THE VIRTUAL UNICOS PROCESS EXPERT: INTEGRATION OF ARTIFICIAL INTELLIGENCE TOOLS IN CONTROL SYSTEMS

I. Vilches Calvo[1], R. Barillère[2], G. Thomas[3], CERN, Geneva, Switzerland

Abstract

In the context of the CERN Large Hadron Collider (LHC), a framework called UNICOS (UNIfied Industrial Control Systems) was developed to produce three-layer control applications [1]. It provides users with means to develop full control applications and operators with ways to interact with all items of the process from the most simple (e.g. I/O channels) to the most abstract object (e.g. a sub part of the plant). This possibility of fine grain operation is particularly useful during commissioning but also to recover from abnormal situations provided the operators have the required knowledge, which is, unfortunately, not always the case.

The Virtual UNICOS Process Expert tooling suite aims at providing operators with means to handle difficult cases for which the intervention of process experts is usually required. This is typically the case when operators are faced with alarms avalanches that triggered interlocks in some parts of large plants. The main idea of the project is to use the openness of the UNICOS-based applications to integrate tools (e.g. Artificial Intelligence tools) that will act as process experts to analyze complex situations, to propose and to execute smooth recovery procedures.

The paper focuses on the first version of the software suite which was developed for the LHC experiments Gas Control Systems (GCS) [2]. The paper first introduces the architecture of the UNICOS based applications and the problems operators are faced with. Then the focus moves to the description of the tooling suite and its integration in the UNICOS architecture. Finally, it describes the benefits of the selected approach and ends up with hints about future evolution of the tooling suite.

INTRODUCTION

The LHC Gas Control System presently provides the LHC Experiments with homogeneous Control Systems for their 27 Gas Systems. These Gas Systems are built on a modular architecture [3], which consists of a set of functional modules where the number and type of which differ from one system to the other. In addition, some of the functional module devices are optional. Although these 27 Control Systems provide end-users with a consistent and homogeneous look and feel, they differ in their operation and require specific process knowledge for each sub-detector. The commissioning phase of the gas systems is a long process and is achieved by a small team with only a few Gas System Experts.

Therefore, time and detailed procedures are necessary for the Process Expert to train its team for it to be able to operate each of these systems. For this purpose, a Virtual UNICOS tool suite is being designed to provide guidance to operators during the most complex operational procedures and to allow capturing the knowledge from the Process Experts so that it can be re-used.

[1]ignacio.vilches.calvo@cern.ch
[2]renaud.barillere@cern.ch
[3]geraldine.thomas@cern.ch

DRIVING REQUIREMENTS

The Operators need guidance and help to understand which set of actions should be executed (and in which order) to drive the System to a given Target state.

The Experts would appreciate a tool for capturing their knowledge, acquired during several years of experience and knowledge of these systems. This knowledge is very volatile and sometimes, it requires a lot of time to remember. The tool would easily capture this knowledge, bring it to the Virtual Operator and show it to the Operator.

Both Experts and Operators would also agree to have a configurable tool, transparently integrated into any UNICOS-based applications.

ARCHITECTURE

Integration in the UNICOS Architecture

The UNICOS-based applications like the LHC GCS project are controlled with industrial Components Off the Shelves (COTS) which are distributed across three layers:

-The *field layer* includes the process sensors and actuators connected to the control system via PLC I/O boards and/or field buses.

-The *process control layer* includes the logic, interlocks and the Finite State Machine (FSM) of the system.

-The *supervision layer* allows the monitoring and control of the system by means of a Supervision Control And Data Acquisition (SCADA) where ETM's PVSSII® has been selected.

The design of UNICOS is based on the physical model of the IEC61512-1(ANSI/ISA-88**)** standard where the process has been broken down into a hierarchy of objects where each object has a unique parent. These objects are organized in three classes:

-The *I/O objects* link the sensors and actuators to the plant for reading data and sending commands.

-The *field objects* can represent a proxy of the hardware element such as valves, motors and others or perform control loop actions.

-The *Process Control Objects* (PCO), which are the node of the project hierarchy, are responsible for the process specific logic (FSM, interlock conditions, etc...). These UNICOS objects are implemented in the PLC with its proxy in the supervision layer and a middleware handles the exchange of commands and status between the PLC objects and their PVSS proxies.

