Automatic Generation of Sensor Queries in a WSN for Environmental Monitoring[1]

F. A. Schreiber
Dipartimento di Elettronica e Informazione
Politecnico di Milano
schreibe@elet.polimi.it

ABSTRACT
The design of a WSN for environmental data monitoring is a largely ad-hoc human process. In this paper, we propose the automatic generation of queries for sensor data extraction, based on the collection of a number of parameters concerning the physical phenomenon to be controlled, the relevant physical variables, the types of sensors to be deployed and their allocation, the data collection frequencies, and other features.

KEYWORDS
Automatic query generation, environmental data monitoring, sensor data extraction

INTRODUCTION

PROMETEO is a large applied research project which aims at giving technological support to the problems encountered within the forecasting, prevention, avoidance, and management of emergency situations by the Civil Defence Departments, from sensors deployment up to the Situation Room decision procedures.

Data management is one of the main issues in the project; data are collected by different kinds of intelligent sensors -- both wire- and wireless-connected to one or more sensor networks --, and must be filtered and processed in several ways in order to be available, at different levels of detail, to the application programs. A general view of the system is shown in Figure 1, where two possible data management philosophies are depicted: in the first approach (upper branch), the collected data are processed by the network nodes by means of ad-hoc languages (e.g. Tiny-DB) (Madden, Franklin, Hellerstein and Hong, 2005) and permanently stored in a database; data are later queried by the application programs by means of standard database languages (e.g., SQL) and possibly processed for building an associate Data Warehouse for emergency planning management purposes; in the second approach (lower branch), the network nodes themselves are responsible for storing raw or value-added data, and application programs can directly query them by means of the same ad-hoc languages mentioned above (Bonnet, Gehrke and Seshadri, 2001).

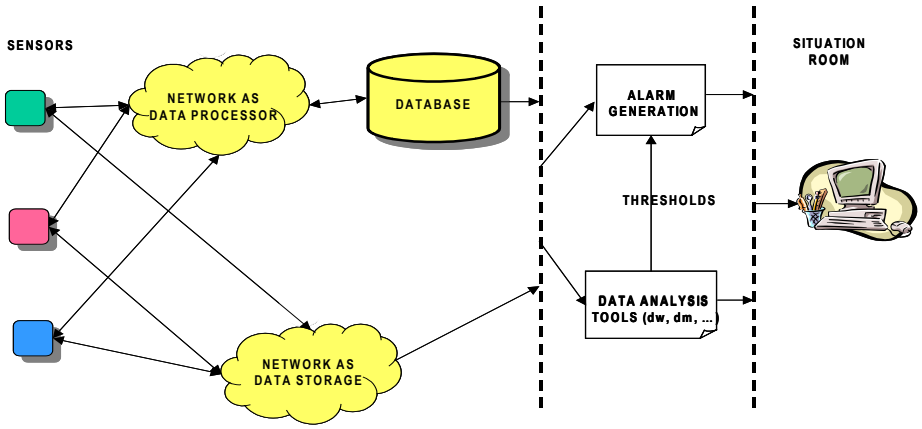

Figure 1 General system architecture

[1] This research is partially supported by the Prometeo project of Politecnico di Milano and by MIUR FIRB project ART DECO.

When dealing with wireless sensors, the designer faces two main technological problems:

- *memory amount*: wireless sensor boards are endowed with a small amount of local storage (in the order of 10^2 KB up to some10th MB) which can only host few, possibly compacted, measurement data for a limited time span, hence the need to periodically download them, via the radio link, to a larger storage device;
- *power amount*: the life of on-board batteries is limited and transmission is the most power consuming function; therefore data transmission must be kept to a minimum in order to prolong the sensor's operational life.

The idea of hiding the user the many technical details, needed to configure a WSN, by means of component libraries has been applied in several contexts (e.g., Chu, Lin, Linare, Nguyen and Hellerstein, 2006). In this paper we examine the problems encountered in data design for a geophysical and meteorological monitoring WSN, and propose a general, parametric data structure which houses both the (meta)data used for configuring the WSN and its measure conditions, and the data actually collected by the sensors. In particular, the meta-data can be used by the designer at network design/deployment time in order to define such parameters as, e.g., the number of devices to be deployed per sq meter, based on the information about the phenomenon to be monitored, the environmental conditions, etc. On the other hand, the meta-data can be used to set the data collection frequency of some critical variables at operation time. Dynamically setting data collection frequencies is particularly useful in WSN in order to optimise the power consuming transmission operations while limiting storage occupation. Application requirements as to raw data processing (set operations, statistical indexes evaluation, data compression and/or summarization) are also considered, in order to automatically produce a library of standard parametric queries, which can be instantiated to collect the sensors data and store them into the system Database.

