электронный журнал
МОЛОДЕЖНЫЙ
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО «Московский государственный технический университет им. Н.Э. Баумана»

УДК 004.4’413

Синтаксический анализатор калибровочных файлов системы спутниковой навигации слежения за автотранспортом

И.В. Санеев

Студент,
кафедра ИУ3 «Информационные системы и телекоммуникации»

Научный руководитель: В.М. Недашковский,
к. т. н., доцент кафедры ИУ3 «Информационные системы и телекоммуникации»

МГТУ им. Н.Э. Баумана
saneevil@mail.ru

Система спутниковой навигации автотранспорта «Мониторинг автотранспорта», использующая глобальную систему дистанционного ГЛОНАСС/GPS мониторинга, разрабатывается в основном для организаций, имеющих в своем распоряжении от нескольких единиц до нескольких сотен единиц автотранспорта. Основными задачами данной системы является:

- слежение за координатами объекта в горизонтальной плоскости и по высоте;
- управление и анализ текущего состояния подвижных и статических объектов;
- контроль режимов работы транспортных средств.

Эффективность управления организаций, с большим количеством транспортных средств, существенно зависит от качества организации рабочих мест диспетчеров, которые постоянно, в реальном режиме, отслеживают каждую транспортную единицу. Диспетчеры в любой момент времени должны иметь возможность увидеть на экране своего компьютера карту местности, где могут проезжать или проезжали подконтрольные автомобили. Важно, чтобы карты появлялись на экране диспетчеров без заметных задержек, быстро менялись масштабы и прорисовывались маршруты движения машин с указанием всей необходимой справочной информации о них.

В системе спутниковой навигации существует два типа карт:

- online.Карты предоставляемые интернет сервисами (например, google или yandex), время работы которых определяется скоростью подключения к сети интернет, а также быстродействием ActiveX-компонента в котором они отображается;
- offline.Карты хранятся на рабочем месте диспетчеров и загружаются непосредственно с жесткого диска, и не требуют подключения к сети интернет. Они представляют собой растровые файлы одного из следующих форматов bmp, tiff, png, jpg, ozf, ozf2, ozf3 и сопровождаются калибровочными файлами с расширением *.map. Время загрузки и работы offline карт зависит от нескольких факторов. Одними из самых значительных из них является конфигурация компьютера, которая определяется диспетчером самостоятельно, и модуль синтаксического разбора калибровочных файлов, который

загружает основные данные карты. Этот модуль содержит немалое количество текстовой информации и нуждается в предварительной синтаксической проверке.
Время загрузки карты должно быть минимальным даже на недорогих персональных компьютерах.

Соответственно быстродействие загрузки карты в немалой степени определяется временем работы модуля синтаксического разбора калибровочных файлов, о котором и пойдет речь в данной статье.

Для того, чтобы понять работу синтаксического анализатора в первую очередь необходимо показать данные, с которыми он имеет дело. Эти данные представляют собой калибровочный файл, который содержит подробную информацию о карте.

Структура калибровочного файла карты представляет собой набор директив, следующих в определенной последовательности, несущих информацию о калибровке карты, проекциях карты и ссылку на графический файл.
Краткое описание некоторых директив калибровочного файла:
1. Заголовок и версия файла калибровки.
Пример: OziExplorer Map Data File Version 2.1
2.Название карты, любая текстовая строка.
Пример: Brisbane Region AC
3.Ссылка на файл с образом карты.
Пример: D:\OziMaps\regional\south.ozf2
4. Настройки “Datum”.
Пример: WGS 84,, 0.0000, 0.0000,WGS 84
5. Проекция карты.
Пример: Map Projection, Lambert Conformal Conic,PolyCal,No,AutoCalOnly .
6.Точки калибровки. Их количество всегда составляет 30.
Пример:
Point01,xy, 494, 235,in, deg, 24, 0,S, 148, 0,E, grid, , , ,S
Point02,xy, 4076, 238,in, deg, 24, 0,S, 154, 0,E, grid, , , ,S
...
Point30,xy, , ,in, deg, , ,N, , ,W, grid, , , ,N
7. Количество граничных точек карты.
Пример:
MMPNUM,4
Записи координаты x,y каждой граничной точки должны быть обязательно указаны. Используется в программах, работающих с данного типа карт для определения границ карты.

Пример:
MMPXY,1,494,234
MMPXY,2,4076,238
MMPXY,3,4012,2855
MMPXY,4,549,2852

Так же должны быть указаны соответствующие графические координаты (широта и долгота) для каждой граничной точки.

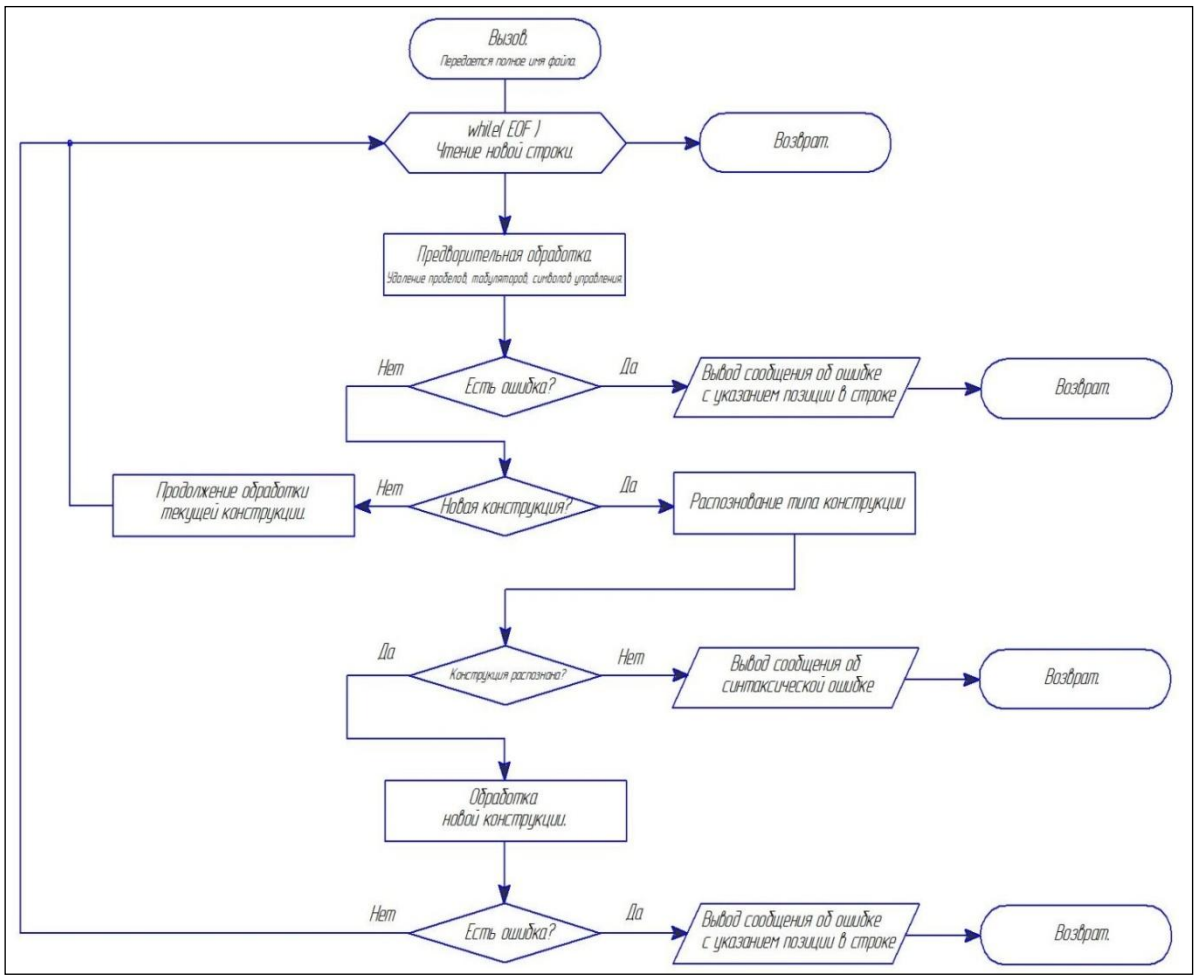
Пример:
MMPLL,1, 147.999332, -23.998496
MMPLL,2, 154.000701, -23.999946
MMPLL,3, 154.001025, -28.001546
MMPLL,4, 147.998983, -28.001469
8. Масштаб карты метры/пиксель.
Пример:
MM1B,170.352987

Каждая директива калибровочного файла offline–карты оформлена в строгом соответствии с синтаксическими правилами, являющимися уникальными для каждой директивы. При ручном наборе это обстоятельство вызывает немало затруднений, и может приводить к синтаксическим ошибкам.

В связи с этим на входе системы, использующей offline-карты, целесообразно использовать модуль синтаксического анализа калибровочного файла.

Полное описание структуры (директив) калибровочного файла позволили автору разработать алгоритм выявления директив и проверки синтаксиса.

Блок схема разработанного алгоритма представлена ниже.



Модуль синтаксического анализа калибровочного файла состоит из следующих основных блоков:

1. Блок ввода и предварительной обработки строк;
2. Блок вывода сообщений о синтаксических ошибках в калибровочном файле;
3. Регистратор перехода на новую конструкцию;
4. Блок распознавания типа синтаксической конструкции;
5. Блок синтаксического анализа конструкций.

Модулю синтаксического анализа калибровочного файла передается полное имя файла, и модуль начинает цикл поочередного просмотра его строк. После ввода очередной строки текстового калибровочного файла строка в исходном виде передается в блок сообщений об ошибках для того, чтобы формировать сообщения со строкой в её первоначальном виде, удобном для пользователя.

Блок вывода сообщений о синтаксических ошибках в калибровочном файле предназначен для более подробного сообщения пользователю о характере ошибки и месте её обнаружения.

В тех случаях, когда место обнаружения не определимо, сообщение относится ко всей строке в целом.

В регистраторе перехода на новую конструкцию решается задача определения новизны синтаксической конструкции в тех случаях, когда конструкция может быть представлена в виде нескольких строк. Если строка является продолжением конструкции, то регистратор отправляет программу на продолжение обработки текущей конструкции.

Блок распознавания типа синтаксической конструкции использует атрибуты синтаксической конструкции для выполнения операции распознавания типа. Эта операция необходима в связи с тем что, при разработке синтаксических правил калибровочных файлов пользователю было разрешено пропускать конструкции, если есть возможность использовать данные по умолчанию.

Основная функция анализа синтаксической конструкции на корректность производится в Блоке синтаксического анализа конструкций.

В случае обнаружения ошибок Блок распознавания типа синтаксической конструкции и Блок синтаксического анализа конструкций выводят сообщения в Блок вывода сообщений о синтаксических ошибках в калибровочном файле.

Пример анализа синтаксической структуры «MMPLL» калибровочного файла.

В структуре «MMPLL» ставятся в соответствие каждой граничной точке карты географические координаты. Пример такой структуры:

- MMPLL,1, 147.999332, -23.998496
- MMPLL,2, 154.000701, -23.999946
- MMPLL,3, 154.001025, -28.001546
- MMPLL,4, 147.998983, -28.001469

Предыдущие данные содержат количество строк в конструкции синтаксической структуры «MMPLL».

Эта величина записана в переменную n_MMPNUM

Код обработки конструкции MMPLL:

```
for( int n = 1; n <= n_MMPNUM; n++ )
{
Блок ввода и предварительной обработки строк
        fget_new_str( current_str, STRING_MAX, fp );
Регистратор перехода на новую конструкцию
        sprintf(Error_type,"MMPLL,%i,",n);
        if( !strstr(current_str,Error_type) )
        {
                sprintf(Error_type,"Пропущена строка MMPLL,%i,",n);
                goto m_ERROR;
        }
```

Блок распознавания типа синтаксической конструкции – не требуется, так как распознавание этой конструкции было произведено при анализе предыдущих строк.

Блок синтаксического анализа конструкции:

```
        int j,kz; j = 1; kz=0;
        for( ; ; j++ )if( *(current_str+j)==',' || *(current_str+j)=='\n' )break;
        if( *(current_str+j)==',' )kz++;
        for( j++; ; j++ )if( *(current_str+j)==',' || *(current_str+j)=='\n' )break;
        if( *(current_str+j)==',' )kz++;
        for( j++; ; j++ )
        {
        if( *(current_str+j)==',' || *(current_str+j)=='\n' )break;
        if( *(current_str+j)=='.' || *(current_str+j)=='-' || *(current_str+j)=='+' ||
                (*(current_str+j) >= '0' && *(current_str+j) <= '9'))continue;
        if( *(current_str+j)== 32 ||  *(current_str+j) == '\t' )continue;
```

```
                    sprintf(Error_type,"Синтаксическая ошибка в строке
                    MMPLL,%i,",n);goto m_ERROR;
                }
                if( *(current_str+j)==',' )kz++;
                if( kz != 3 )
                {
                    sprintf(Error_type,"Синтаксическая ошибка в строке
                    MMPLL,%i,",n);goto m_ERROR;
                }
                for( j++; ; j++ )
                {
                if( *(current_str+j)=='\n' || *(current_str+j)=='\r' )break;
                if( *(current_str+j)=='.' || *(current_str+j)=='-' || *(current_str+j)=='+'  ||
    (*(current_str+j) >= '0' && *(current_str+j) <= '9'))continue;
                if( *(current_str+j)== 32 ||  *(current_str+j) == '\t' )continue;
                sprintf(Error_type,"Синтаксическая ошибка в строке
MMPLL,%i,",n);goto m_ERROR;
                }
            }
```

Разработанный синтаксический анализатор соответствует всем, предъявленным к нему требованиям: использует минимальные ресурсы процессора, анализирует широкий набор калибровочных файлов, подробно сообщает пользователю программы об обнаруженных ошибках или опечатках, может модернизироваться с целью его дальнейшего развития.

Литература

1. Фостер Д. Автоматический синтаксический анализ. — М.: «Мир», 1975.— 71 с.
2. Страуструп Б. Программирование: принципы и практика использования C++.— М.: ИД «Вильямс», 2011. — 1248 с.
3. Саттер Г. Решение сложных задач на C++. — М.: «Вильямс», 2002. — 400 с.
4. Гамма Э. и др. Приемы объектно-ориентированного программирования. Паттерны проектирования. — СПб.: «Питер», 2007.— 344 с.