International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 11 (2020), pp. 3555-3558
 https://dx.doi.org/10.37624/IJERT/13.11.2020.3555-3558

Method of Geodata Processing

Shamil Aktasovitch Khamadeev[1], Lenar Ajratovich Galiullin[1]

[1]PhD, Associate Professor, Kazan Federal University,
ID Scopus: 57211321538, ORCID: 0000-0002-6981-6214,

[2]PhD, Associate Professor, Kazan Federal University,
ID Scopus: 39361435200, ORCID: 0000-0001-8640-1734,

Abstract

An analysis of the developed system was carried out. As a result of the analysis, the main precedents of the system were determined, a BPMN diagram was built, an alternative solution was analyzed, and the basic requirements for the system were derived. The basic algorithms of the system were designed, a block diagram of these algorithms was built, and the features of working with OSM data were considered. The technologies used to implement the project were affected. The main application packages are described and the classes of which the main packages are composed. As a result, an application was obtained that meets the requirements set for it. The mapping data update was accelerated, the quality of the received data has not changed, the OSM data processing algorithms work in accordance with the tasks. The software implementation of the project was completed. The following works were performed: the technologies used were described, the software packages were described, class diagrams for all application packages were described. application design has been completed. Six algorithms have been developed that describe the basic processes necessary for the successful operation of the system. These processes are: OSM data processing, Node processing, Way processing, Relation processing, Data writing to the database. Some difficulties that may arise when working with OSM data were also considered.

Keywords: Geodata; Map; System; Programming; System.

I. INTRODUCTION

The use of cartographic data for the operation of various kinds of applications is now becoming more and more relevant [1]. Typical applications of this kind are navigators and food or taxi services. The popularity of these applications lies in the fact that these applications are easy to use and give the client some visibility when using [2]. The taxi service “Maxim” offers customers the opportunity to order a taxi in a large number of cities, both in Russia and abroad. The company has an application that uses map data for its work. To use the map in the application, you need to have a source that will supply map data, in this case, the source is OpenStreetMap [3]. OpenStreetMap, abbreviated as OSM, is a non-profit web-mapping project for creating a detailed free and free geographical map of the world. Data from OSM contains a rather large amount of information and for most application cards this amount of information is redundant, therefore, incoming information must be processed [4]. It should also be borne in mind that cartographic data is periodically updated, therefore maps in the application must be updated for these changes. Given the above circumstances, the company has developed an additional application for data processing. Recently, the company has been developing rapidly, opening branches in many countries, which has increased the load on the data processing application [5]. The processing process has slowed down significantly, which affects the speed of updating maps, so the application needs to be upgraded. Thus, the relevance of this work is to accelerate the processing of data from OSM.

II. METHODS

The current implementation of the processing process consists of several stages:

1. Downloading data for processing.

2. Data processing using additional software.

3. Uploading processing data to the database.

4. Processing database data using SQL scripts.

This implementation is not the most optimal, since the application acts as an intermediary between the additional software and the database [6]. Most of the time the application is in standby mode. Executing SQL scripts also takes a lot of time, and the additional software used requires quite a lot of memory resources to work.

To solve these problems, it was decided to include all the operations performed in the application. The new data processing option eliminates program downtime and will also increase system performance due to the lack of work on the database side.

There are many products that can process data from OSM [7]. In this case, we will consider two of them: osm2pgsql and Imposm 3.

osm2pgsql is a utility that converts the OpenStreetMap data format into data for loading into a PostgreSQL database [8].

This utility converts data with losses. It contributes only those objects that have tags defined in the configuration file, and it converts points and lines to lines and polygons. With the help of this program, the current version of the application works and its cons are described above [9].

Imposm 3 is a utility that works similar to osm2pgsql, but this utility is developed using other technologies and works much faster [10]. This utility is not suitable, because it will act as additional software for our application, and before processing the data, it creates additional NoSQL databases from them, due to which data processing is accelerated, but this increases memory costs. Another disadvantage is the difficulty of adapting the processed data from this software to the database schema used by the company.

III. RESULTS AND DISCUSSION

Based on the result obtained during the analysis of the system, the basic algorithms necessary for the successful operation of the system were compiled.

The system starts by entering the basic parameters. In fact, this step is not always performed, since the system can be started without entering parameters, then it will work with default parameters. This is done to simplify the work with the system for the end user. Next, read the settings from the configuration files, which contain information about countries for which data can be updated. With the help of the main parameters, you can restrict data updates by country or, on the contrary, update country data, despite any restrictions. Next is a check for the need to update data. If nothing is specified in the main parameters, a situation may occur in which data updates for countries will not be made. This is because the system checks the database for information on the country whose data you want to update. And if the database for this country is stored, then the update will not be performed.