Next section introduces the application environment and the Database schemata, in the third one the meta-data for the configuration tool are introduced and modelled; the last three sections are devoted to the definition and the automatic generation of the sensor queries.

GEOPHYSICAL VARIABLE MONITORING AND DATA PROCESSING

Geophysical monitoring is the set of activities allowing to characterize and control a geophysical failure (e.g., fractures in a landslide body, deep movements, sliding surfaces, etc.) all along its lifespan. Measurement instruments can be either managed by a human operator, or they can work autonomously, collecting and transmitting data to base stations. Geophysical information can be supplemented by topographic surveys, both terrestrial and GPS (Global Positioning System) based, and by a set of hydro-meteorological data (rainfall, snowfall, wind speed and direction, air temperature and humidity, sun radiation, flow, etc.) coming from large measurement networks. Altogether, these data allow to build models for the environmental behaviour of each landslide and for forecasting possible avalanches and floods (ARPA, 2003; ARPA, 2004).

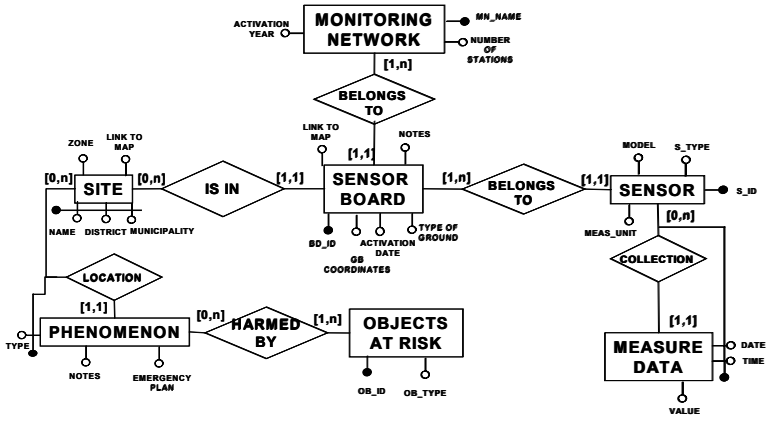

Figure 2. The Database conceptual schema

ARPA Lombardia, the Agency for the environment of the Lombardia region, organized measurement data in a database, a simplified conceptual schema of which is shown in the Entity-Relationship diagram of Figure 2, and whose logical schema is the following:

```
Measure(s_id, date, time, value)
```

```
Sensor(sensor_id, sensor_type, sensor_model,  board_id, unit)

Sensor_board(bd_id, activ_date, ground_type, GB_coord, link_to_map, notes,
site_name, site_distr, site_municip, mn_name)

monit_net(mn_name, activ_year, n_of_stations)

Site(name, distr, municip, zone, link_to_map)

Phenomenon(name, distr, municip, type, emergency_plan, notes)

Object_risk(ob_id, ob_type)

Harmed(site_name, site_distr, site_municip, phen_type, ob_id)
```

On this schema several views are defined, with different data aggregation criteria, to support the analysis procedures: min, max and average values are daily stored for snow and hydrometric heights and for air temperature; cumulated day values are stored for rainfall, the number of used data points is also stored to detect possible data absence intervals or failures in the sensors.

To give an idea of the database size consider that, at the end of 2004, it stored more or less, 907.000 values for *snow height*, 2.518.000 values for *hydrometric height*, 5.230.000 values for *air temperature*, 5.673.000 values for *rainfall*, and so on for a total of $30*10^6$ measure data occupying approximately 1 Gbyte of disk space (ARPA, 2003; ARPA, 2004).