The Virtual Operator introduces, at the level of the supervision layer, a User Interface (UI) as a PVSS *CtrlExtension* with an Expert System embedded. The principle of this Virtual Operator is to simulate a human Operator by analyzing feedback status from I/O object or field objects and sending commands to the top hierarchy objects, namely the PCO.

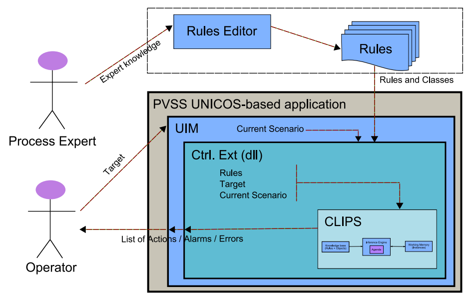

Figure 1: Virtual operator global architecture.

For the Virtual UNICOS Operator a PCO is referred to *Process Part*.

Operator – PVSS Interface

The Operator selects the Target state of the System from the user interface. An objective for the target can be, for example, to recover the System from an abnormal situation. To reach it, the Operator can select target states for one or more Process Parts of the System. The Current Scenario (current PCO values and states, alarms, etc...) is sent to the Virtual UNICOS Operator together with the Target. The AI tool applies the rules on it and then, it returns an ordered list of actions the System should follow, in order to get to the Target and, if the Operator wishes, the Virtual Operator executes all the actions automatically. The AI tool checks the new scenario every time an action is executed.

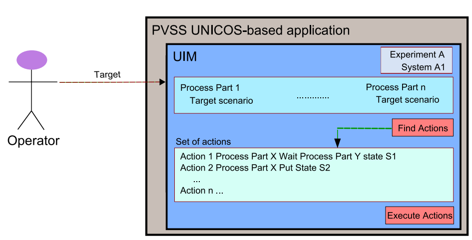

Figure 2: PVSS user machine interface.

In case something wrong happens while executing, the AI tool stops and it asks the Operator for fixing the problem (an alarm has been triggered for example). Once it is fixed, the AI tool shows the new list of actions for the new scenario created after solving the problem (there is no need to start from scratch).

Expert System/Rule-based System

An Expert System or Knowledge-based [4] System emulates the decision-making ability of the Experts, intending to act in all respects like them.

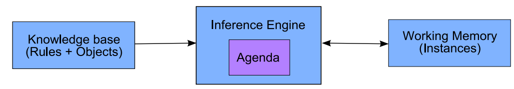

Figure 3: Rule-based system.

Internally the Virtual UNICOS Operator Rule-base System contains:

- *Knowledge base*: the knowledge of the Expert needed to solve the problems coded in the form of rules.

-*Working Memory* (WM): a collection of data objects that represents the current state of the System (values, sensor pressures, states, alarms, etc ... from each Process Part). It is a global database of instances used by the rules.

-*Inference Engine*: makes inferences by deciding which rules the objects placed on the VM satisfy. It also prioritizes the satisfied rules and it executes the rule with the highest priority. The Inference Engine contains:

**Agenda*: a prioritized list of rules created by the inference engine, which patterns are satisfied by objects in WM.

**Rete algorithm*: fast pattern matcher that obtains its speed by storing information about rules in a *pattern network* [4]. Instead of having to match instances against every rule on every recognized-act cycle, it looks only for changes in matches on every cycle.

Several solutions have been evaluated as AI tool. The Expert System language selected is CLIPS [5] (C Language Integrated Production System).

The main reasons are that CLIPS provides high portability (can be ported to any system that has an ANSI compliant C compiler), easy integration/extensibility (can be embedded within procedural code, called as a subroutine or easily extended by the user), an interactive environment for testing, an extensive documentation and low cost (free).

CLIPS provides support for three types of programming paradigms: rule-based, object-oriented and procedural programming. The Virtual Process Expert is based on the rule-based (allows knowledge to be represented as a set of actions to be performed for a given situation) and on the object-oriented programming capabilities, usually referred to as COOL (CLIPS Object-Oriented Language), which allows systems to be modelled as modular components to be easily reused to model similar systems or to create new components.

Process Expert – Rules Editor - Rules

The Expert knowledge is acquired and then translated into rules files by a rules editor. The rules editor is a tool to ease the transfer of knowledge between the Expert and the AI tool. Currently, the rule editor contains a template Excel files. These files are filled by the Experts and then translated into rules by the UNICOS-based application developers.

