Using Models to allow for Object Oriented Programming of the Vacuum Control Systems

R. Gavaggio, M. Steffensen and P.M. Strubin,
CERN, AT-VA
CH-1211 GENEVA 23

Abstract

Recently, most vacuum measurement and control equipment have been described using generic models, representing their behaviour and data exchange. This is a first step towards describing these equipment as objects, in the sense of programming. In such objects, the data (e.g. a pressure read by a gauge) can be described as the attributes of the object. This data will be accessed using the common get and set primitives. The behaviour of the object in response to commands (e.g. open valve) will be implemented using methods, the models being the basis of abstract methods. These methods may have to be overshadowed for each specific hardware implementation. As the CERN accelerator control systems are build in layers, the aim of this approach is to have the equipment specific software at the lowest possible level in these layers. This paper gives a summary of the existing models and an overview on how this approach can be implemented in the context of the CERN accelerator control systems, where we have to deal with equipment ranging from dumb supplies to intelligent controllers.

1 INTRODUCTION

Vacuum equipment used in accelerators has grown in variety and complexity over the years. Unfortunately, there exist a very large number of different interfaces, making it more and more difficult to share applications at user interface level.

In 1991, a working group was set up at CERN, with the aim of producing operational models for different type of equipment, including vacuum equipment [1]. These models define the data flow to and from the equipment as well as the different possible values of the status.

In the past years, object oriented programming has become more and more popular. The modelling of equipment is completely in line with this approach.

2 MODELS

Vacuum equipment can be grouped in basic classes, like pumps, gauges, valves, etc. From these classes, it is possible to build more complex elements, like pumping stations or even complete vacuum sectors. The behaviour of each type of equipment must be described as well as the data flow to and from the equipment [2].

So far, no assumption is made on how the data is transmitted. At the level of, say, a gauge power supply, it can be anything from an analogue output proportional to pressure up to a network interface capable of exchanging messages. What is important, however, is that the defined data flow must be available in one form or another. It is equally important that the behaviour of the equipment as a result of a command follows the definition of the model.

As an example, we show below the various part forming the model of an ion gauge connected to the appropriate power supply, as it is made visible to the control system.

2.1 Definition and operation

- Bayard-Alpert, modulated, ionisation gauge connected to a remotely controllable power supply.
- The gauge is operated by heating and regulating one of its filament and measuring the ion current on a collector. A special electrode, called modulator, can be used to evaluate the X-ray contribution to the pressure reading. The useful operating range is between 10^{-3} and 10^{-10} Pa.

2.2 Context and data flow

The model must define the data flow to and from the equipment (Figure 1 and Table 1). Full lines and bold text denote mandatory variables, whereas shaded lines and plain text show optional variables. However, to cope with the largest possible number of available equipment, some of the data can be defined as optional (e.g. setting and/or reading the emission current of the filament)

2.3 State diagram

The last part of the model is the representation of the various possible states the equipment can take, as well as the ways to reach into these states (Figure 2). All stable states must be shown on the state diagram. Transitional states may exist when an operation takes a significant time to happen when compared with the typical time needed to make an acquisition from the control system. All states, stable and transitional, must be reflected in the possible values of the variable STATUS.

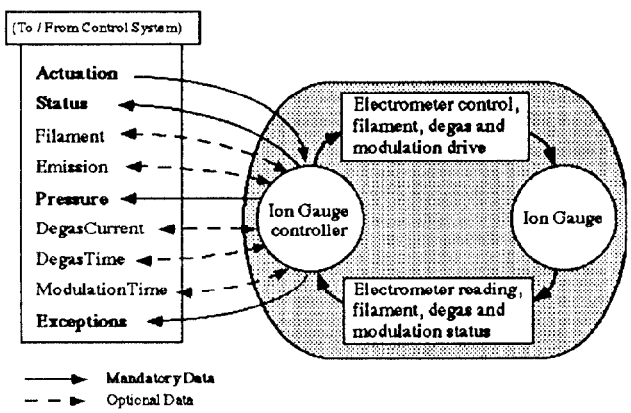

Figure 1. Context and data flow

Variables	Type	Values	Units
Actuation	Discrete	OFF, ON, DEGAS, MODUL	
Status	Discrete	OFF, STARTING, ON, START_DEGAS, DEGASSING, MODULATING	
Filament	Integer	Positive only	
Emission	Real	Positive only	A
Pressure	Real	Positive only	Pa
DegasCurrent	Real	Positive only	A
DegasTime	Real	Positive only	s
ModulationTime	Real	Positive only	s
Exception	Structure		
FaultStatus	Boolean	False, True	
FaultList	Set	BrokenFilament, EmissionFault, GridVoltage, etc...	
WarningStatus	Boolean	False, True	
WarningList	Set	UnderRange, OverRange, etc...	