THE WSN DATA COLLECTION SPECIFICATION TOOL

As we mentioned in the introduction, when using wireless sensors, a careful trade-off must be made for optimizing local storage usage vs. data transmission in order to save power while keeping all the information needed by application programs. This amounts to collecting no less, but also no more data values than needed, by suitably tuning the data collection frequency for each sensor in the network; the collected data could also be compressed and summarized directly on the sensor board, obtaining the double advantage of saving memory space -- so delaying the need to free it by transmission -- and shortening transmission time (Tulone and Madden, 2006). Adapting data collection frequencies to the instantaneous environmental conditions adds a third adaptability policy to those based on network contention and power consumption considered in Madden et al., 2005.

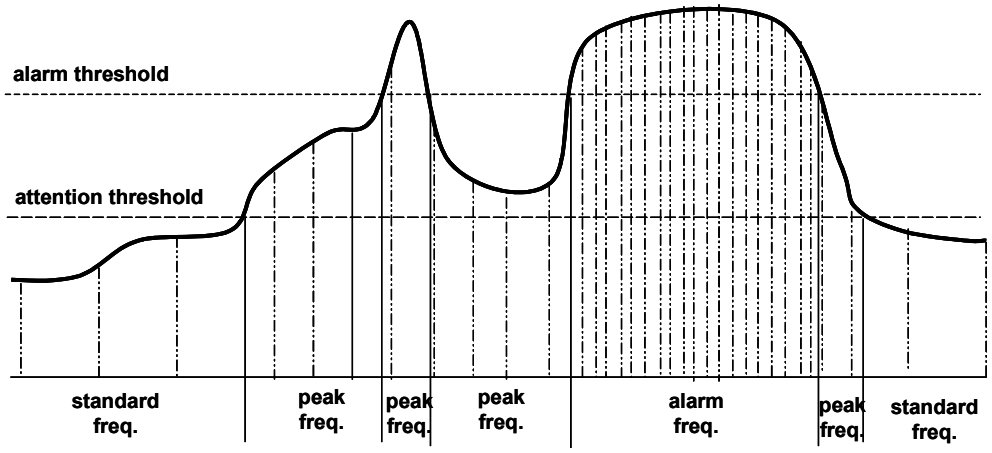

Figure 3. Thresholds and data collection frequency in an environmental variable

To clarify this point, let us consider sensors monitoring the hydrometric height of a water basin. In normal steady state conditions, it is enough to take a measure once a day; however, when rainfalls are heavy, for instance above a given ``attention threshold", it might be wiser to monitor it every 6 hours, and, if an ``alarm threshold" is reached for two consecutive measures, the measure frequency should be increased to one every 30 minutes and an alarm sound is to be issued (Figure 3).

Note that the data stored in the database of Figure 2 represent information about the sensor network as it has been deployed, together with the environment where it has been installed. However, more information is needed when the

task of sensor deployment has to be accomplished, since the diverse nature of the different phenomena that have to be monitored calls for different collection policies; these, in turn, are translated into appropriate deployment strategies, along with appropriate application programs that reside on board of each sensor. Indeed, the same sensor type can be used to monitor different kinds of geophysical failures (e.g., accelerometers can be used, with different modalities, for monitoring landslides, as well as avalanches or earthquakes) which exhibit very different time constants and spatial patterns, thus measure set-points must be established for each sensor and for each phenomenon. Therefore the database of Figure 2 must be supplemented with auxiliary *meta-data*, i.e. *configuration information*, in order to properly set the measurement system up, and a feedback path must be established between sensor data and measure set-points.

The meta-data needed to set a WSN up must answer the following questions:

- which are the physical properties characterizing each phenomenon;
- which sensor types are useful for monitoring a phenomenon;
- which are the time constants and the spatial patterns characterizing each phenomenon;
- which are the data collection frequencies and the thresholds set-points for each sensor.

