, че прилагането на софтуерни шаб-лони :
    - подобряват основни качествени характеристики на кода: повишават експлоатационната характеристика на софтуера, а имен-но индекса на поддържаемостта; намалява времето за разработка на софтуера, поради възможността за преизползване на кода.
    - влошава някои характеристики на софтуера, като: цикломатична сложност, дълбочина на наследяване, свързване на класа и линии на кода, но те не касаят експлоатационните качества на софтуера.

ПРИНОСИ НА ДИСЕРТАЦИОННИЯ ТРУД

1. Научно – приложни приноси

1.1. Предложена е обобщена класификация на шаблоните за проектиране, базирана на различни техни признаци.

1.2. Предложен е подход за разработка на клас софтуерни системи, базиран на комбинация от софтуерни шаблони за проектиране (ПБКСШП).

1.3. Предложен е подход за откриване на шаблон за проектиране от групата на GoF (Сингълтън) в изходен програмен код.

2. Приложни приноси

2.1. Разработена е информационна система за целите на морския транспорт с използване на ПБКСШП.

2.2. Разработено е софтуерно приложение за идентифициране на софтуерни шаблони за проектиране от групата на GoF (Сингълтън и др.) в изходен програмен код.

ПУБЛИКАЦИИ ПО ТЕМАТА НА ДИСЕРТАЦИОННИЯ ТРУД

1. **Nikolaeva**, D. „Design patterns - approach to automation of programming work.”, УДК 004.4'2. Електронно издание. Секция 3. Информационные технологии в управлении, автоматизации и мехатронике, Международной научно-технической конференции молодых ученых, аспирантов и студентов (Севастополь, 13-15 мая 2015 г.), pp.246-250, https://elibrary.ru/item.asp?id=24828215

2. **Nikolaeva**, D.Zh. „Place of Design Patterns in education through the Eyes of a phd.”,XI Международно научно-техническа конференция „Стратегия на качеството в индустрията и образованието”, Юни 01-05, 2015 г., ТУ-Варна (Украйна, Русия, България), том 1, Направление „Качество в образованието”, pp.470-475, ISBN 978-966-2637-35-9, ISBN 978-966-2637-34-2 (T.1)

3. **Nikolaeva**, Dimitrichka Zheleva, and Violeta Todorova Bozhikova. "Approaches to Increasing the Quality and Reliability in the Field of Design Patterns.", XXIV Международна научна конференция „Електронна техника – ЕТ 2015” 15-17 септември 2015 г. Созопол, pp. 278-281, ISSN 1314-0078

4. **Николаева**, Димитричка.; Божикова, Виолета. Подход за изграждане на клас софтуерни системи чрез комбинация от софтуерни шаблони. Пета научна конференция с международно участие "Компютърни науки и технологии" 28 - 29 септември 2018 г., Варна, България, том 2, стр.119-126, ISSN 1312-3335

5. **NIKOLAEVA**, Dimitrichka; BOZHIKOVA, Violeta. One Approach to Improve the Software Quality by Applying Software Design Patterns. In: 2019 16th Conference on Electrical Machines, Drives and Power Systems (ELMA). IEEE, 2019. p.1-6, (Scopus)

6. **NIKOLAEVA**, Dimitrichka Zheleva; BOZHIKOVA, Violeta Todorova; STOEVA Mariana. A simple approach to design patterns identification in programming code. In: 2019 IEEE XXVII International Scientific Conference Electronics-ET. IEEE, 2019. September 13 - 15, 2019, Sozopol – под печат, (Scopus)

Научно-изследователска работа по други договорни теми и задачи:

- **ПД13/2016г.:** „Подход за разработка на софтуер, базиран на софтуерни шаблони за проектиране”, проект в помощ на докторанти

Научноизследователски проекти:

- **НП8:** „Изследвания в областта на архитектурното, инфраструктурното, алгоритмичното и методологично осигуряване на Cloud базирана среда за провеждане на учебния процес”, 2015г. с ръководител доц.д-р Н. Рускова
- **Националната научна програма**: “Информационни и комуникационни технологии за единен цифров пазар в областта на науката, образованието и сигурността

(ИКТвНОС)" (споразумение за безвъзмездна помощ DO1-205 / 23.11.18 г.), финансирана от Министерството на образованието и науката.

Специални благодарности на:

доц. д-р инж. Виолета Божикова, доц. д-р инж. Христо Вълчанов, доц. д-р инж. Венета Алексиева, доц. д-р инж. Мариана Стоева, доц. д-р инж. Недялко Николов и на всички, които са били съпричастни към моята дисертационна работа.

Annotation

The main goal of the doctoral dissertation is to propose software development approaches through software design templates. Different methods, approaches and techniques are analyzed based on software templates and pattern combinations, as well as such identifying software templates in program code. Their advantages and disadvantages have been identified. As a result of the analysis, a generalized classification of design templates is proposed.

A software development approach has been developed based on a combination of design software patterns (ABCDPs). An approach has been developed for discovering software patterns for source code design. (AFSDPs)

An information system for the purposes of maritime transport and an application for identifying a software template in source code (Singleton) have been created.

An assessment is made of the approaches developed. The results confirm the effectiveness and efficiency of the approaches.