**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Руководитель департамента  программной инженерии факультета  компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Р. А. Родригес Залепинос  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Программа для обнаружения облаков на спутниковых снимках**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.04.01-01 81 01-1-ЛУ** | | |
|  | | |
|  |  | |
| Исполнитель  студент группы БПИ182  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /М.А. Шакура /  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г. | |
|  | | |
|  | |  |

**Москва 2020**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.04.01-01 81 01-1-ЛУ |  | |  | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Программа для обнаружения облаков на спутниковых снимках**  **Пояснительная записка**  **RU.17701729.04.01-01 81 01-1**  **Листов 17** | | | | |
|  | |  | | |
|  | | |
|  | | | | |
|  | | | |  |

**Москва 2020**

**АННОТАЦИЯ**

В данном программном документе приведена пояснительная записка к программедля обнаружения облаков на спутниковых снимках.

В разделе «Введение» указано наименование программы, краткое наименование программы и документы, на основании которых ведется разработка.

В разделе «Назначение и область применения» указано функциональное назначение программы, эксплуатационное назначение программы и краткая характеристика области применения программы.

В разделе «Технические характеристики» содержатся следующие подразделы:

− постановка задачи на разработку программы;

− описание алгоритма и функционирования программы;

− описание метода организации входных и выходных данных;

− описание и обоснование выбора состава технических и программных средств.

В разделе «Ожидаемые технико-экономические показатели» указана предполагаемая потребность и экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами.

Настоящий документ разработан в соответствии с требованиями:

1) ГОСТ 19.101-77 Виды программ и программных документов;

2) ГОСТ 19.102-77 Стадии разработки;

3) ГОСТ 19.103-77 Обозначения программ и программных документов;

4) ГОСТ 19.104-78 Основные надписи;

5) ГОСТ 19.105-78 Общие требования к программным документам;

6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом;

7) ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению.

Изменения к Пояснительной записке оформляются согласно ГОСТ 19.603-78, ГОСТ 19.604-78.

Перед прочтением данного документа рекомендуется ознакомиться с терминологией, приведенной в Приложении 1 настоящей пояснительной записки.

ОГЛАВЛЕНИЕ

[1. ВВЕДЕНИЕ 4](#_Toc40356299)

[1.1. Наименование программы 4](#_Toc40356300)

[1.2. Документы, на основании которых ведется разработка 4](#_Toc40356301)

[2. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 5](#_Toc40356302)

[2.1. Постановка задачи на разработку программы 5](#_Toc40356303)

[2.2. Описание алгоритма и функционирования программы 5](#_Toc40356304)

[2.2.1. Описания алгоритма работы взаимодействия сервера и клиента 5](#_Toc40356305)

[2.2.2. Описания алгоритма детектирования облаков 5](#_Toc40356306)

[2.2.3. Описание работы клиентского приложения 8](#_Toc40356307)

[2.2.4. Описание алгоритма работы сервера 10](#_Toc40356308)

[2.3. Описание и обоснование выбора метода организации входных и выходных данных 10](#_Toc40356309)

[2.3.1. Описание метода организации входных и выходных данных 10](#_Toc40356310)

[2.3.2. Обоснования выбора метода организации входных и выходных данных 10](#_Toc40356311)

[2.4. Описание и обоснование выбора состава технических и программных средств 11](#_Toc40356312)

[2.4.1. Состав технических и программных средств 11](#_Toc40356313)

[2.4.2. Обоснование выбора технических и программных средств 11](#_Toc40356314)

[3. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 12](#_Toc40356315)

[3.1. Предполагаемая потребность 12](#_Toc40356316)

[3.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами 12](#_Toc40356317)

[ПРИЛОЖЕНИЕ 1 13](#_Toc40356318)

[ТЕРМИНОЛОГИЯ 13](#_Toc40356319)

[ПРИЛОЖЕНИЕ 2 14](#_Toc40356320)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 14](#_Toc40356321)

[ПРИЛОЖЕНИЕ 3 14](#_Toc40356322)

[АЛГОРИТМ ПОЛЬЗОВАНИЯ 14](#_Toc40356323)

[ПРИЛОЖЕНИЕ 4 15](#_Toc40356324)

[ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ 15](#_Toc40356325)

[ПРИЛОЖЕНИЕ 5 15](#_Toc40356326)

[ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ, МЕТОДОВ И СВОЙСТВ 15](#_Toc40356327)

# ВВЕДЕНИЕ

## Наименование программы

Наименование программы – «Программа для обнаружения облаков на спутниковых снимках».