These questions can be answered by the following logical schema for the configuration meta-data, other meta-data can be added if needed:

`C_phenomenon (phen type, sensor type, property)`

`C_phenomenon-constants (phen type, time_const, space_const, ΔX, ΔY, ΔZ, error)`

`C_sensor (sensor id}, sensor_type, zone, GB_coord)`

`C_sensor-constants (phen type, sensor type, std_freq, peak_freq, alarm_freq, attention_thresh, alarm_thresh)`

A few words to explain the meaning of some non-intuitive attributes:

- `property`: for each phenomenon and for each sensor type, this attribute expresses the peculiar characteristics that the physical variable, measured by that type of sensor, assumes for that phenomenon. Possible values are: *relevance* (i.e., how much the parameter influences the phenomenon); *homogeneity* (i.e., data values collected by nearby sensors should exhibit low variance, and possibly could be averaged); *peaks* (i.e., the variable might undergo sudden variations, which are to be considered as meaningful for the specific monitoring task and thus are not to be missed); *attention_threshold* (i.e., above this value the variable could exhibit peaks); *alarm_threshold* (i.e., above this value an alarm should be issued); *unpredictable* (i.e., the failure can be occasional and nearly instantaneous, an alarm should be issued in any case); *precision* (i.e., the precision to be used in collecting data, it is determined by the phenomenon *and* by the sensor type).
- `time_const`: the time constant typical of the specific phenomenon. For example, if landslides are being monitored, the time constant is in the order of seconds, much smaller than in the case of land subsidence, where it is in the order of days or months.
- `space_const`: the spatial range on which the variable can be considered *homogeneous*. This is a critical concept which heavily depends on the type of measure. While it can be useful in compressing data (e.g., by averaging them on the *homogeneity area*) it can not be applied where the physical conditions require for a detailed survey.
- Δ`{ }`: coordinate range of the area interested by the failure.
- `error`: the admissible error from the nominal sensor position.
- `zone`: the topographical name of the area interested by the failure - where the sensor board at geographical coordinates `GB_coord` - is placed.

Incidentally, it can be noticed that some data and meta-data coincide (e.g., `sensor_id`, `sensor_type`, …); actually, data and meta-data can be looked at as different views on the same database and they are not to be

duplicated, however we are not going to deepen this topic here. Thus, queries are issued on this Database not only to collect the measured values of the geophysical variables and possibly to obtain set and aggregate values from them, but also to provide the feedback on the measurements set-points, as we are going to see in the next section.

QUERY DEFINITION

The large amount of possible queries in complex environmental monitoring systems, and the need to dynamically configure the system in order to adapt it to changes in the network topology and in the measurement conditions - even for the same phenomenon -, claim for automatic generation of a library of parametric query classes that can be further specialized by instantiating parameters as needed.

Therefore we defined and built a tool, described in the next section, which, starting from a simple standard query skeleton, allows to specialize the query on the basis of the phenomenon, the physical variable (sensor type), and the instantaneous external conditions. Though in the Prometeo project several data querying interfaces are used, here, for the sake of uniformity, we use TinyDB - a language where queries are basically written in SQL, with some restrictions and some new constructs - as the running example query language (Madden et al., 2005). since, at the moment, it is the most known declarative data querying interface for sensor networks. The tool starts from the following basic query, which is progressively modified according to the phenomenon, and to the sensor it is installed to:

```
SELECT expression
FROM sensors                                                    (1)
SAMPLE PERIOD frequency
```

Where `sensors` is a virtual table partitioned across all the devices in the network. Referring to our example Database schema, `sensors` is a projection over the measurement attributes of the join of `sensor_board`, `sensor`, and `measure` tables. Additional patterns are added, based on the parameter values collected in the above tables. Table 1 contains some of the query patterns that are inserted, based on the characteristics of the physical variables and of the phenomena. Even if patterns in Table 1 refer to environmental monitoring, the method is quite general and can be generalized to whatever application domain (Yao and Gehrke, 2003).

property	time aggregate	space aggregate
homogeneity		`SELECT AVG(expr),MAX(expr),MIN(expr) WHERE {position | space constant | condition} GROUP BY {area | condition}`
attention threshold	`ON EVENT attention threshold overcome SAMPLE PERIOD peak frequency STOP ON EVENT below attention threshold`	
alarm threshold	`ON EVENT alarm threshold overcome SAMPLE PERIOD alarm frequency STOP ON EVENT below alarm threshold`	
alarm issue	`ON EVENT alarm threshold overcome OUTPUT ACTION threshold-alarm`	
unpredictability	`ON EVENT external event OUTPUT ACTION threshold-alarm`	

Table 1. Query patterns

QUERY GENERATION

This section briefly describes the query generation tool (Figure 4) which, starting from the standard query skeleton, allows to specialize the query on the basis of the phenomenon, the physical variable (sensor type), and the instantaneous external conditions.