As soon as all countries for updating are identified, the process of downloading files with OSM data will begin. In parallel with this process, the operation of downloading the substrate for the card will begin. In this case, under the substrate is meant the basis for the card. The base is the outline of the continents and oceans in simple colors. On top of this foundation, OSM data will eventually be drawn. If the data downloading process was successful, then the data processing process begins.

The process of data processing is followed by the process of storing the received information in the database. These two processes will be discussed in detail later. After saving the data regardless of the result, a report will be generated on the program. The content of the report will depend on the success of data recording. As soon as the report is generated, the application will end.

The process of processing data by country, or more precisely the processing of data from OSM, is the main application process, which takes up most of the program and is most difficult to design. This process will be divided into four parts. The first part will describe the main part of the process, and the remaining three will be handling OSM entities. The beginning of the data processing algorithm begins with reading the OSM file. It should be noted here that the file is read line by line, which means that you cannot return to the same object twice. As a result, there are some difficulties when processing OSM entities. Reading a file occurs line by line due to the fact that the file size contains a huge amount of information and storing a file in memory all the time is a very expensive procedure. After the start of reading the file, the system will begin to receive OSM entities and will receive them until the file is read to the end. According to the format accepted in OpenStreetMap, there are 4 types of entities:

-Bound;

-Node;

-Way;

-Relation.

We are only interested in Node, Way and Relation. Each of these entities will be processed by the corresponding algorithm. In the file for processing, all Nodes are listed in order, then Way and at the end are Relation. If the entity is necessary for updating the mapping data, then it will be processed for subsequent writing to the database. If not all necessary entities are processed before the file is read to the end, then the file will be read again.

Each object in OSM is represented by a set of points or even a single point. That point is the minimum unit of this cartographic resource. Each object in OSM is represented by a set of points or even a single point. That point is the minimum unit of this cartographic resource. Point is a basic element in the OSM data structure. A point has the parameters “latitude” and “longitude”.

Points are used to define a line, however, a point can also be an independent element of the map, and can be used to designate a separate, unrelated object.

The points that make up the line often have no properties and are needed only to describe the line; however, this is not an unshakable rule. Figure 6 shows how Node is displayed on the official OSM website.

However, this is a graphical representation of a point; data will come to the system in the format of a binary file from which data will need to be extracted.

The Node processing algorithm is the basis for all other OSM entity processing algorithms used in this project. The algorithm begins by checking for tags at the point. If the point has tags, then it is checked whether the tag data is needed for our data. In the case when the tag is suitable, all the data necessary for drawing the point is extracted from the point. Otherwise, the id of this point is checked. If the point is part of another object, then all the data necessary for rendering is also extracted from Node. If the point has not passed through any parameters, the point is ignored.

The second type of entity that we need to process is Way or a different line. A line is an ordered relationship of at least

two and no more than two thousand points that describe linear objects, such as streets, paths, railways, and so on.

The lines are closed and open. Closed lines can be polygons if, in addition to the line itself, the object includes an area that this line limits.

Processing Way at first is no different from processing Node. First, the tags and id of the OSM entity are also checked, and the need to process this entity is revealed. Differences begin during the line information collection process. Since the line consists of points, in order to build a given line it is necessary to obtain information about all these points. In the line itself there is only a link to these points. Due to the fact that all OSM entities are read in turn, it is impossible to obtain information about these points in a single pass through the file. Therefore, Way builds in two file reads. For the first time, we collect a list of points necessary for assembling the line and all the necessary tags. In the second pass, we collect information about the points we need and collect the line.

The third type of data to process is Relation. Relations are used to indicate the geographical relationship between different objects. This entity is the most complex OSM data element. The complexity of Relation is that a relation can consist of any type of entities discussed earlier, including the relation itself.

IV. SUMMARY

In the course of work, the process of updating the cartography data of the taxi order service was analyzed. As a result of the analysis, it was found that the processing speed from OSM is not acceptable, and also that this process consumes a large amount of memory resources. When considering the problem process, it was revealed that in the data processing application a large number of downtime due to working with additional software and a database. As a result, it was decided to modernize the process of processing cartographic data.

To achieve the goal, the following steps were taken:

- An analysis was made of existing systems for processing OSM data.

- Business requirements for the system were identified.

- User requirements have been identified.

- Functional requirements for the system have been developed.

- Non-functional requirements have been developed /

Based on the data received, you can begin designing and developing a new version of the OSM data processing application.

V. CONCLUSIONS

When working with OSM data, certain difficulties may arise that lead to incorrect operation of the system and its algorithms. These difficulties arise from the fact that OpenStreetMap is an open web mapping project supported by the efforts of its members. The data is contributed by many people from all over the world, and although the data passes a certain check, errors sometimes appear in them or the data is not formatted according to OSM standards.

