On the issue of software for environmental information systems

Daria Gonchar¹, Svetlana Nikulina², Natalia Fateeva¹, Joanna Belova¹, Alexander Vasyukov³, and Ksenia Tishchenko*¹

¹Kaluga Branch of the Federal State Budgetary Educational Institution of Higher Education "Bauman Moscow State Technical University (National Research University)", Kaluga, Russia
²Federal State Autonomous Educational Institution of Higher Education "Peoples' Friendship University of Russia", Moscow, Russia
³Federal State Budgetary Educational Institution of Higher Professional Education "Kaluga State University named after K.E. Tsiolkovsky", Kaluga, Russia

Abstract. The increase in the number of sources of pollutants into the atmospheric air is one of the largest environmental problems in the modern world. In order to control emissions of harmful substances, calculations of maximum permissible concentrations of pollutants are used. Calculations of emission sources and maximum permissible concentrations are quite complex, and it is impractical to carry them out manually, since the calculation process takes a lot of time, while there is a high probability of errors. To simplify such a task, special software tools are used – unified programs for calculating atmospheric pollution (UPRZA). The article is devoted to the issues of increasing the level of automation of calculations of emission sources and maximum permissible concentrations using UPRZA. The issues of the standard structure of the UPRSA are considered, input parameters, algorithms for calculating the parameters of the emission source and maximum permissible concentrations are determined in accordance with the methods used. Methods of increasing the level of automation are proposed. The analysis of possible ways of development of these software complexes is carried out. The article describes the process of developing a unified program for calculating atmospheric pollution, the main advantages of its application.

1 Introduction

Calculations of emission sources and maximum permissible concentrations are quite complex, and it is impractical to carry them out manually, since the calculation process takes a lot of time. At the same time, the probability of errors is high. To simplify such a task, special software tools are used – unified programs for calculating atmospheric pollution (UPRZA). Modern UPRZA allow you to perform calculations with an accuracy of up to ninety-seven percent relative to the reference calculations.

* Corresponding author: Ksenia45608@mail.ru

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).
However, large volumes of source data containing information about emissions and their sources make calculations difficult. Data entry and validation is a time-consuming process.

The task of developing an UPRZA with an increased level of automation, which will allow collecting and analyzing input data for performing calculations of atmospheric pollution without user participation, is very relevant.

The scientific novelty of the development lies in the presence of a parser module for automatic data collection and analysis for UPRZA. Modern software tools for pollution calculations do not have such functionality, which significantly slows down and complicates work with them.

2 The initial data for the calculations in the UPRZA

Emissions of pollutants have a set of parameters, among which the value of the atmospheric air quality criterion (maximum permissible concentration (MPC) of pollutants) plays an important role. As a result, the results of calculations of the concentrations of substances in the atmosphere are compared with this value.

The necessary input data for all operating UPRZА are most often the same - climatic characteristics of the area. The increased dispersion of pollutants of the air mixture in the atmosphere is mainly influenced by the value of the outdoor air temperature, the coefficient of temperature stratification of the atmosphere and the value of wind speed. Its dispersed composition has a great influence on the dispersion of the substance, for which the settling coefficient of the substance is introduced [1,2].

A number of factors affect the process of dispersion of pollutants in the atmospheric air:
- stability of the atmosphere;
- the nature of the location of enterprises on the ground, relief;
- height and dimensions of industrial buildings, their dimensions and height;
- mouth diameter and height of the discharge source (pipe);
- temperature and emission density;
- the aggregate state of pollutants, etc.

The dispersion of pollutants in the atmosphere depends on meteorological factors:
- wind speed and direction;
- presence and type of inversions;
- temperature stratification;
- air humidity;
- atmospheric pressure;
- calms, precipitation, fog, etc.

As a rule, when developing projects for calculating maximum permissible emissions, concentrations of harmful substances are considered only in the surface layer, that is, averaged over a height of 2 m. To solve other problems, it may be necessary to determine the values of concentrations of pollutants at different heights from ground level. Not all UPRZA can carry out such calculations.