The first activity consists in choosing, within the list of phenomena, the phenomenon that has to be monitored by the query we are going to generate. This choice may be made by a human operator or automatically, by some other

application which interacts with the query generator. Subsequently, the table `C_phenomenon` – Table 2 - is queried, in order to find which sensors are relevant, and thus should be used, to monitor the considered phenomenon; for example, landslide monitoring claims for the use of the following sensors: vibration (accelerometer), displacement (strain gauge), rainfall (pluviometer), air humidity (hygrometer). For each sensor type, the same table contains the characteristics that the physical variable, measured by the sensor, assumes for the given phenomenon; thus Step 3 collects such characteristics, and generates a set of parametric queries for each sensor type, based on the criteria of Table 1. Moreover, Table 3 shows an instance of the relation `C_phenomenon-constants` for the flooding, landslide and avalanche phenomena; this information is used in defining sensors density and placement and for grouping them into the different topographical zones. Finally, in Step 4 the parameters are replaced with specific values which are collected from the configuration database for each sensor type, an example of which is shown in Table 4[2].

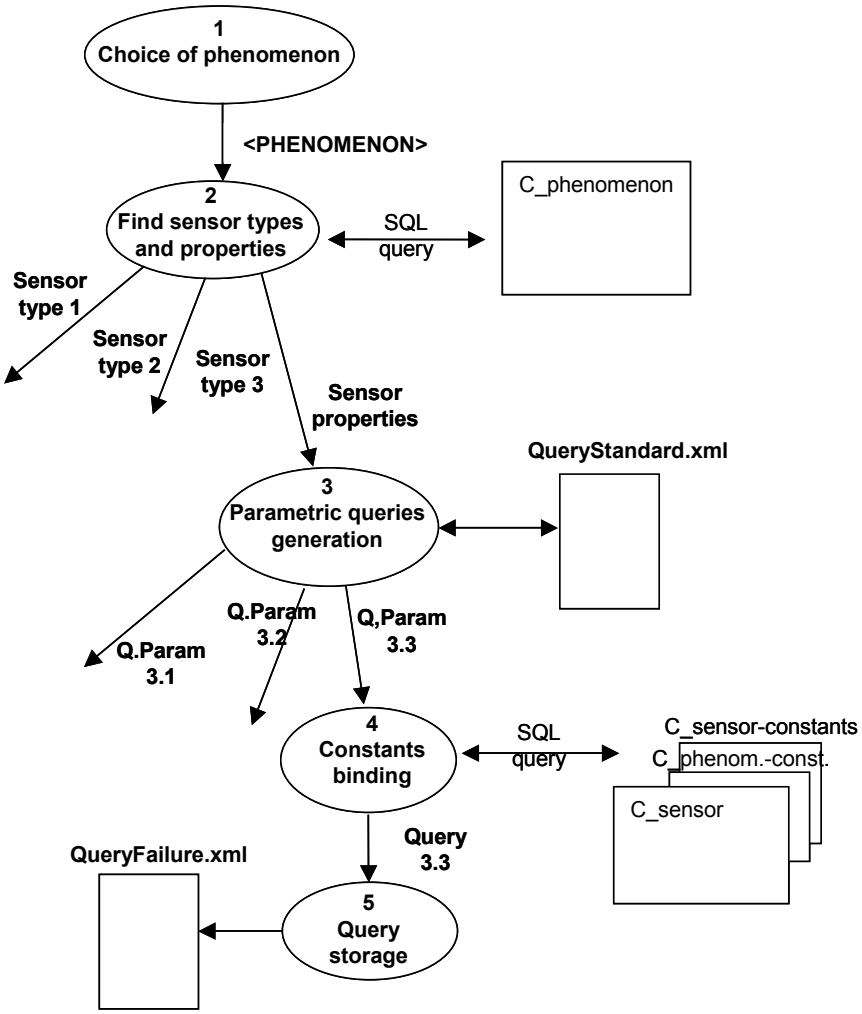

Figure 4. Phases of query generation

phenomenon type	sensor type	property
flood	pluviometer	homogeneous

[2] Figures in tables 3 and 4 are used only for exemplifying purposes and do not necessarily correspond to real data

flood	pluviometer	peak
flood	pluviometer	attention threshold
flood	pluviometer	alarm threshold
flood	flowmeter	none
landslide	accelerometer	unpredictable
landslide	pluviometer	attention threshold
landslide	pluviometer	alarm threshold
landslide	thermometer	homogeneous
landslide	hygrometer	alarm threshold
landslide	Strain gauge	unpredictable