One of the problems that may occur is the lack of an entity in the file. Basically, this problem arises because there is too much relation in the file that goes beyond the processed file, and some of its dependent parts are missing from the file. This problem leads to the fact that the entity cannot be fully assembled, and while the system has not developed entities, it does not stop working. To solve this problem, the following was done: if no entity processing occurs during the file reading, it is considered that the remaining entities have parts that go beyond the file and the entity data is removed from the processing queue.

Another problem is the lack of specific tags for the collected entity. Relationship geometries, that is, their set of coordinates is collected based on the geometry of their dependent parts. These dependent parts have certain tags by which you can understand that this entity is part of the geometry. Tags are specified by the OSM editors themselves, and sometimes they forget to put down the necessary tags, or worse, indicate the wrong tags. If there is no tag, the geometry will be incomplete, which can be checked using the geometry validation, however this rule only works for closed geometries. If the editor indicates the wrong tags, then the geometry may pass validation, but it will be incorrect and this problem will be noticeable when rendering data. Unfortunately, this problem is very difficult to fix on the part of the system, since automatic validation does not give absolute results, and manual viewing of all geometries is not advisable. Basically, this problem should be solved by the OSM editors themselves, who periodically fix these errors.

Sometimes problems arise when assembling entity geometries. An entity of type Relation contains within itself references to entities that are its constituent parts. The constituent parts are in order, and all that is needed is to assemble the geometries of the constituent parts of the entity one after another until a complete geometry is obtained. This rule is true if the relation consists only of points. However, if there is a Way in the component parts. In this case, certain difficulties may arise. To assemble a Relation consisting of paths, it is necessary to assemble the geometries of these paths. Each of these paths has its own starting point and its ending point. And to connect these paths, it is necessary that the starting point of one path coincides with the ending point of the other path. But sometimes, instead, these two paths may coincide with either start or end points. If this happens, the geometry of one of the paths is turned over in order to coincide with the other geometry.

ACKNOWLEDGEMENTS

The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

REFERENCES

[1] Galiullin LA, Galiullin IA. Development of Methods and Tools for the Internal Combustion Engines Diagnostics. Journal of Computational and Theoretical Nanoscience. 2019 Nov 1;16(11):4565-8.

[2] Galiullin LA, Galiullin IA. *Fault diagnostic method for Ic engines.* Journal of Advanced Research in Dynamical and Control Systems. 2019;11(8):2273-2279.

[3] Khuzyatova LB, Galiullin LA. *Optimization of parameters of neuro-fuzzy model.* Indonesian Journal of Electrical Engineering and Computer Science. 2019;17(3):1206-1209.

[4] Khuzyatov S, Galiullin L. *Algorithm of clusters organization for internal combustion engine parts manufacturing.* Journal of Advanced Research in Dynamical and Control Systems. 2019;11(8):1758-1761.

[5] Iliukhin AN, Galiullin LA. *Automation of the production of ICE parts based on cluster analysis.* Journal of Advanced Research in Dynamical and Control Systems. 2019;11(8):1779-1782.

[6] Valiev R, Bochkov V, Bashkirov S, Romanov E, Chistjakov V. Mössbauer study of surface layers of high-speed steel after laser treatment. Hyperfine Interactions. 1992 Apr 1;69(1-4):589-92.

[7] Valiev RA, Gaisin FM, Romanov ES, Shakirov YI. Synthesis of iron oxide powders in a discharge with a liquid electrode. Physics and chemistry of materials treatment. 1991 Nov;25(6):654-8.

[8] Valiev RA, Gajsin FM, Romanov ES, Shakirov YuI. *Synthesis of ferric oxide powders in liquid electrode discharge.* Fizika i Khimiya Obrabotki Materialov. 1991;6:90-95.

[9] Valiev RA, Gaisin FM, Shakirov YI. Special traits of powder obtained in discharge between steel electrode and electrolyte. Soviet Powder Metallurgy and Metal Ceramics. 1991 Jun 1;30(6):448-50.

[10] Valiev RA, GAISIN F, Shakirov YI. Properties of the powder produced in a discharge between steel electrode and electrolyte. Poroškovaâ metallurgiâ (Kiev). 1991(6):4-7.

BIOGRAPHIES OF AUTHORS

Shamil Aktasovitch Khamadeev – Associate Professor, Naberezhnye Chelny Institute (branch) KFU/Higher Engineering School/Department of Information Technology and Energy Systems/Department of Information Systems, NI. Academic degrees: Candidate (technical sciences), specialty 05.13.06 – Automation and control of technological processes and production (by industry). Knowledge of languages: English (Independent Speaker).

Lenar Ajratovich Galiullin – Associate Professor, Naberezhnye Chelny Institute (branch) KFU/Higher Engineering School/Department of Information Technology and Energy Systems/Department of Information Systems, NI. Academic degrees: Candidate (technical sciences), specialty 05.13.06 – Automation and control of technological processes and production (by industry), the title of the dissertation "Automation of the technological process of diesel testing based on the fuzzy neural network method".