## Документы, на основании которых ведется разработка

Приказ декана факультета компьютерных наук И.В. Аржанцева "Об утверждении тем, руководителей курсовых работ студентов образовательной программы «Программная инженерия» факультета компьютерных наук" № 2.3-02/1112-04 от 11.12.2019

# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

## Постановка задачи на разработку программы

Разрабатываемая программа должна соответствовать:

- описанным в техническом задании функциональным характеристикам («Программа для обнаружения облаков на спутниковых снимках». Техническое задание)

- описанным в техническом задании требованиям к интерфейсу («Программа для обнаружения облаков на спутниковых снимках». Техническое задание)

- описанным в техническом задании требованиям к надежности («Программа для обнаружения облаков на спутниковых снимках». Техническое задание)

## Описание алгоритма и функционирования программы

### Описания алгоритма работы взаимодействия сервера и клиента

Программа реализована в виде клиент-серверного веб-приложения.

Клиентское приложение реализовывает следующие функции:

- отправлять запрос серверу содержащий файл со сценой спутника;

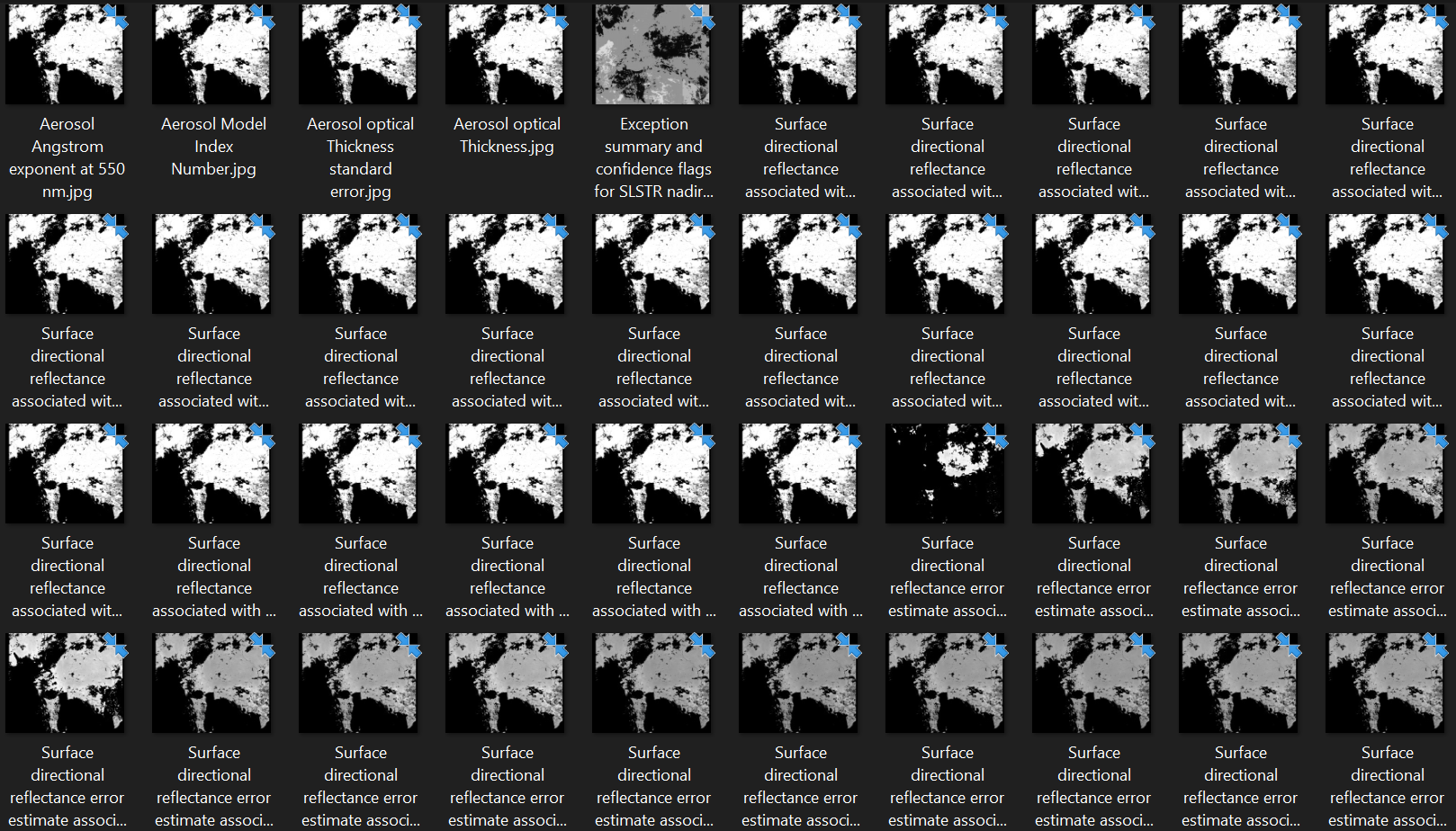
- возвращать файл с информацией о расположении облаков на снимке;