Also, a number of techniques for its sub-sectors can be developed for each industry. For example, if there is one approved methodology for enterprises of the coke chemical industry, then nine different methods are in effect for various sub-sectors of non-ferrous metallurgy. Not all UPRZA have such a variety of options for calculating atmospheric air pollution. The calculation schemes underlying each methodology differ from each other, often fundamentally, and depend on a specific branch of the national economy [3].

Different developers usually implement approximately the same set of techniques in the form of programs. First of all, it includes methods for calculating emissions of pollutants
from motor vehicles, thermal power plants, oil and gas industries, characterized by complex
calculation schemes and the availability of extensive reference material.

When performing calculations in the UPRZA for a certain enterprise, the concept of
background concentration of pollutants is used – the concentration created by all sources of
emissions, except for the selected one, in this region. Usually, the value of background
concentrations can be requested at the Hydrometeorological Service posts [4].

3 Fundamentals of the functioning and use of the UPRZA

The basis for the development of a new UPRZA is the availability of source data and the ease
of their further use, which will allow users who do not have special knowledge in the field of
atmospheric diffusion to work with software for calculating atmospheric pollution.

The principle of operation of all existing UPRZ is almost the same. Such programs allow
calculating concentrations of pollutants in the atmospheric air based on input data on the
source of emissions and climatic conditions of the region. These programs can be used in
calculations for a wide variety of sources of emissions of harmful substances, which makes
them unified. The results are calculated based on the accepted methods [3].

The approximate sequence of operation of the UPRZA is as follows:
- Entering input data.
  - formation of reference information (information about the structure of the
    facility and sources of emissions);
  - mandatory verification of the completeness and relevance of the directory of
    substances and groups of summation of pollutants;
  - recording of meteorological data, values of background concentrations, data
    on emission sources and their parameters;
  - adding other source data, working with the topographical basis, entering
    buildings, special zones, etc.;
  - setting calculation conditions (creating settlement points and settlement sites)
- Calculation and analysis of the dispersion of sound.
- Generating a report on the results.
  - presentation of results in tabular form;
  - presentation of results on the map of the region (graphic form);
  - selection of a file format with a ready-made report on the results.

Input of initial data is a laborious process due to the fact that most of the values are entered
manually by both developers and users, creating the possibility of errors. A variety of
different criteria and values make the calculation task more difficult by increasing the
processing time of input data and further calculations [5].

One of the important features that UPRZA should have is their ability to work as part of
the systems of calculated monitoring of atmospheric air quality, together with other
programs. At the same time, it is possible to integrate the calculation results into other
programs for further use [6].

For users of UPRZA, an important criterion is the presence of an intuitive interface, which
allows users to avoid difficulties with their development. Complex terminology can also
negatively affect the quality of work of users who do not have special knowledge in the field
of protection of the atmospheric environment. The existing UPRSA have a large set of
functions, some of which may not be used in the calculation process, in this case a "visual
noise" is created. One of the ways to solve this problem is to create separate modules with
different functionality, which, if necessary, can be connected to the main system.

Another necessary criterion for choosing a suitable UPRZA is compatibility with various
operating systems. Often the lack of such variability causes difficulties for users.
Every year, based on the testing of new UPRZA, the Ministry of Natural Resources of the Russian Federation draws up an official list of existing programs. Currently, in Russia and the CIS countries, enterprises-nature users and other organizations use about 4,000 copies of existing UPRZ. The most common UPRZA are "Ecologist", "Ecologist PRO", "Prism", "Atmosphere", "LiDa".

After analyzing the frequently used UPRZA from the point of view of functional content and user interface, it is possible to determine the criteria necessary for convenient use of them at the present time:

- Accounting of all necessary parameters for calculations.
- Availability of reference books.
- The graphic editor module.
- Presentation of calculation results in various forms: tabular, graphic, etc.
- Maximum software automation.
- Compatible with other programs for importing and exporting data in various formats.
- Simple and intuitive interface.
- Compatible with various operating systems.

Unfortunately, none of the existing UPRZA meets all the listed requirements at the same time.

