inter.noise
HAMBURG 2016

Development of a GIS based software for real time noise maps update

Andrea Cerniglia[1]

[1] ACCON GmbH, Germany

ABSTRACT

The paper describes the methods to obtain the global map of the investigated area, as a combination of the pre-computed data sets with the data coming from the monitoring stations. More in details, the paper will illustrate the basic platform aimed at presenting noise maps to the citizens on the Dynamap web site.

Keywords: Dynamic Noise Maps, Dynamap I-INCE Classification of Subjects Number(s): 76.9

1. INTRODUCTION

The LIFE-DYNAMAP project (Dynamic Acoustic Mapping - Development of low cost sensors networks for real time noise mapping) aims at developing an automatic dynamic noise mapping system able to detect and represent in real time the acoustic impact that is due to road infrastructures [1]. The purpose of the project is the European Directive 2002/49/EC for the assessment and management of environmental noise (END), referring to the need of updating noise maps every five years, as stated in the END [2].

To ease the update of noise maps and reduce their economic impact, DYNAMAP aims at building an automatic and integrated system for data acquisition and processing, able to detect and report in real time the acoustic impact that is due to infrastructure noise sources. The system consists of low cost sensors, which measure the sound pressure levels that are emitted by the noise sources present in the area, which has to be mapped, and of a software tool based on a GIS platform that performs real time noise maps updating.

In DYNAMAP, a monitoring system is under development [3] to check and test the feasibility of updating noise maps in real time along a major road (the Big Ring outside Rome - Italy) and in the agglomeration of Milan (Italy).

The Task B4.1 [4] of the Project (Development of a GIS based software for real time noise maps update) is related to the investigation on the computational hardware capabilities combined with different operating systems and tools to identify the best compromise between costs and efficiency of the needed configuration. The paper describes the principle of operation of the software routines for to collect, store and manage acquired data for the implementation of dynamic noise maps.

2. PRINCIPLE OF OPERATION

As described in the project proposal, the Dynamap system consists of several low-cost monitoring stations that collect and send data continuously to a web server through the usege of the Internet. In fact, the computer is not just a simple web server: it is a more complex machine that perform several tasks simultaneously, such as collecting and storing data, scaling and summing maps, presenting

[1] andrea.cernigia@accon.it

results on the web. In order to manage all these operations, it is thus necessary to store acquired data somewhere, recall stored data and basic maps from a database, perform operation on maps according to the acquired data, store and display results. Figure 1 shows the structure of the system.

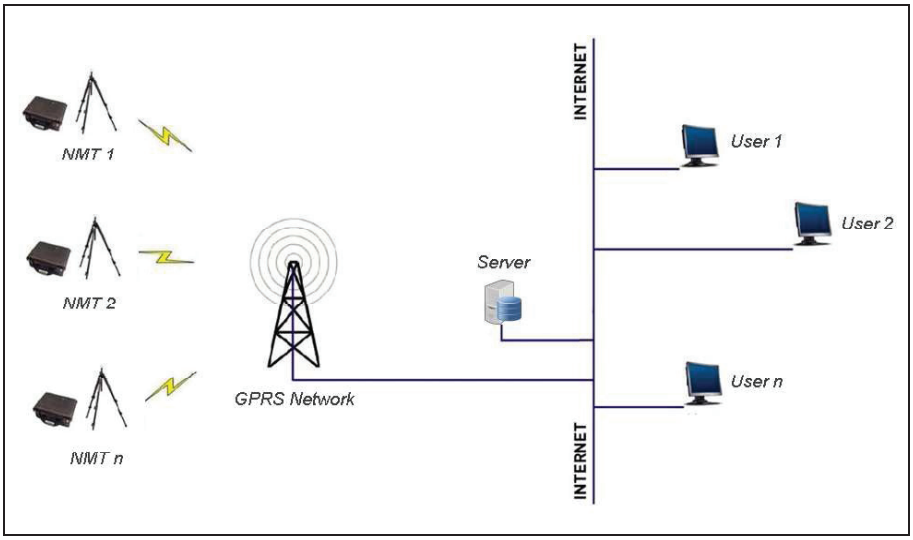

Figure 1 – Structure of the system

Figure 2 shows more in details the principle of operation of Dynamap system.