Table 1. Variable names and types

3 MAPPING MODELS TO CLASSES

The data flow defined by the model can quite naturally be mapped onto attributes and methods of a class. As many of the attributes can be found in several type of equipment (e.g. pressure values), one can have abstract classes like gauges, with subclasses as ion gauges, cold cathode gauges, etc. The attributes reflect the various data structures. They can hold the result of an acquisition (e.g. pressure), a set point value (e.g. emission) or the status variable of an equipment. Methods are required to implement the ACTUATION variable of the model.

An example of such a class hierarchy was tried out using an object oriented pictorial programming tool on a Macintosh (Prograph, from TGS Systems). Figure 3 shows the attributes for the abstract classes Gauges and Ion gauges, as well for a instanciable class of a modulated Bayard-Alpert gauge (corresponding to a gauge type used in LEP).

The abstract class Gauges defines the attributes and methods available to all type of gauges, that is: Pressure, Status and Exceptions. The abstract class Ion Gauge adds the attributes specific to all type of ion gauges, in particular the emission and degassing parameters. The real class VGSV, which can be instanciated, adds the attributes required for a modulated ion gauge.

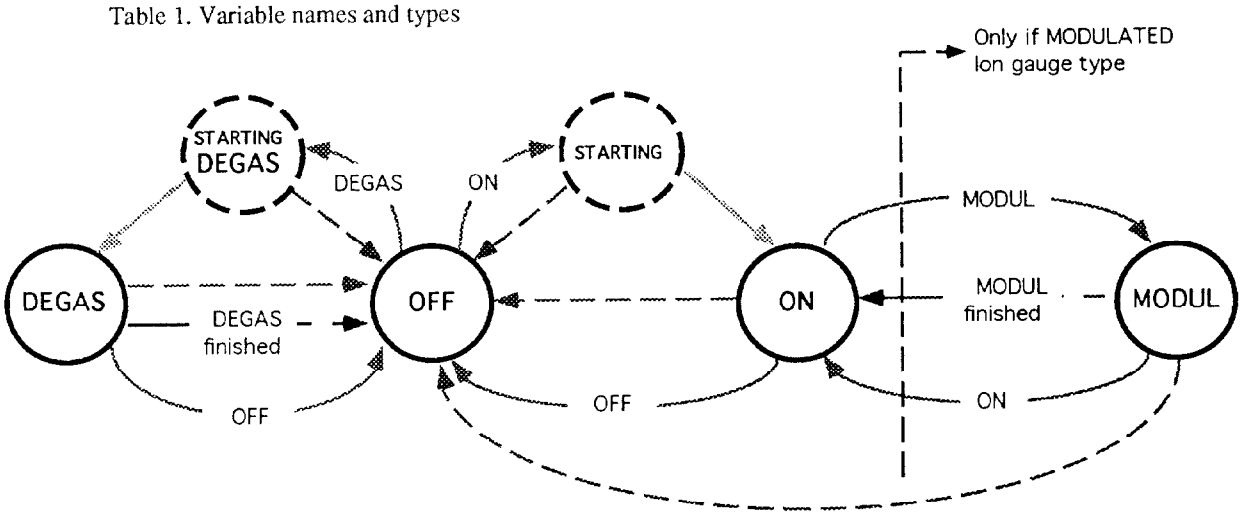

Figure 2. Possible states

The filled attribute symbols denote inherited attributes. Each attribute is given a default value, which can be overwritten in the sub-class if required. Note also that the exceptions have been defined as a class themselves to reflect the structured data type and to allow for specific methods to clear exceptions.

One can see from these classes that the abstract class Gauges is probably all what is needed for normal operation, as it implements all the mandatory variables of the model. The class Ion Gauges implements the complete set of variables of the model. But it also includes attributes like ion current and sensitivity which are inputs to the method that evaluates the pressure. Both these attributes and the method are local to a specific implementation of an ion gauge controller.

The possible values of the variable ACTUATION are translated into methods. In this way, it is possible to satisfy to the various hardware implementation by overshadowing the generic methods of the abstract class by specific one in the instanciable class.

4 IMPLEMENTING THESE CONCEPTS IN A REAL CONTROL SYSTEM

These concepts, although quite easy to define in theory, may need some effort to be actually implemented. Objects, in general, receive messages to activate the methods which are defined within their scope. In addition to the methods implementing the various values of the variable ACTUATION, each object must at least have two primitive

methods: get and set attributes. The main purpose of these two methods is to make the access to data independent from its representation. In fact, there is probably no need for any other method for a first implementation. If the get and set methods are made general enough to be able to return not only values of attribute, but also the name of attributes or even the name of embedded objects, it will be possible to dynamically drive any of these objects.