Each Process Part is represented by a Class in the Rule-based System. Each class has properties (slots) representing, for example, states, values, etc ... of the Process Parts. Each rule is made up of a bunch of preconditions and post-conditions representing transitions between Process Parts states (constraints of the System) and decisions (Expert knowledge).

The preconditions are the Target, the Current Scenario, dependant values and states (one Process Part could be waiting for another Process Part values or states before it can be moved from one state to another). These dependant Process Parts values and states are one of the most important knowledge bits obtained from the Experts.

The post-conditions are actions to be done, new sub-targets (targets for the dependent Process Parts), new states and values (properties) for the objects. At the level of the Virtual Operator, the post-conditions represent an intermediate scenario that will be executed, in order to move forward to the get to the Target.

If all the preconditions of a rule are true, the rule is fired and the postconditions are triggered.

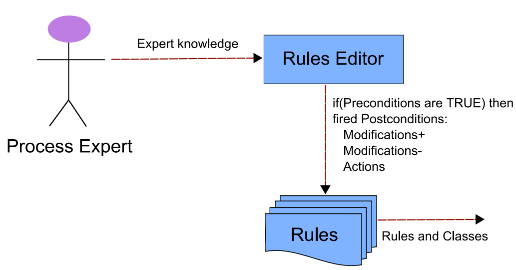

Figure 4: Knowledge acquisition.

When the Virtual UNICOS Operator found a way to reach the Target, it stops and it returns the set of actions to the Operator, who then has a step-by-step planning to follow, as if he were a Process Expert.

PVSS – Virtual Operator Communication Interface

The communication between CLIPS and UNICOS-based applications is done by the PVSS *CtrlExtension* functionality. These extensions allow C++ functions to be added to the PVSS programming language.

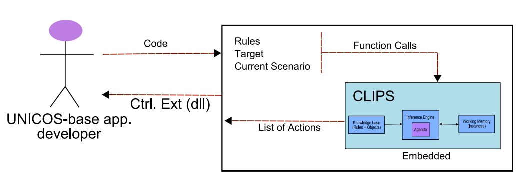

Figure 5: Control extension.

The PVSS *CtrlExtension* consists of a class derived from *BaseExternHdl* where the developer can add the functions required. The Virtual Operator, thanks to CLIPS portability, has embedded CLIPS source code into the *CtrlExtension* so CLIPS functions can be called easily and any functions can be modified or added if needed. Calls to CLIPS are made like any other subroutines.

The *CtrlExtension* must be exported as a dll so that it can be called by the PVSS UI Manager. Then, any UNICOS-based applications can use CLIPS by calling these functions from any PVSS script or UI.

ACHIEVEMENTS

At the time of writing, a first version of the Virtual UNICOS tool is being finalized. This first version has been satisfactory tested with a few real use cases given by the GCS Experts (alarms, states, values, etc).

The feedback received from the Experts (functionality and UI) is being very positive.

Although, at this date, we have no benchmark, the performance has no impact on the Control application.

The AI tool is very flexible to new changes. The rules can be changed independently of the Interface. By modifying a rule it can modify the behaviour of the whole System.

PERSPECTIVES

The rule editor is being improved for providing a new look and feel to the Experts to easily add/mod/del the knowledge (the rules).

The PVSS *CtrlExtension* will likely be replaced by a PVSS API (Application Programming Interface) which offers a set of functions enabling it to extend this process control system by adding special managers.

More use cases are being taken now which will increase exponentially the utility of the AI tool

It can easily be extended to represent similar systems.

The AI tool will be ready to analyse the alarms, helping the Operator to understand why they have been triggered explaining the causes.

REFERENCES

[1] P. Gayet et al., "UNICOS A framework to build Industry like control systems: Principles and methodology", ICALEPS'2005, Geneva, Switzerland.
[2] The LHC GCS project URL: http://itcofe.web.cern.ch/itcofe/Projects/LHC-GCS
[3] G. Thomas et al., "LHC GCS: A model-driven approach for automatic PLC and SCADA code generation", ICALEPS'2005, Geneva, Switzerland.
[4] Giarratano and Riley, "Expert Systems: Principles and programming. Third Edition", PWS.
[5] "CLIPS. Reference Manual. Volume II. Advanced Programming Guide", Version 6.24.