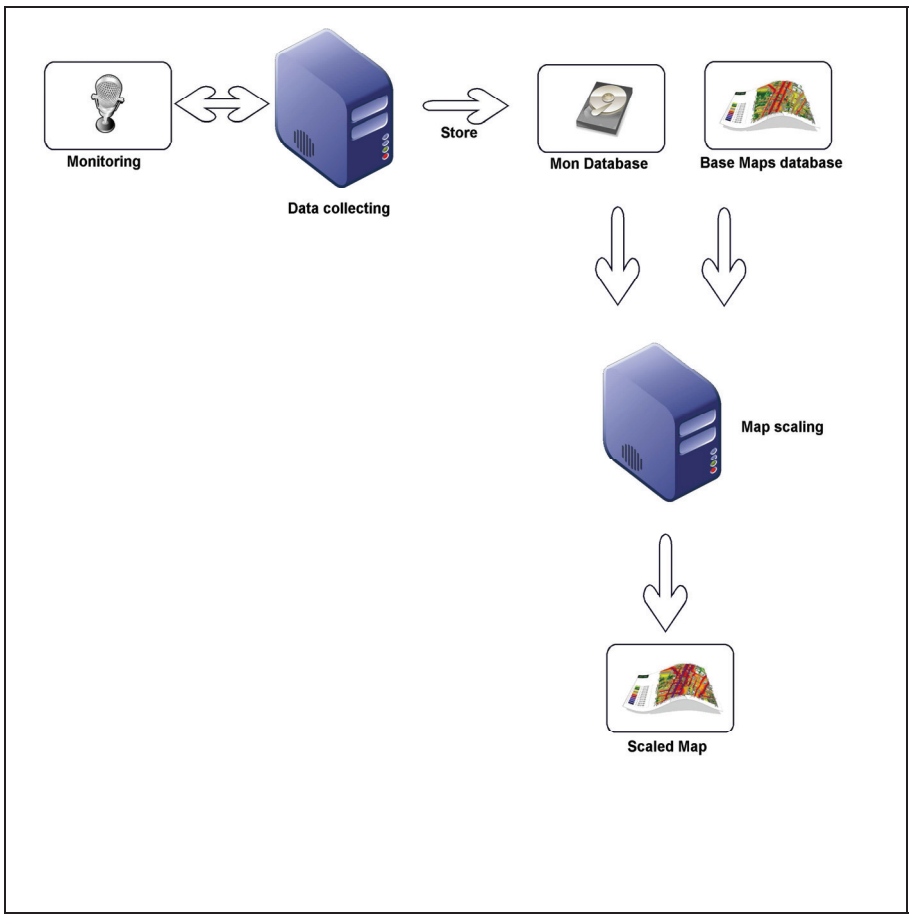

Figure 2 – Structure of the system

Communication between monitoring station and data collecting unit is not limited to the transmission of the acquired data but it is, in fact, a two-way communication, as shown in Figure 3.

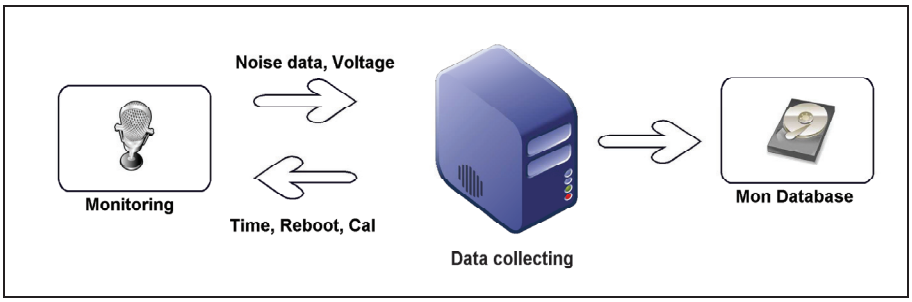

Figure 3 – Communication between monitoring station and Data collecting unit

This two-way communication is necessary in order to control each monitoring station and to send settings and commands to it, when necessary; for example, a crucial point of the system is related to the time synchronization between monitoring stations that should post acoustic measurement associated with a consistent timestamp tag: this task is automatically performed at regular time intervals by a message sent from control unit to each monitoring station. In the same way, the activation of a check signal can be triggered by the control unit to verify that the monitoring station is working in a proper way, as well as a reboot of a specific monitoring station can be forced by sending proper commands.

3. PLATFORM SELECTION

The main idea in the selection process for computational hardware and software was to define a cheap, open, scalable and reliable configuration for Dynamap system implementation. In order to do that, different kind of tests, involving different machines, operating systems and tools, were checked. In addition, also some stress tests were conducted to check the behavior of the configurations under possible critical situations, at the moment mainly focused on data acquisition and storage.

The above-mentioned tests proved that an entry-level 3 GHz machine is a good hardware choice for data collection and storage (25 measuring point, 1 LAeq per second) and for basic map scaling. Regarding for software, although most tested configurations worked without any problem, open source systems were preferred because they are free of charge and they can be deeply investigated to solve possible problems or to optimize critical parts of the processes.

Thus, after several tests, the selected configuration is the following:

Table 1 – Software configuration

Operating System	Linux (CentOs 7)
Web Server	Apache 2.4
Scripting Language	PHP 5.6
General Purpose dBase	MySQL 5.7
GIS dBase	PostgreeSQL 9.4 + PostGIS 2.11

4. DATABASES STRUCTURE

Regarding the databases, there are six separate of them in the Dynamap system. The choice of store information on separate databases is due to the fact that, for practical reasons, it is possible that, in its final implementation, databases will be located on different machines. Implemented databases are:

- Dynamap_Mon_Data
- Dynamap_Mon_Results
- Dynamap_Basic_Maps
- Dynamap_Maps_Results
- Dynamap_Access_Privileges
- Dynamap_Administration

Each database consists of several tables that store different kinds of data.

4.1 Dynamap_Mon_Data database

The Dynamap_Mon_Database has two tables. The first table, named 'location', stores information about each monitoring station. Table 2 are shows location table parameters.

Table 2 – Dynamap_Mon_Data database: location table

Field name	Type	Len	Description	Example
ID	INT	4	Primary key auto increment	0001
Station_ID	Varchar	6	Station ID mnemonic	0032a
Lat	Varchar	9	Latitude in degrees and dec.minutes (DDDMMdddC)	04154133N for 41°54.133'N
Lon	Varchar	9	Longit. in degrees and dec. minutes (DDDMMdddC)	01227408E for 12°27.408'E
Height	Varchar	5	Signed heght above sea level x 10	+00772 for 77.2 meters
Address	Varchar	50	Address of the station	Vatican City - 1, St Peter square
Timezone	Varchar	3	UNIX Timezone	+01
Conf	Varchar	10	Configuration of monitoring station	BW001AF___
Firmware	Varchar	10	Firmware version	1.0.0
Serial	Varchar	10	Instrument serial number	0123
Calib	Varchar	4	Signed calibration	-262 for -26.2 dBV
Battery	INT	3	Battery capacity	015 for 15 Ah
Comsump	INT	3	System absorbion	110 for 110 mA
Inst_Date	INT	11	UNIX timestamp	1439632800 for 12:00:00 15/08/2015
Cal_Date	INT	11	UNIX timestamp	1439632800 for 12:00:00 15/08/2015
Note	Varchar	100	Note	Solar panel power supply
Conn_Type	INT	1	0 = GPRS, 1=Network, 2=WIfi, ...	0
Conn_Encr	INT	1	Connection encryption (only for WiFi)	0
Conn_ID	Varchar	16	Phone number or network/wifi name	+393390123456789
Conn_User	Varchar	10	Connection username	Dynamap
Conn_Pass	Varchar	10	Connection password	DonaldDuck
Collected	INT	10	Number of collected samples	00008640000
Reboot	Boolean	-	Flag to force reboot (0=norm, 1= reboot)	0
Autocal	Boolean	-	Flag to force autocalibration (0=norm, 1= cal)	0
Status	Char	1	Status of monitoring stat. (0=Ok, 1,2,3 =type of fail)	0

Most of the parameters listed in table 2 are self-explanatory, except for Timezone, Battery, Absorbtion, Reboot, Autocal, Status. Below there is an brief explanation of the other parameters.

- *Timezone:* this parameter corresponds to the difference between local time and UTC. It is necessary for a correct time management, also according to winter/summer time.
- *Battery and Comsump:* these parameters are related to power supply. The first parameter stores battery capacity, whereas the second is the monitoring station's power consumption. These parameters, together with a continuous voltage monitoring, allow to compute the residual autonomy of the system in case of external power supply failure (poor light intensity in case of solar panel, or problem with mains).
- *Reboot:* flag that can be used to reboot the monitoring station when necessary.
- *Autocal:* flag aimed to start an autocalibration procedure if needed and implemented.
- *Status:* variable that can be set both by the monitoring station and system to send alerts.

The second table in the database Dynamap_Mon_Database stores the noise acquired data as explained in table 3.

Table 3 – Dynamap_Mon_Data database: Values table

Field name	Type	Len	Description	Example
ID	INT	5	Primary key auto increment	00001
Time	INT	11	UNIX Timestamp of measured value	1439632800 for 12:00:00 15/08/2015
Delay	INT	2	Delay between Timestamp Arrival time and Time	01 for delay of 1 second
S	INT	2	Delay between data sent and ACK received	01 for ACK received in 1 sec
F00	INT	4	10 x dBA value in the form XXXY	0938 for 93.8 dBA
PS	INT	3	10 x power supply voltage in the form XXY	0126 for 12.6 Volt

In the table parameters ID, Time, F00 and PS are self-expalantory.

The reason why *delay* values are stored, is to collect information about the variation of the amount of queued data on the monitoring station memory over the day (i.e. to detect critical period of the day for data transmission).

The reason why *S* value is store, is to collect information about 'how efficient' is the data transmission over the day (i.e. to detect the time needed to reach server).

The two above-mentioned information seem to be similar at first glance, but they carry different content.

To be more clear, Figure 4 shows the time history over 8 hours of the queued values expressed as a percentage of monitoring station internal memory usage (*Delay*). The diagram shows that, at the 10th minute of every hour (and in some other points), a brief stop in data transmission occurred.

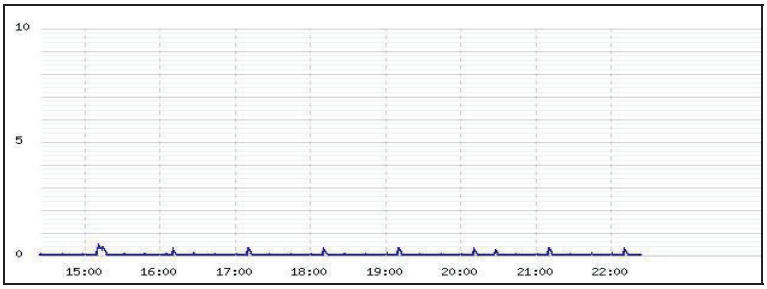

Figure 4 – delay between Timestamp arrival and time

Figure 5 shows the time history over the same 8 hours as above, but it is related to the delay between data transmission from the monitoring station and received ACK from the server, expressed as a percentage of 'Goodness' where 100% means 0 seconds, 80% means 2 seconds and 20% means 8 seconds. The comparison between the two graphics shows that a delay in ACK receiving does not necessary mean data queuing.

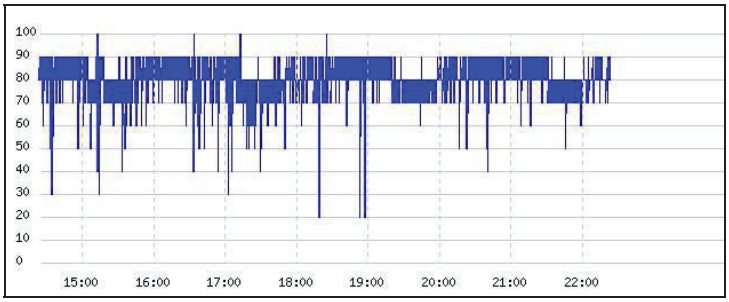

Figure 5 – delay between Data Ssent and ACK received

4.2 Dynamap_Mon_Results database

In noise monitoring, hourly levels are data of particular interest. For this reason, the system automatically computes hourly values. Thus, for each station and for each hour, these values are computed and stored in a separate database named Dynamap_Mon_Results. The database contains one table per day for each monitoring station. Dynamap_Mon_Results tables have their name in the form 'nrXXXXYYMMDD', where 'nr' is the common prefix for (n)oise(r)esults, 'XXXX' is the station number and 'YYMMDD' stands for year, month and day. Table 4 shows the structure of these tables.

Table 4 – Dynamap_Mon_Results database: Values table

Field name	Type	Len	Description	Example
ID	INT	5	Primary key auto increment	00001
H00L	INT	4	10 x dBA value in the form XXXY for hour 00	0938 for 93.8 dBA
H00S	INT	4	Number of samples in the hour for hour 00	3600
..				
..				
H23L	INT	4	10 x dBA value in the form XXXY for hour 23	0938 for 93.8 dBA
H23S	INT	4	Number of samples in the hour for hour 23	3600

4.3 Dynamap_Basic_Maps database

In order to publish dynamic noise maps, the system should scale and sum relevant basic noise maps. These maps are stored in a specific database as value of single X,Y coordinate. Dynamap_Basic_Maps database contains this information in a table according to the structure described in Table 5. The reason why there are the four pointers is for interpolation purposes.

Table 5 – Dynamap_Basic_Maps database: Values table

Field name	Type	Len	Description	Example
ID	INT	8	Primary key auto increment	0001
Point_ID	Varchar	6	Point ID mnemonic	0099b
Lat	Varchar	9	Latitude in degrees and dec.minutes (DDDMMdddC)	04154133N for 41°54.133'N
Lon	Varchar	9	Longit. in degrees and dec. minutes (DDDMMdddC)	01227408E for 12°27.408'E
Height	Varchar	5	Signed heght above sea level x 10	+00772 for 77.2 meters
Value	INT	4	10 x dBA value in the form XXXY	0938 for 93.8 dBA
Ptr_A	INT	8	Pointer to adjacent related point A ID	00000109
Ptr_B	INT	8	Pointer to adjacent related point B ID	00000111
Ptr_C	INT	8	Pointer to adjacent related point C ID	00000010
Ptr_D	INT	8	Pointer to adjacent related point D ID	00000210

4.4 Dynamap_Maps_Results database

The Dynamap_Maps_Result database contains, for each time of calculation, the noise level of each point after the scaling and sum procedure in a table following the structure shown in table 6.

Table 6 – Dynamap_Maps_result database: Values table

Field name	Type	Len	Description	Example
ID	INT	8	Primary key auto increment	0001
Timestamp	INT	11	UNIX Timestamp of scaled and summed point	1439632800 for 12:00:00 15/08/2015
Point_ID	Varchar	6	Point ID mnemonic	0099b
Lat	Varchar	9	Latitude in degrees and dec.minutes (DDDMMdddC)	04154133N for 41°54.133'N
Lon	Varchar	9	Longit. in degrees and dec. minutes (DDDMMdddC)	01227408E for 12°27.408'E
Height	Varchar	5	Signed heght above sea level x 10	+00772 for 77.2 meters
Value	INT	4	10 x dBA value in the form XXXY	0938 for 93.8 dBA

Once new maps are stored, the map display tools, at the moment still under development, will permit to show results to citizens on the Dynamap web site, with different screen layouts and different privileges.

4.5 Other databases

The structures of the other databases, Dynamap_Access_Privileges and Dynamap_Administration are currently under development (Action B4.2 of Dynamap project).

5. CONCLUSIONS

The test procedures that was implemented to check the preliminary version of the system proved that the selected configuration was capable to collect data from 25 monitoring station (1 L_{Aeq} per second each), perform the scale and sum process of six pre-computed basic noise map and present some basic result to 50 simultaneously users without any problem. The implemented databases structure revealed a robust data organization that can be easily interfaced by external software routines for advance noise maps presentation, currently still under development.

REFERENCES

1. DYNAMAP – Development of low cost sensor networks for real time noise mapping, LIFE Projects 2013, Environment Policy and Governace – July 2014
2. EU Directive (2002), Directive 2002/49/EC of the European parliament and the Council of 25 June 2002 relating to the assessment and management of environmental noise, Official Journal of the European Communities, L189/12, July 2002
3. L.Necini, Francesc Alias, Xavier Sevillano, Dynamap: a system with low cost hardware and artificial intelligence to compute reaal-time noise maps, TecniAcustica, Murcia 2014
4. Andrea Cerniglia, Dynamap Report B4.1 - Dynamap GIS based software for real time noise maps update