Table 2. An instance of the table `C_Phenomenon`

phenomenon type	time constant	space constant	ΔX	ΔY	ΔZ	error
flood	1 hr	10^2 m	1 km	10 km	1 m	10 %
landslide	30 min	10^2 m	10^2 m	10^2 m	1 m	1 %
avalanche	10 min	10^2 m	10^2 m	10^2 m	1 m	5 %

Table 3. An instance of the table `C_Phenomenon-constants`

phenomenon type	sensor type	standard freq.	peak freq	alarm freq	attention thresh.	alarm thresh.
flood	pluviometer	6 hr	2 hr	30 min	50 mm	75 mm
landslide	pluviometer	10 hr	5 hr	1 hr	50 mm	75 mm
landslide	thermometer	2 hr	1 hr	30 min	35 C°	40 C°

Table 4. An instance of the table `C_sensors-constants`

The query skeleton `QueryStandard.xml` is an XML file, whose structure is defined by a DTD, and the generated queries are stored in `QueryFailure.xml`, an XML file as well. As an example, the kernel reference query (1), from which all the other queries are produced, can be generated by the following XML file:

```
<ROOT>
        <SELECT>
           <CODE> SELECT </CODE>
           <VAR param="sensor_type" >< /VAR>
        </SELECT >
        <FROM>
           <CODE>FROM sensors </CODE>
        </ FROM>
        <SAMPLE_PERIOD>
            <CODE>SAMPLE PERIOD< /CODE>
            <PARAM p1="O">frequency< /PARAM>
        </SAMPLE\_PERIOD>
< /ROOT>
```

The <VAR> tag means that an input value shall be given to the specified parameter; the <PARAM> tag means that a value shall be given to a parameter, on the basis of the specified parameter attribute.

Inputs to the procedure are the properties of the physical entities from `C_phenomenon` and the standard query skeleton from the `QueryStandard.xml` file. Properties are considered in a fixed order (homogeneity, attention_threshold, peaks, alarm_threshold, unpredictability) and the relevant tags are incrementally added or

removed as needed. Figure 5 shows the fundamentals of the algorithm which generates the `QueryFailure.xml` document in which the code of the actual queries is stored.

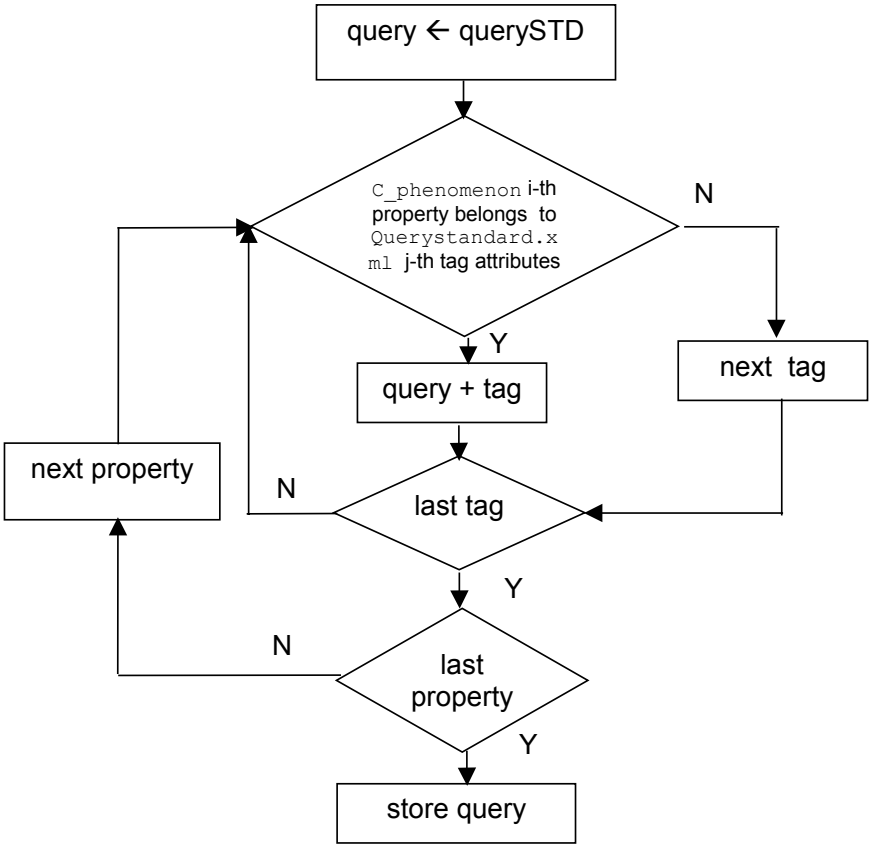

Figure 5. Algorithm for query generation

For instance, in case of homogeneous variables, we collect their average, minimum and maximum values, computed w.r.t. an area within the range of the space constants considered for the phenomenon. Thus, in Step 3, for each homogeneity area, a given sensor (the main sensor) is selected and the following `SELECT` clause replaces the standard one:

```
SELECT AVG(expr),MAX(expr),MIN(expr),zone
FROM sensors
WHERE sensor_type=$type
GROUP BY zone
HAVING ΔX ≤ $ΔX and ΔY ≤ $ΔY and ΔZ ≤ $ΔZ
SAMPLE PERIOD std_frequency
```

The `HAVING` clause is established by the spatial features of the phenomenon; for example, in case a landslide is being monitored, the space constant dictates the sensor density, which is also determined by the steepness of the slope where the sensors have been placed. Therefore the `HAVING` clause requires that the sensors taking part to the computation of the average, minimum, and maximum of the measured variable be within a certain distance (in terms of `x`, `y` and `z` coordinates) from the main sensor.

In Step 4 the actual query code is produced by instantiating the specific values for the variables and the parameters:

```
SELECT AVG(temperature),MAX(temperature),MIN(temperature),zone
FROM sensors
WHERE sensor_type = thermometer
GROUP BY zone
```

```
HAVING ΔX ≤ 300 and ΔY ≤ 300 and ΔZ ≤ 5
SAMPLE PERIOD 2
```

THE PROTOTYPE

A prototype parametric queries generator, which applies the techniques outlined in the preceding sections, has been built and tested. It is written in Java and it uses MySQL database management system (Boles, Cannone and Cotti Cottini, 2006; Motta and Oliveira, 2006). It allows the user to insert the metadata into the database tables and to generate the queries to be loaded on the sensors.

Figure 6 shows the interface for the latter function; the user can choose a phenomenon from the first menu and one of the sensors relevant to it from the second one, then – by clicking on the lower button – the query is generated and displayed in the window at right.

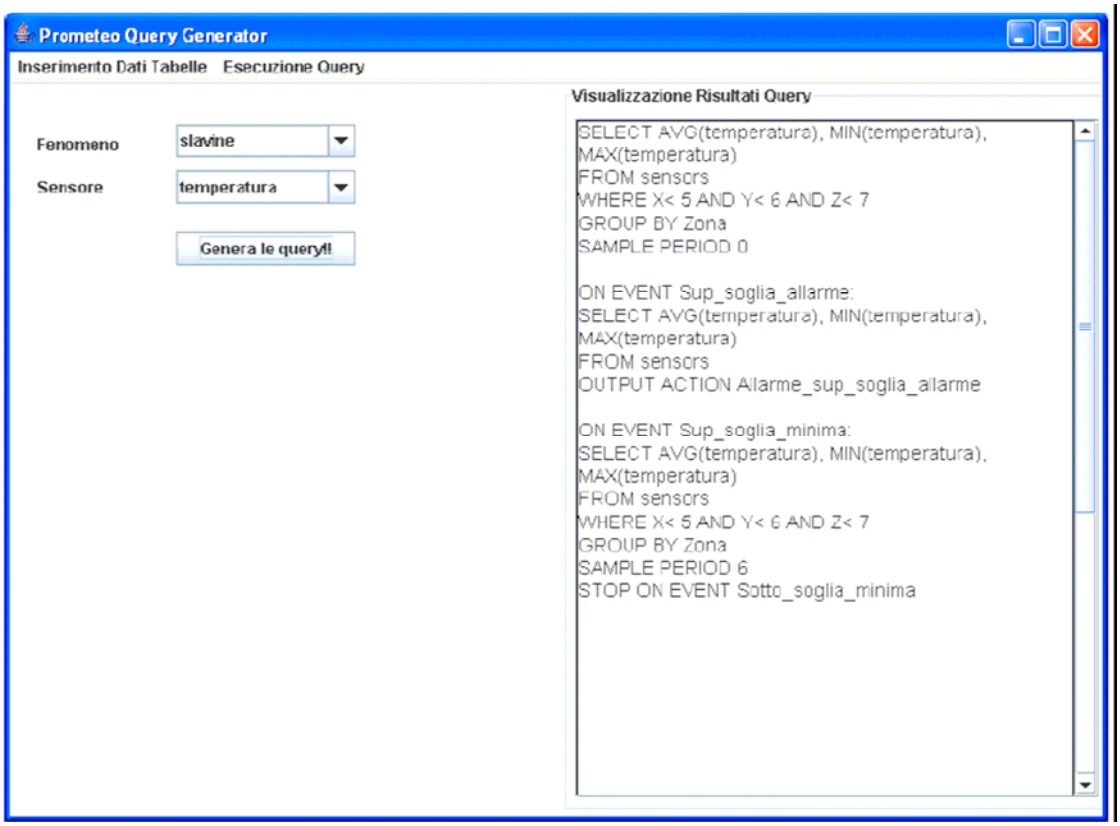

Figure 6. The prototype screen for generating the query code

CONCLUSIONS

A tool for the automatic generation of queries for wireless sensors to be used in an environmental monitoring network has been outlined. A table-driven approach allows also a naïve computer user an easy composition of the queries text as needed by the type of the monitoring action. At the moment, our proposal is mainly a methodological one; however, its feasibility has been validated by a simulated prototype. The development and the deployment of the system on the field is still to be planned by ARPA Lombardia.

ACKNOWLEDGMENTS

I thank Dr. Gregorio Mannucci, of ARPA Lombardia, for giving us an insight of geophysical measurement, Prof. Letizia Tanca for the many helpful discussions, and the B. Eng. students who developed the prototype.

REFERENCES

1. ARPA (2003) Lombardia: Centro Monitoraggio Geologico. Cd-rom: Dati idrometeorologici 1987–2003.

2. ARPA (2004) Lombardia: Centro Monitoraggio Geologico. Cd-rom: Attivita' 1987–2004.

3. Boles, M., Cannone, R. and Cotti Cottini,. F. (2006) Generatore di query per il monitoraggio di fenomeni ambientali mediante reti di sensori wireless. *B.Eng.Thesis - Politecnico di Milano - Dipartimento di Elettronica e Informazione*.

4. Bonnet, P., Gehrke, J. and Seshadri, P. (2001) Toward sensor database systems. *Proc. 2nd Int. Conf. on Mobile Data Management (MDM'01), LNCS 1987*, 3–14.

5. Chu, D., Lin, K., Linares, A., Nguyen, G. and Hellerstein J.M. (2006) sdlib: A Sensor Network Data and Communications Library for Rapid and Robust Application Development. *Proc. IPSN'06, Nashville*, 1-9

6. Madden, S., Franklin, M. J., Hellerstein, J. M. and Hong, W. (2005) Tinydb: an acquisitional query processing system for sensor networks. *ACM-TODS*, 30, 1, 122–173.

7. Motta, T. and Oliveira, L. (2006) Linee guida per l'acquisizione dati nel progetto prometeo. *B.Eng.Thesis - Politecnico di Milano – Dipartimento di Elettronica e Informazione*.

8. Tulone, D. and Madden, S. (2006) Paq: Time series forecasting for approximate query answering in sensor networks. *Proc. EWSN 2006, LNCS 3868*, 21–37.

9. Yao, Y. and Gehrke, J.(2003) Query processing for sensor networks. *Proc. CIDR Conference, Asilomar*.