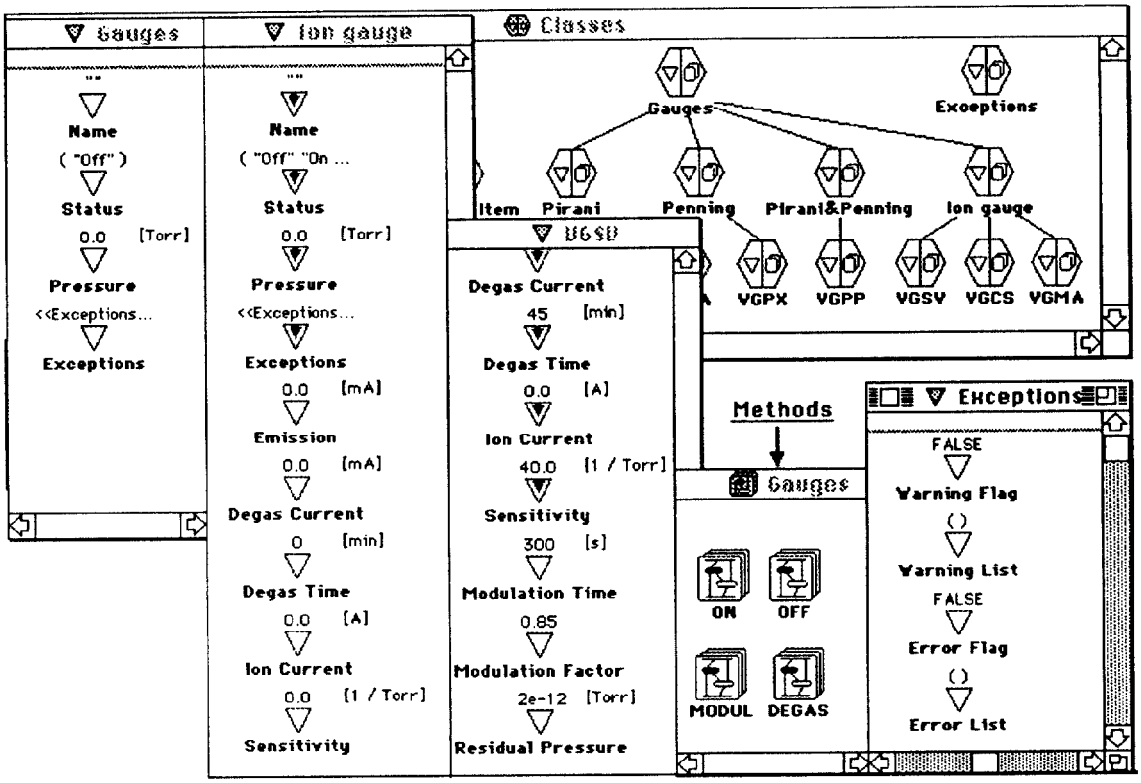

Figure 3. Class hierarchy for gauges

The class definitions of the instanciable classes must be stored in a central database, along with the default values (and possibly the limits) of the various attributes. Lists of object names and description can be produced from the available databases which describe the topology of the various accelerators. Provided one adds one or more attribute to every object to hold position information, display programs using synoptics can be fully data driven. Once object oriented databases will be available, it will be possible to store the objects themselves (attributes and methods)

The location where the model or the object will be implemented may also considerably vary depending upon the architecture of the control system. In any case, it should be at the lowest possible level, to maximise its usefulness. The minimum requirement to fulfil is to be able to process the set and get attribute methods and to take the appropriate action when a method related to ACTUATION is invoked.

In the case of the various vacuum systems at CERN, quite a range of implementations are possible. Where there is intelligence available at the level of the power supply, as in LEP, the models can be implemented in the various power supplies. The message type which is used is also suitable to invoke the methods in the objects.

In other cases, the use of industrial equipment drives us to keep the models in the concentrating equipment, e.g. the VME chassis available in various places [4].

5 CONCLUSIONS AND FOLLOW UP

The definition of operational models for vacuum equipment has already been done for most of the equipment. These models lead themselves quite naturally to a representation in object based programming. The available infrastructure of the control system of LEP and SPS will allow us to make a trial implementation within a short time.

The obvious benefit of this approach will be to be able to re-use the software written for the higher levels of the controls system, e.g. the man-machine interface and to develop towards a completely data driven environment. In the long term, it is desirable to propagate this model or object based approach towards the manufacturers of equipment.

6 REFERENCES

[1] G. Benincasa et all, Finale report on the uniform equipment access at CERN, CERN/PS 93-16 (1993)

[2] P. Strubin et all., Operational Protocols for Vacuum Systems, CERN AT/VA 91-6 (1991)

[3] P. Strubin, Controls for the vacuum system of LEP, J Vac Sci Technol, A5(4), Jul/Aug 1987

[4] W. Koelemeijer, M. Nicoules, M. Steffensen, P.Strubin, Upgrading the Controls of the SPS Vacuum System, EPAC 92