Серверное приложение должно реализовывать следующие функции:

- обрабатывать входящие файлы, возвращать требуемый результат.

### Описания алгоритма детектирования облаков

Алгоритм сравнивает изображения спутникового снимка в разных спектрах, на основе чего делает определяет место положение облаков.



*Рисунок 1.* Спутниковый снимок разбитый на спектры

Данная программа была разработана для обработки снимков спутника **Sentiel-3** м параметром **product type** равным **SY\_2\_SYN\_\_\_** .

Для обработки спутникового снимка запускается отдельный модуль (программа) в файле Cloud.py:

# Python 3.6

import cv2

from netCDF4 import Dataset

from zipfile import ZipFile

import numpy as np

class Channel:

def \_\_init\_\_(self, name, picture, flags="", wavelength=-1):

self.name = name

self.picture = picture

self.flags = flags

self.wavelength = wavelength

self.updateMeta()

def updateMeta(self):

self.min = np.min(self.picture)

self.max = np.max(self.picture)

self.shape = self.picture.shape

def normalise(self):

self.picture -= self.min

self.picture = self.picture \* 255 / np.max(self.picture)

self.updateMeta()

def save(self, folder, printMeta=True):

try:

cv2.imwrite(folder + "/" + self.name + ".jpg", self.picture)

if printMeta: print(self.name + " saved in " + folder)

except:

if printMeta: print("Can't save " + self.name, file=sys.stderr)

def printMeta(self):

print(self.name, '\n',

'min:', self.min,

'max:', self.max,

'size: ', self.shape,)

class Scene:

def \_\_init\_\_(self, filename, onlyPic=False, onlyFull=False):

self.channels = {}

nc = Dataset(filename, "r")

for channelName in nc.variables:

# print(nc.variables[channelName].long\_name)

flags = ''

wavelength = -1

try:

flags = nc.variables[channelName].flag\_meanings

except Exception as e:

pass#print('Empty flags')

try:

wavelength = nc.variables[channelName].wavelength

except Exception as e:

pass#print('Empty wavelength')

channelPicture = np.array(nc.variables[channelName])

channel = Channel(name=nc.variables[channelName].long\_name,

picture=channelPicture, wavelength=wavelength,

flags=flags)

if (len(channel.shape) > 1 or not onlyPic) and \

(channel.shape[0] - channel.shape[1]) \* 10 < channel.shape[0] or not onlyFull:

self.channels[channelName] = channel

try:

with ZipFile('/var/www/FlaskApp/FlaskApp/1.zip', 'r') as zip:

flags\_input = zip.open(zip.filelist[0].filename+ 'flags.nc')

spectrum\_input = zip.open(zip.filelist[0].filename+ 'Syn\_Oa10\_reflectance.nc')

flags\_output = open('/var/www/FlaskApp/FlaskApp/flag.nc', 'wb')

spectrum\_output = open('/var/www/FlaskApp/FlaskApp/Syn\_Oa10\_reflectance.nc', 'wb')

flags\_output.write(flags\_input.read())

spectrum\_input.write(spectrum\_output.read())

flags\_input.close()

flags\_output.close()

spectrum\_input.close()

spectrum\_output.close()

scene1 = Scene('/var/www/FlaskApp/FlaskApp/flag.nc', onlyPic=True, onlyFull=True)