4 Increasing the automation of UPRZA due to parsing

The field of development of UPRZA has many prospects for development. One of the promising tasks is to increase the level of automation of the system. Thanks to modern programming capabilities, many functions have become available that simplify the development and operation of software tools. Manual input of both reference and source data on emissions and weather conditions entails a loss of time and increases the likelihood of errors, which affects the quality of information processing. To simplify the development and further use of UPRZA, it is proposed to use a database and website parsing module.

Parsing is the collection of necessary data, their structuring and processing without direct human involvement. Data is collected through site parsing. With voluminous input data, this process simplifies the use of software (software). Parsing allows you to get data in a short time, provides an opportunity to configure a variety of criteria for sampling data from sites and the frequency of their collection [7, 8].

Parsing consists of the following steps:

- Selection of the site (or several sites) from which the information will be read;
- Setting data sampling conditions;
- Collecting relevant information;
- Sorting data;
- Conducting error checking;
- Creating a report in the required format or directly uploading data to the next module.

Let's consider the process of creating a parser module for UPRZA. The object-oriented programming language C# is chosen as the language for writing this function. The object-oriented approach allows you to develop large, but at the same time flexible, scalable and extensible applications. This approach is very popular due to the convenience of using objects. C# is used in all subject areas, it allows you to describe bulky constructions and build connections between them by referring to functions and methods.
C# is a cross between C++ and Java programming languages, due to which it is convenient to use it in combination with metadata markup languages. It also allows you to work with both XML documents and XPath.

XPath or XML Path Language is a query language that allows you to navigate through XML documents. XML is one of the metadata markup languages that does not perform any actions, but serves to describe and store structured data. XPath is used to navigate to the necessary objects in a text block and is used in conjunction with various technologies, such as XSLT, XQuery, XLink and XPointer.

5 Development of a module for parsing websites

To receive data from the site, the application must execute a web request (HTTP request) that is indistinguishable from browser requests. A web request is part of an interaction in a client-server relationship. To do this, the Get_Req and Post_Req classes are used.

These classes perform Get and Post requests, in which the address of the browser's web page is passed to the class constructor.

The type of HTTP request (also called HTTP method) indicates to the server what action we want to perform with the resource. In this example, only two methods are used: GET - getting a resource and POST - creating a resource.

The Referer property defines the header of the client's request and contains the URL of the request source. Useragent allows you to get the source string of the user agent passed from the client browser. Proxy is used to install a proxy server through which web requests are sent.

Creating the Get_Req class and declaring operands:

```csharp
public class Get_Req
{
    HttpWebRequest _request;
    string _address;

    public Dictionary<string, string> Headers { get; set; }

    public string Response { get; set; }
    public string Accept { get; set; }
    public string Host { get; set; }
    public string Referer { get; set; }
    public string Useragent { get; set; }
    public WebProxy Proxy { get; set; }

    public GetRequest(string address)
    {
        _address = address;
        Headers = new Dictionary<string, string>();
    }
}
```

Creating a Post_Req class and declaring operands:

```csharp
public class Post_Req
{
    HttpWebRequest _request;
    string _address;
```
public Dictionary<string, string> Headers { get; set; } 

public string Response { get; set; } 
public string Accept { get; set; } 
public string Host { get; set; } 
public string Data { get; set; } 
public string ContentType { get; set; } 
public WebProxy Proxy { get; set; } 
public string Referer { get; set; } 
public string Useragent { get; set; } 

public PostRequest(string address)
{
    _address = address;
    Headers = new Dictionary<string, string>();
}

Calling the Run method starts executing the web request. After the requests are executed, 
the Cookie container records the data received from the web portal, which allows subsequent 
requests to be made with the stored cookie data. The results of query execution are placed in 
the Response property.

