**Технически Университет**

**Варна**

**Факултет: ФИТА**

**Катедра: СИТ**

Дисциплина:

Обектно-ориентирано програмиране

Тема на проекта:

XML Parser

Изготвил:

Теодор Асенов Иванов

ФН:21621650

Глава 1. Увод

* 1. Описание и идея на проектът

Идеята на проектът е създаване на програма*,* която може да комуникира с потребителя, чрез Command Line Interface. Програмта позволява на потребителя четене и различни операции върху XML файлове. Характеристиките на XML елементите се ограничават до:

* идентификатор на елемента
* списък от атрибути и техните стойности
* списък от вложени елементи или текст

Поради това може да кажем, че нашата програма не работи с изцяло валидни XML файлове, а по-скоро със структурирани текстови файлове. Програмата поддържа уникални идентификатори на всички елементи.

Предоставени са следните операции на потребителя за работа с файлове:

* Open – Зарежда съдържанието на файла от подадената директория, а ако такъв не съществува, се създава нов с конкретното име и празно съдържание.
* Close - Затваря отвореният файл. Затварянето изчиства информацията от текущия файл и не са възможни операции, докато не се отвори друг файл.
* Save – Записва новото съдържание от текущо отворения файл.
* Save as – Записва новото съдържание от текущо отворения файл, като дава възможност на потребителя да посочи пътя (директорията) на файла.
* Help – Извежда кратка информация за поддържаните от програмата команди.
* Exit – Излиза от програмата
* Print – Извежда на екрана съдържанието на отвореният файл. Печатането е XML коректно и е форматирано визуално по подходящ начин.
* Select – Извежда стойност на атрибут по подаден идентификатор и ключ от потребителя.
* Set – Писвояване на стойност на атрибут по подаден идентификатор, ключ и стойност от потребителя.
* Children – Извежда информация за вложените елементи на елемент.
* Child – Извежда се информация за n-тия по ред вложен елемент.
* Text – Достъп до текстът на елемент.
* Delete – Изтриване на атрибут на елемент по ключ
* Newchild – Добавяне на нов наследник на елемент
* XPath – операции за изпълняване на прости XPath 2.0 заявки към даден елемент. Резултатът от заявката представлява списък от елементи.
  1. Цел и задачи на разработката

Целта на разработката е да създадем програма, с която потребителят може да комуникира лесно и ефиксано. Задачите по разработката са:

* Да проектираме програмата
* Да реализираме програмата
* Да тестваме програмата

Глава 2. Преглед на предметната област

* 1. Основни концепции и дефиниции, които ще бъдат използвани

В нашият проект ще приложим основните принципи на Обектно-ориентираното програмиране, както и SOLID принципите. Те ще ни помогнат да създадем функционална програма, която ще има разширяемост за в бъдеще.

* 1. Дефиниране на проблеми и сложност на поставената задача

Проблеми:

* Невалидни данни – Потребителят може да предостави невалидни входни данни или да се опита да достъпи невалиден файл. Това може да доведе до грешки по време на изпълнение.
* Съставяне на XML логика в програмата – За реализация на проекта не е позволено използването на готови библиотеки за работа с XML

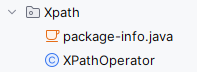
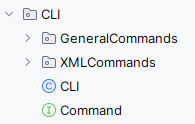
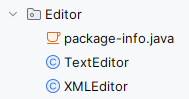
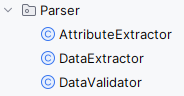
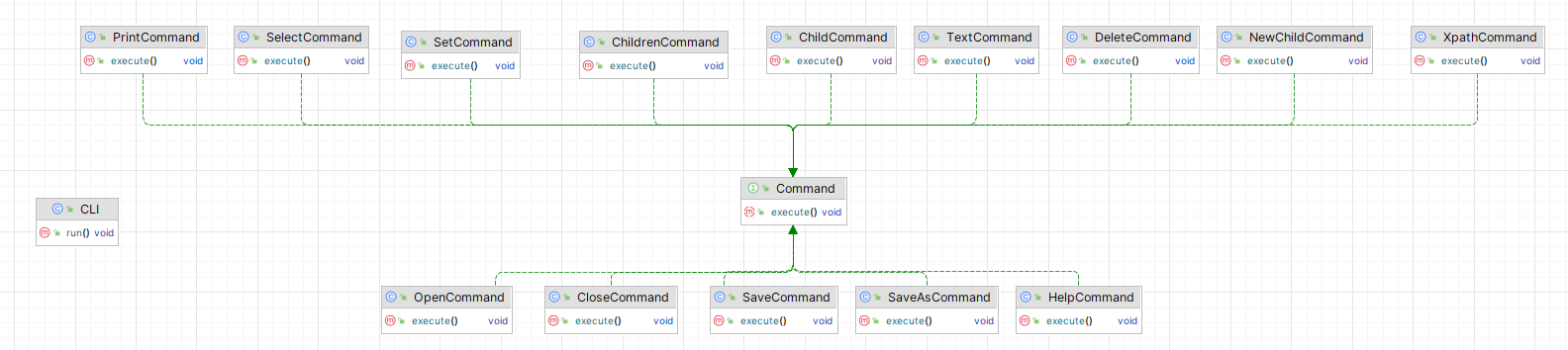
Сложността на задачата може да се определи като средна, понеже е необходимо да се измисли уникален начин за работа с XML файлове.