scene2 = Scene('/var/www/FlaskApp/FlaskApp/Syn\_Oa10\_reflectance.nc', onlyPic=True, onlyFull=True)

general\_mask = scene1.channels['OLC\_flags'].picture

cloud\_mask = scene1.channels['CLOUD\_flags'].picture

cloud\_land\_mask = scene2.channels['SDR\_Oa10\_err'].picture

landpik = np.full((general\_mask.shape + (3,)), [255, 187, 153])

landmask = np.zeros(general\_mask.shape)

landpik[general\_mask // 4096 % 2 == 1] = [0, 200, 0] # OLC\_land

landpik[general\_mask // 1024 % 2 == 1] = [100, 0, 0] # OLC\_fresh\_inland\_water

landmask[general\_mask // 4096 % 2 == 1] = 1

landmask[general\_mask // 1024 % 2 == 1] = 0

cloud\_mask[landmask == 1] = 0

landpik[cloud\_mask % 2 == 1] = [250, 250, 250]

cloud\_land\_mask[landmask == 0] = 1000

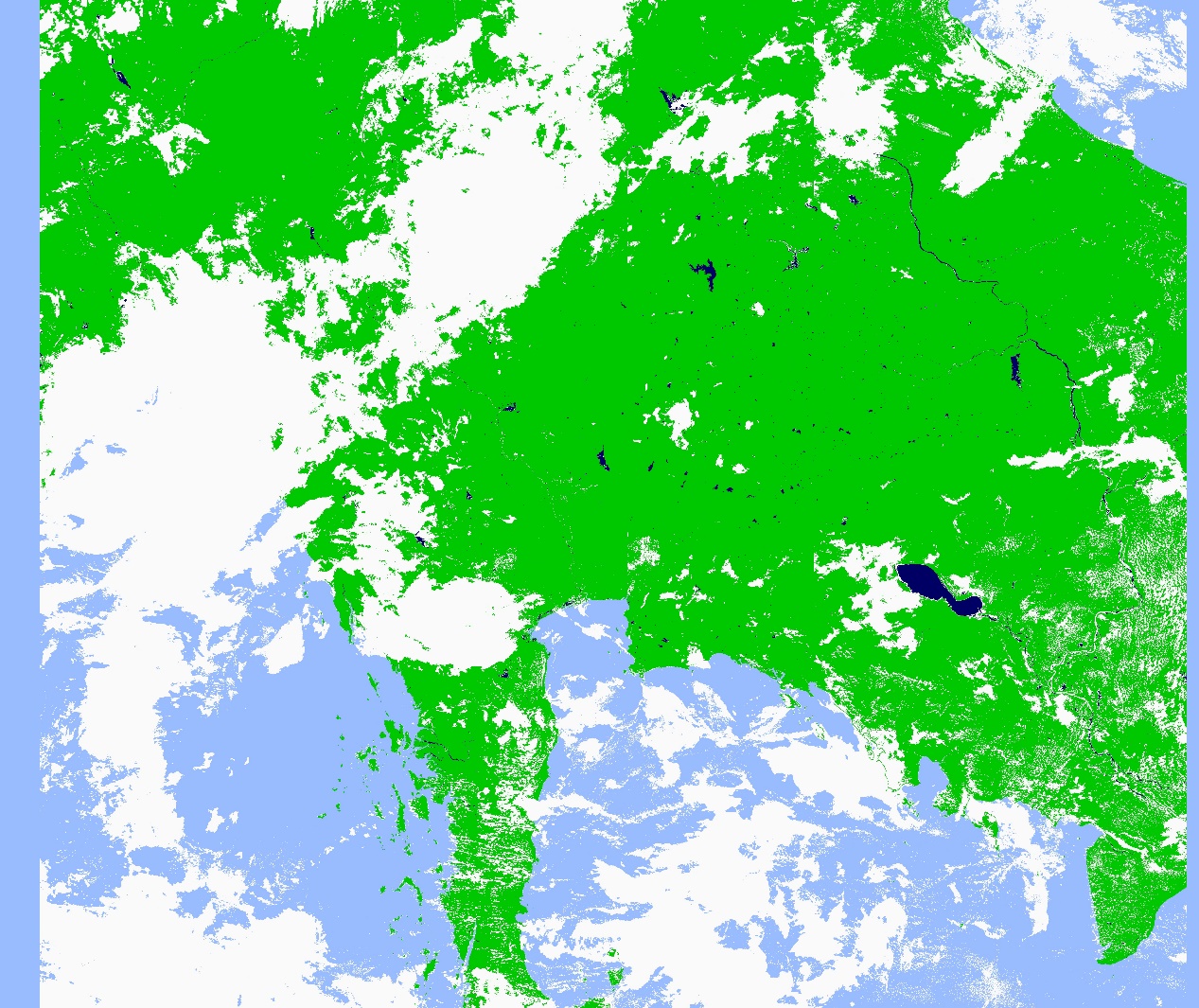
landpik[cloud\_land\_mask < 100] = [250, 250, 250]

cv2.imwrite("/var/www/FlaskApp/FlaskApp/output1.jpg", landpik)

print("ok")

except Exception as e:

print(e)



*Рисунок 2.* Результат выполнение программы

Зелёный цвет – обозначение суши,

Голубой – моря и океаны

Синий – материковые водоёмы

Белый – облака

### Описание работы клиентского приложения

При запросе к сайту <https://shakura.dev/CloudDetect> сервер возвращает html страницу которая функционирует как полноценное клиентское приложение. При загрузки изображения, его отправка происходит на адрес <https://shakura.dev/uploadLabel>.

<html>

<head>

<script>

function uploadFile(file) {

var url = '/uploadLabel';

var xhr = new XMLHttpRequest();

var formData = new FormData();

xhr.open('POST', url, true);

xhr.addEventListener('readystatechange', function(e) {

if (xhr.readyState == 4 && xhr.status == 200) {

// Done. Inform the user

}

else if (xhr.readyState == 4 && xhr.status != 200) {

// Error. Inform the user

}

});

formData.append('file', file);

xhr.send(formData);

}

</script>

<style>

.c {

position: relative;

margin:0;

}

.c img {position: absolute;}

.c div{

position: absolute;

background: whitesmoke;

border-radius: 10px;

text-align: center;

align-content: center;

top: 50%;

left: 50%;

margin-top: -50px;

margin-left: -150px;

}

.c form{

margin: 10px;

}

.c a{

margin: 10px;

}

</style>

</head>

<body class="c">

<img src="/pic/CloudBack.jpg" width="100%" draggable="false">

<div class="box">

<form id="uploadbanner" enctype="multipart/form-data" method="post" action="/uploadLabel">

<input id="fileupload" name="myfile.txt" type="file" />

<input type="submit" value="submit" id="submit" />

</form>

<a target="\_blank" href="https://scihub.copernicus.eu/dhus/#/home">

You can take exemples here.

</a>

</div>

</body>

</html>

### Описание алгоритма работы сервера

После загрузки изображения на сервер, сервер начинает его обрабатывать

from flask import Flask, send\_file, request, json, redirect, render\_template

import urllib.request

import numpy as np

sys.path.append("/var/www/FlaskApp/FlaskApp")

import os

import warnings

from subprocess import check\_output

app = Flask(\_\_name\_\_)

@app.route('/CloudDetect')

def clouddetect():

return send\_file('/var/www/FlaskApp/FlaskApp/static/CloudDetect.html', CloudDetect.html")

@app.route('/uploadLabel',methods=['POST'])

def uploadLabel():

if 'myfile.txt' not in request.files:

return "Error"

file = request.files['myfile.txt']

file.save('/var/www/FlaskApp/FlaskApp/1.zip')

a = check\_output("python3.6 /var/www/FlaskApp/FlaskApp/Cloud.py", shell=True)

app.logger.error(a)

return send\_file('/var/www/FlaskApp/FlaskApp/output1.jpg', 'output1.jpg')

## Описание и обоснование выбора метода организации входных и выходных данных

### Описание метода организации входных и выходных данных

Входные данные в программе – Файл в формате .zip, содержащий файлы-сцены в формате .nc.

Выходные данные в программе – jpg файл с информация о расположении облаков.

### Обоснования выбора метода организации входных и выходных данных

Механизм веб интерфейса был выбран потому, что он позволяет сократить объем исходного кода и восстановить объекты в их первоначальном виде.

## Описание и обоснование выбора состава технических и программных средств

### Состав технических и программных средств

В состав технических средств должен входить персональный компьютер, включающий в себя:

1) не менее 1 Гб оперативной памяти;

2) свободная память на диске не менее 1 Гб;

3) монитор, разрешение не ниже 800х600 пикселей.

4) Устройства ввода\вывода (клавиатура, монитор).

5) Интернет браузер (Chrome/Mozila)

Для сервера:

1) Операционная система Linux.

2) не менее 2 Гб оперативной памяти;

3) свободная память на диске не менее 3 Гб;

4) Устройства ввода\вывода.

### Обоснование выбора технических и программных средств

Данный состав технических и программных средств позволит программе отображать объекты, осуществлять быстрый отклик на действия пользователя и работать без перебоев.

# ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## Предполагаемая потребность

Информация о расположении облаков;

## Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

В рамках данной работы расчет экономической эффективности не предусмотрен.

# ПРИЛОЖЕНИЕ 1

## ТЕРМИНОЛОГИЯ

Ниже приведен список необходимых терминов для ознакомления.

**Сервер –** компьютер с операционной системой, подключенной к сети интеренет.

**Ubuntu –** операционная система для вычислительных машин (в том числе серверов).

**Python –** язык программирования.

**JavaScript –** язык программирования.

# ПРИЛОЖЕНИЕ 2

## СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Copernicus Open Access Hub - <https://scihub.copernicus.eu/dhus/#/home>
2. SYNERGY ALGORITHMS FOR CLOUD SCREENING, AEROSOL RETRIEVAL AND ATMOSPHERIC CORRECTION <https://pdfs.semanticscholar.org/60af/be4998cfd1d787c226c7c9cecddda40c5e5b.pdf>
3. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.505-79 Руководство оператора. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
11. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.

# ПРИЛОЖЕНИЕ 3

## АЛГОРИТМ ПОЛЬЗОВАНИЯ

В главном меню пользователь может выбрать одно из двух действий:

1. Загрузить снимок для анализа
2. Открыть базу данных доступных снимков

При нажатии кнопки загрузки сцены, открывается окно выбора файла, в котором необходимо выбрать ZIP архив с необходимой сценой.

При нажатии на кнопку с примерами спутниковых снимков, происходит перенаправление пользователя на страницу с базой данных снимков спутника.

# ПРИЛОЖЕНИЕ 4

## ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ

Таблица 4.1

Описание и функциональное назначение классов

|  |  |
| --- | --- |
| Класс | Назначение |
| Channel | Класс, представляющий отдельный канал, из которых состоит спутниковая сцен |
| Scene | Класс, представляющий спутниковую сцену, из которых состоит спутниковый снимок |

# ПРИЛОЖЕНИЕ 5

## ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ, МЕТОДОВ И СВОЙСТВ

Таблица 5.1

Описание полей, методов и свойств класса Channel

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Поля** | |  |  | | |
| Имя | | Тип | Назначение | | |
| name | | string | Краткое название канала | | |
| longname | | string | Полное название канала | | |
| picture | | numpy array | Изображение | | |
| flags | | string | Обозначение размеченных флагов (если имеется) | | |
| wavelength | | float | Длинна фолны (если имеется) | | |
| min | | float | Минимальное значение пикселя изображение канала | | |
| max | | float | Максимальное значение пикселя изображение канала | | |
| shape | | tuple int | Форма изображения | | |
| **Методы** | |  |  | | |
| Имя | Аргументы | | | Назначение |
| \_\_init\_\_ | name, longname, picture, flags="", wavelength=-1 | | | Конструктор класса |
| updateMeta | - | | | Обновляет поля min, max, shape |
| normalise | - | | | Сдвигает спектр изображения канала до интервала [0; 255] |
| save | folder, printMeta=True | | | Сохраняет изображение с канала как картинку в указанный файл |
| printMeta | - | | | печатает метаинформацию о канале |

Таблица 5.2

Описание полей, методов и свойств класса Scene:

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** | |  |  |
| Имя | | Тип | Назначение |
| channels | | Dictionary of Channtls | Список всех каналов в сцене |
| **Методы** | |  |  |
| Имя | Аргументы | | Назначение |
| \_\_init\_\_ | filename, onlyPic=False, onlyFull=False | | Конструктор класса |

Таблица 5.12

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированных |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |