Federal State Budgetary Educational Institution

higher education

Ufa University of Science and Technology

Department of Computational Mathematics and Cybernetics

Laboratory work No.3

“Methodology of object-oriented modeling. Stage of creating a physical model”

Made by: Kozlov I.A.

Group number: PRO 234

Checked by: Rizvanov D.A.

Ufa 2023

**Purpose of the work:** introduction to the basic elements of defining, presenting, designing, and modeling software systems using the UML language.

**Task:**

1. Review the material on object-oriented modeling (Appendix 1 (lab. rab. 2 and lecture materials))

2. Decide on diagrams from the family UML models at the stage of creating a physical model of an automated system (the topic was selected during laboratory work 1).

3. Develop UML diagrams of the physical model creation stage.

4. Document use cases.

The primary description should contain the following sections.

a. Short description.

b. Participating entities.

c. Prerequisites required for initiating a use case.

d. A detailed description of the event stream, which includes: the mainstream, which can be broken down to show subordinate event streams (subordinate streams can be further divided into even smaller streams, in order to make the document readability more convenient); alternative streams for identifying exceptional situations.

e. Postconditions that determine the state of the system, after which the use case is completed (Example in Appendix 1).

5. Write at least 10 requirements according to the requirements syntax:

[circumstances] [subject] [action] [object] [restriction]

Example: When a signal is received [circumstances], the system [subject] must set the [action] signal discharge [object] for two seconds [limit].

6. Formulate non-functional requirements.

7. Develop a data processing algorithm (see the example of code design with explanations in the archive pikpo3\_python.zip or pikpo3\_java.zip).

a. The processing algorithm should be implemented using an object-oriented approach. The data handler is implemented in a separate class (DataProcessor), which has 3 basic methods: read the data source (read), starting data processing (run), displaying the result on the screen (print\_result).

b. Implement the necessary processing methods in the class (for example, cleaning, assigning categories, etc.).

c. Use the "Factory" pattern to call different instances DataProcessor depending on the type of input data set (for example, csv-file, txt-file).

d. Upload the code to GitHub (see " Memo for working with GIT»).

e. Develop UML class diagram the data handler.

8. Write a report. The report should include a set of static and dynamic models, a description of the data processing algorithm, screenshots with the results of the algorithm execution (a fragment of the input data set and output after processing), and a screenshot GitHub- a repository with uploaded project files.

**Description of the work:**

1) UML provides support for all stages of the IS lifecycle and provides several graphical tools for these purposes – diagrams.

At the stage of creating a conceptual model, business case models and activity diagrams are used to describe business activities, business object models and sequence diagrams are used to describe business objects.

At the stage of creating a logical IS model, the description of system requirements is set in the form of a model and a description of system precedents, and preliminary design is carried out using class diagrams, sequence diagrams and state diagrams.

At the stage of creating a physical model, detailed design is performed using class diagrams, **component diagrams**, deployment diagrams. We will use the second one for creating physical model of the IS.

Изображение выглядит как текст, диаграмма, снимок экрана, линия

Автоматически созданное описание

2) Below there is documentation of use cases table.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Use case | Short description | Subject | Prerequisites | Main stream | Postconditions |
| Find the information about weather on the selected day | The main functionality of the system. Users want to get information about weather | Clients and website/application | Open the website or application | Users open the website or application. There are several menu and users can choose “get information about current day/3 days/week/month”. After that user click on wanted button and see the result | Users get the information |
| Receive notifications when the weather changes | Weather is subject to change that’s why users should have functionality to get new weather after changing | Clients and website/application | Agree with sending notifications | When weather is changed, users get notification about that. In this notification there is data about correct weather | Users get notifications about changing the weather |
| To get technical support | Users should be able to get support | Clients and website/application | Fulfill the form about problems | When user has error or other problems, he clicks on button “help me” and form of support is opened. User fulfills these forms and after some amount of time he will be supported | Users get support |
| Create user account | Creation is needed for changing data thanks to user’s observations | Clients and website/application | Put on the button | When user click on “create account”, he get the form for registration | Users have account |
| Use API | For making applications based on working website | Devs and website | To request rule for using API | Menu of website or application has field “for developers”. After clicking on this button, devs will be received the form after fulfilling of which system administrator considers giving access to API for these developers | Devs can use API |
| Get report of using the system | For analytic research | System administrator and server | Get request from owner | Owner requests report from system administrator. Administrator gets information about using the website/application thanks to server and send to owner | Owner has statistical data of using website/app |
| Give special rules for users | Some users can be very important for system’s working that’s why they need special rules | System administrator and server | Users must be with good reputation | Users who have good reputation ask system administrator to be given special rules | Users get special rules |

*Documentation of use cases*

3) Also, system has several requirements:

1. when user open the website, the website must load itself less than for 4 seconds (speed of the Internet is ignored);

2. when user open the application, the application must load itself less than for 10 seconds (speed of the Internet is ignored);

3. when user open the information about weather, the website must give correct information about weather which not only consist of temperature;

4. when devs try to use API, they must be notified that there is prohibition in using API, for instance, it must be non-commercial using.

5. when system administrator requires server to give statistic data, server must send that for 10 minutes;

6. when website or desktop application try to request server to send information, server and website/application must have SSL-connection or another tool of security;

7. when the website can not work properly, system administrator must reload the server for 2 hour;

8. when user wants to get technical support, specialists must answer not more than for 1 day;

9. when users fulfill application form for special rules, system administrator must answer not more than 1 week;

10. when 10 users with special rules submit the same requests for data modification, server must update information for 5 minutes;

4) Non—functional requirements are requirements that define the properties that the system must demonstrate, or restrictions that it must comply with that are not related to the behavior of the system. For our system there are following non-functional requirements:

1. Efficiency should be an important part of the system.

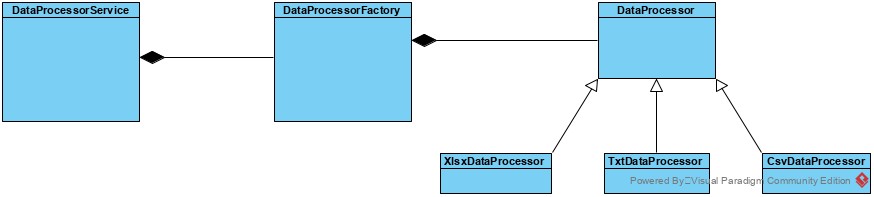
2. The website and application should be user-friendly.

3. The system must be extensible.

4. The system must be reliable.

5) Data processing algorithm is in the github, but idea of working program is following: we use the pattern “Factory method” in order to our program can work with any files. Firstly, we call method run\_service() of DataProcessService class. After that we use method get\_processor() of Factory class in order to get current processor (one of three available). Finally, we use methods run() and print\_result().

Below there is diagram which shows how our classes are organized.



**Conclusion:** I have learned the basic elements of defining, presenting, designing, and modeling software systems using the UML language.

Github: