Development of reusable software components for monitoring data management, visualization and analysis

Daniele Inaudi
SMARTEC SA, Switzerland

ABSTRACT

The permanent or regular monitoring of structures with sensors of any type can generate a consistent volume of data. Furthermore, it is often necessary to store additional information that is useful for the analysis of the measurements. This data should be comprehensible even after tens of years. Our experience has shown that the use of relational database structures can greatly simplify the handling of this large data-flow. With an appropriate data structure, the measurement data and other related information on the monitoring network, the structure and its environment can be organized in a single file that will follow the structure's life in the years.

The standardization of a database structure for storing monitoring data also allows the development of re-usable components for data acquisition, data analysis and representation. The use of relational database structures greatly simplifies the quality management and can help in the certification of monitoring systems.

This contribution presents a new open and free standard for database structures aiming to the archival of long-term monitoring data (SDB standard). The implementation of this standard in data acquisition, analysis and representation software modules will also be described.

Keywords: Structural monitoring, Database

1 INTRODUCTION

The installation of a permanent monitoring system in a structure brings with it the necessity of managing a large volume of data. Even in a structure with only a few tens of sensors that are measured every hour, the number of data points that are generated can become so large that a manual analysis becomes tedious and very time-consuming. Furthermore, the data acquisition phase is only the first step in a chain of operations that ultimately lead to high-level information useful for the structure’s owner. These analysis steps are more-and-more based on the use of software tools that require access to the raw data. This leads to the need for a structured data-storage system that can be accessed by the data acquisition software (DAQ) and by the data representation and analysis packages (DRA).

The natural choice for storing data is of course a database. In the last 5 years we have developed and deployed a database structure that proved its usefulness and reliability in more than 100 applications [1]. This relational database is called SDB (for Structural Data Base) and allows the storage of data from many different types of structures. Furthermore, the structure was designed to be open, so that it is possible to add fields and tables that might be needed for a particular sensor system.

Since many research groups and companies worldwide are currently developing DAQ and DRA applications, we feel that a standard core database structure could help making these tools more inter-operable. For this reason we propose the core SDB structure as an open standard for storing structural monitoring data. The SDB Standard is endorsed by SMARTEC SA, GeoDev SA, Omnisens SA (in Switzerland) and Infokom and GESO (in Germany).

2 DATABASE STRUCTURE

Given a project that involves many objects to be permanently or regularly monitored, the goal of the data model is to model for the scope of the given project all relevant entities and their relationships for assuring the long term management of the large volume of measured data, the short term and dynamic sensor-configuration and measurement-
planning and the need of system-specific data. The data model is therefore organized upon three layers: a measurement-specific layer, a configuration layer and a system-specific layer. A picture of the data model is represented in Figure 1.

![Data Model Diagram](image)

Figure 1: the proposed data model. The UML convention is adopted for the representation.

### 2.1 The measurement-specific layer

Following entities are involved in a monitoring process:

- **Object** or objects to be monitored in the scope of the given project
- **Sensor** or sensors that are used by a monitoring system to collect data about defined properties of the monitored objects
- **Channel** or channels of a sensor that represent a specific collection unit for a given property of the monitored objects that has been defined to be relevant for that project
- Position of single channels in form of a **geometry network** and a **topology network**
• **Session** or sessions that represent a set of collection events of the relevant properties defined for an object that relate to a given state in time of that monitored object

• **Measurement** or measurements that represent a single Sensor collection event in a session

• **Value** or values that represent the value of the property that has been collected by a channel through a measurement

• **Log** that represents an abstract supervisor of all events that take place in the monitoring system, such as calibrations, warnings, failures, user observations, etc…

**OBJECT**

An object represents a spatial entity (bridge, dam, landslide, etc.) that is part of a monitoring project. The object is given a name and has a creation date in the scope of the project. No special characteristics of the object (type, form, dimension, etc.) are modelled. In the scope of a project many objects can be defined. The project itself is defined through the resulting database-file. Different projects are defined as new database-files and are managed through the operating system.

**SENSOR**

A sensor represents a component that collects data about defined properties of the monitored objects. A sensor is made of one or more channels (see below), which are the actually collecting units. The sensor itself doesn’t collect the data, but the channels are responsible of that task. For example the X, Y and Z displacement-measurements of a GPS-instrument are the channels of the GPS sensor. A sensor is given a name and a number identifying the specific type of sensor. A list of sensor types is defined and available with the standard.

**CHANNEL**

A channel represents a single collection unit, which is part of a sensor. Through the channel data about a defined property of the monitored object is collected. The channel is given a name and has an associated active calibration entity (see below). This calibration has to be considered as the present valid calibration for the channel. The calibration defines the polynomial transformation between raw measurement data (primary data, e.g. a resistance) and actually used data (secondary data, e.g. a temperature derived from a resistance).

**POINT**

A point is a simple geometric entity used by the model to define the position in the space of a single channel. A point is given a name and three coordinates, which are defined in a specific reference system. The reference system can be a geocentric reference system, a land projection system or a local reference system. A number identifies the specific reference system for the point. A list of reference systems is provided with the standard. The unit through which the coordinates are expressed is also defined for a point.

**GEOMETRY**

The relationships through points in the space and the channels are given by the geometry entities. Through the geometry entity is possible to describe the position in the space of a channel, with one or more points. It is also possible to model the case of many channels at a given point (standard case by GPS). In the case of many points the topological relationships between them is given through an index. This simple index permits to navigate along the discrete geometrical representation of the channel. For example a point sensor will have a single point (with index 1), a baseline sensor will have two points (with index 1 and 2) and a sensor installed along a curve will have a larger number of points.
The geometry relationship has a timestamp to store its date of creation. Because many objects are in a continual displacement the actual position of a channel doesn’t correspond with the position given by a geometry entity.

SESSION
A session entity represents a set of collection events (measurements, see below) that permit to define a given state in time of the monitored object. The main goal of a session entity is the correlation of different measurements at a given point in time or in a defined interval of time. In fact a session time period can vary from a moment in time to a defined interval (e.g. 1 day).
A session entity is given a name and a time-window, in which the set of measurements are representative for the state of the object. The model considers the possibility to flag the session entity when his validity is not given.

MEASUREMENT
A measurement entity is defined as a single collection event in a session through a defined sensor. Considered by the model are the cases of zero, one or multiple measurements for a given sensor in a given session. A measurement event for a given sensor involves all the channels defined for that sensor.
A measurement entity is given a time-stamp to relate the produced values with the time. The measurement itself can be an instant or an interval of time. The model considers the possibility to flag the measurement entity when his validity is not given.

VALUE
The value entity is defined as the value of the related property that has been collected by a channel in a given measurement. The model considers two distinct situations: the case of discrete and representative values (maxima, minima, etc.) of the related measurement and the case of a complete and continual values list. In the first case a value is stored as a scalar one, in the second case a binary representation of the produced values can be used.

LOG
The log entity is defined as an abstract supervisor of all events that take place in the monitoring system. The goal of this entity is to store all of the important events that permit to reconstruct the operational history of the monitoring process. Examples for such events are calibrations, warnings, failures, user settings at the system-specific layer, etc… Different sources for the events are possible such as user-driven, session-driven or sensor-driven events.
The model defines for a log a specific event type, the source type, the source name and the relative time-stamp. A list of the possible log-events and relative source types is available with the standard.

2.2 The configuration layer
In figure 1 the grey entities are specific for the configuration layer. The following entities are specified by this model:
- **Agenda** and relative **agenda-items** to permit the planning of manually-configured or automatically-configured execution of measurement events
- **User** entity to store information about people that work with the monitoring system or are involved in the decision process supported by the monitoring process
- **Calibration** entity to permit the calibration of single channels.

AGENDA
The agenda entity is defined to permit the planning process of measurement executions. An agenda can be considered as a specific configuration for the measurement event. This configuration is sensor-independent and allows manually or automatically execution of sensor measurements. The planning process results in associating different configurations with different sensors.
The model sees also an agenda as a list of sensors that have to be measured for a given project. Instructions on how the sensors should be measured are defined. For different possible situations, it is possible to create different agendas, containing a different configuration of sensors to be measured (e.g. an agenda for manual measurement of only a few sensors, one for automatic measurements on all sensors). This gives the planning process a great flexibility. If a sensor has to be measured more than once in a session there will be many entries in the relative agenda. An agenda is given a name and a list of instructions for the execution event.

AGENDAITEM

The agenda-item entity specifies which sensor has to be measured with a given agenda. Through the definition of agenda-items results the planning process for the measurement of single sensors.

USER

The user entity permits to store information about people involved in the monitoring process. That information results to be important by the definition of automatic alarm systems based upon new technologies (e.g. SMS or phone-calls with predefined messages). This information is also relevant for application-level access control (e.g. limiting the access to application services).

CALIBRATION

The calibration entity is defined to permit the calibration of single channels. The calibration concept is based upon the difference between an active calibration and an original calibration. An active calibration is the present valid calibration for a given channel. An original calibration is the actually used calibration for a resulting value in a measurement. This concept permits to define different calibrations for the channels, and to always store the actually used calibration. The calibration itself is defined as a transformation function between primary and engineering units. The engineering unit is calculated with the following formula:

\[ EU = \text{coef}_0 + \text{coef}_1 \times PU + \text{coef}_2 \times PU^2 + \text{coef}_3 \times PU^3 \]

where EU is the engineering unit and PU is the primary unit.

2.3 The system-specific layer

The model defines this third layer as a system-specific and vendor-specific layer. No entities are defined for this layer. Possible entities that can be defined as part of this layer are:

- Application level access controls
- Sensor specific configuration parameters
- Session and measurement specific configuration parameters
- Others

2.4 Implementation specification for the data model

The following specification is intended for the implementation of the data model with relational database technology. The specification is based on the SQL-99 standard. The entities of the data model are mapped to tables and the attributes are mapped to fields of that table. Data types for the attributes are part of the SQL-99 standard. The preferred database implementation is the one of MS Access, but other (MS SQL Server, Oracle,…) can also be used.

3 IMPLEMENTATION EXAMPLES

The presented Database structure is the core of SMARTEC’s SOFO SDB software. This software is a DAQ used to measure our SOFO sensors [2,3] and other compatible instruments (e.g. Advantech’s ADAM 4000 Modules). Besides the
SOFO DB software we have developed a data visualization package called SOFO View allowing the representation of data from an SDB database, a web interface and a data analysis package called SPADS [4] and dedicated to the analysis of curvature measurements from bridge monitoring projects. These modules are schematically represented in Figure 2.

Figure 2 DAQ and DRA components from SMARTEC SA
Figure 3. Implementation example of a DAQ software. The navigation section shows how it is possible to move within different Agendas, Sensors, Campaigns and Measurements.

Figure 3 shows an example of implementation of a DAQ software module.

4 CONCLUSIONS

The use of relational database structures greatly simplifies the management of large data sets generated by automatic monitoring systems. The Database structure can be used as a central node between the data acquisition (DAQ) and the data representation & analysis (DRA) software packages. This allows new measurement systems to be immediately recognised by the existing DRA packages and vice versa. We propose the SDB database structure as an open standard allowing interoperability between DAQ and DRA packages from different producers and developers.

REFERENCES