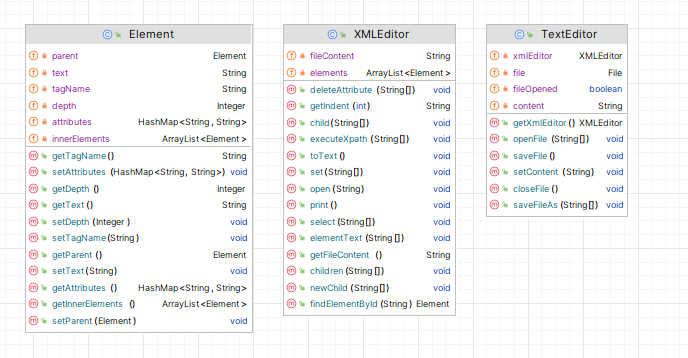
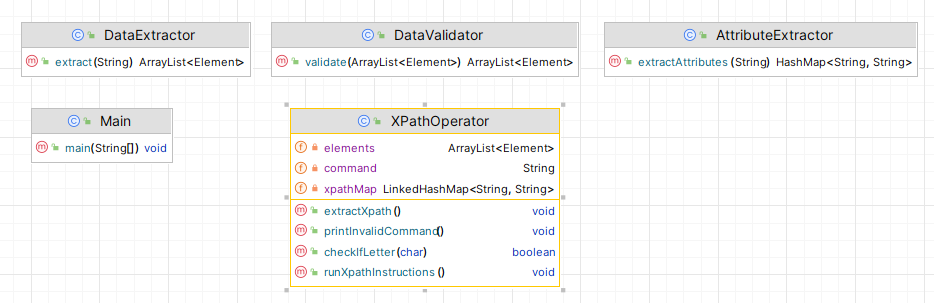
* 1. Подходи, методи за решаване на поставените проблеми

Ще се реализира клас, който ще бъде отговорен за връзката с потребителя. Той ще следи валидността на подадените от потребителя команди и ще обработва невалидни входни данни. Ще се добави и валидация при отварянето на файлове. Трябва отвореният от потребителя файл да спазва нашите правила за XML файлове , за да не възникнат грешки по време на изпълнение.

Понеже за рализацията на програмата не е позволено използването на готови библиотеки за работа с XML, ще създадем класове, които ще реализират XML логиката. Текстът от входния файл ще се превръща в XML структура, с която ще работи програмата. Трябва да се реализира и обратната логика, за да може XML структурата да бъде превърната в текст при необходимост.

Глава 3. Проектиране

* 1. Обща структура на проекта, пакети, които ще се рализират
* Пакет : CLI – Съдържа отделните пакети, които съдържат различен тип команди
  + Пакет: General Commands – Основните команди на програмата
  + Пакет: XML Commands- XML-специфични команди
* Пакет : Editor – Съдържа редакторите за текстов файл и XML
* Пакет : Parser – Съдържа класовете, които от текстът на входния файл създават XML структурата и я валидират
* Пакет : Element – Съдържа класът, представляващ елемент в нашата XML структура файлове
  1. Диаграми/блок схеми



Глава 4. Реализация, тестване

* 1. Реализация на класовете
     1. Изпълнение на команди, подадени от потребителя:
* Ще се създаде клас CLI, който ще има статичен метод, за изпълнение от главната функция. Той ще изпълнява конкретните команди.
* Всяка команда представлява отделен клас. Съответно при изпълнението на всяка команда се създава инстанция от нейния тип. Тази инстанция изпълнява командата, като нужната логика за изпълнение ще бъде капсулирана в класовете TextEditor(клас за работа с файлове) и XMLEditor(клас за изпълнение на XML команди).
* При евентуални невалидни входни данни, ще се предотвратят грешки по време на изпълнение и потребителя ще бъде насочен към това да подава правилни входни данни.
  + 1. Пряка операция с текстови файлове:
* Ще се създаде класа за работа с текстови файлове TextEditor. Той ще съдържа логиката в това да отвори файла, да извлече неговото съдържание и да го затвори. Също, ще може да запаметява файла. Класът ще има поле във себе си което ще бъде от тип XMLEditor, който ще съдържа XML логиката.
  + 1. Реализация на XML логиката:
* Ще се създаде класа, който ще съдържа главната XML логика и ще извършва различни операции по XML структурата. Той съдържа поле с текстовото съдържание на текущия файл и поле, представляващо списък със всички елементи.

За да се създаде XML структурата ще:

* Се създаде клас Element – той ще представлява XML елемент. Полетата му ще представляват свойствата на един елемент: Име, атрибути, текст.

Също всеки елемент ще бъде свързан с елементът представляващ неговия родител, както и неговите вложени елементи (деца). Чрез този подход ще съставим една Дървовидна структура, която ще може да обхождаме при операции.

* Се създаде клас DataExtractor - Той ще се използва от класа XMLEditor за да превърне текстът от входния файл в структура от свързани елементи.

За олесняване ще се създадат също класовете:

* + AttributeExtractor – ще извлича атрибутите на всеки елемент
  + DataValidator – ще валидира създадената структура, за да се уверим, че подаденият XML файл спазва зададените правила.
* Също, за улеснение ще се създаде клас XPathOperator. Той ще съдържа логиката за XPath заявки. Ще ги извлича от подадената команда и ще ги изпълнява. Класът ще се използва от XMLEditor, който съдържа нашата структура и извършва отделните XML операции.