Fragment of the Run method:
public void Run(CookieContainer cookieContainer)
{
    _request = (HttpWebRequest)WebRequest.Create(_address);
    _request.Method = "Get";
    _request.CookieContainer = cookieContainer;
    _request.Proxy = Proxy;
    _request.Accept = Accept;
    _request.Host = Host;
    _request.Referer = Referer;
    _request.UserAgent = Useragent;
}

Next, a container is created to receive the results and output them. A Proxy server is set 
through which the request is sent, and a new container for cookie data.
Fragment of container creation:
static void Main(string[] args)
{
    var proxy = new WebProxy("127.0.0.1:8888");
    var cookieContainer = new CookieContainer();

    We will specify a criterion for data sampling. In this case, the date and approximate time 
are used:

    var date = "24.07.2022";
    var time = "16:00";

    In the future, the DateTime system property should be added to this criterion. Today, to 
automatically enter the current date and time by default.

    To get the address of weather data on a set date, we create the execution of the request 
itself.
Request formation:

```javascript
var postRequest = new PostRequest("https://rambler.weather.ru/");
post_Req.Data = "$ajax_call=y&INPUT_ID=title-search-
input&d={date}&t={time}&l=2";
post_Req.Accept = ";/*/*";
post_Req.Useragent = "Yandex/5.0 (Windows NT 10.0; WOW64; Trident/7.0;
r:11.0) like Gecko";
post_Req.ContentType = "application/x-www-form-urlencoded";
post_Req.Referer = "https://rambler.weather.ru/";
post_Req.Host = "rambler.weather.ru";
post_Req.Proxy = proxy;
post_Req.Headers.Add("Bx-ajax", "true");
post_Req.Run(cookieContainer);
```

The next step is to search for the address of the weather data in the HTML code:

```javascript
var strStart = post_Req.Response.IndexOf("search-result-group search-result-
weather");
strStart = post_Req.Response.IndexOf("<a href="", strStart) + 9;
var strEnd = post_Req.Response.IndexOf("","", strStart);
var getPath = post_Req.Response.Substring(strStart, strEnd - strStart);
```

After the required data is found, it must be read and displayed on the screen. For example, only one data type is specified – card.weather, which includes the following values: cloudy, sunny, rain, thunderstorm, partly cloudy, etc. Several such data types are created for each value required for calculations.

Saving data and output to the console:

```javascript
var getRequest = new GetRequest("https://rambler.weather.ru{getPath}");
post_Req.Accept = ";text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apn
g,/*;q=0.8,application/signed-exchange;v=b3;q=0.9";
post_Req.Useragent = "Yandex/5.0 (Windows NT 10.0; WOW64; Trident/7.0;
r:11.0) like Gecko";
post_Req.Referer = "https://rambler.weather.ru/";
post_Req.Host = "rambler.weather.ru";
post_Req.Proxy = proxy;
post_Req.Run(cookieContainer);
```

```javascript
Console.WriteLine("weather={card.weather}");
```

```javascript
Console.ReadKey();
```

Fig. 1. Results of the parser module program execution.
The result of the program contained in the module for parsing is the meteorological data of the region on the specified date. It is not necessary to display this information on the console or on the screen, it is enough to check them to exclude errors and redirect the values to the meteorological data input lines for calculations in the UPRZA itself. The execution time of the parsing task presented in the article was 3.12 seconds.

The use of this site parsing module will reduce the calculation time in the UPRZA, due to the automatic input of the source data. The execution of web requests starts and stops by the user's decision.

6 Conclusions

With the advent of UPRZA, the process of calculating atmospheric pollution has become much simpler and clearer. This allows for more effective control over the current state of emissions of harmful substances into the atmosphere.

Analyzing the already existing UPRSA, the authors of the article, in order to solve the task, made a choice in favor of creating a new system for calculating atmospheric pollution with separate modules of additional functionality. This method of organizing the structure of the new software will simplify working with it for future users.

The article presents fragments of the development, as well as the results of the site parsing module for the new UPRZA. The presence of such a module will make working with the program for calculating pollutants in the atmospheric air much more convenient. The chosen language for development, C#, will allow us to further refine the new system by adding new necessary functions, since it is an extensible programming language.

In conclusion, we can say that the task of increasing the automation of the developed UPRZA has been completed. Preliminary testing showed that the work of the parser module passes without errors and gives the expected result at the output.

References

1. A.A. Malyshev, N.N. Solodkov, N.A. Korobkova, Theoretical and Applied Ecology 2, 93-100 (2022)