RESULTS OF THE OPC EVALUATION DONE WITHIN JCOP FOR THE CONTROL OF THE LHC EXPERIMENTS

R. Barillère, V. Baggiolini, M. Beharell, D. Chmielewski, P. Gras, H. Milcent, K. Kostro,
CERN, Geneva, Switzerland
A. Liiou, INR, Russia
V. Khomoutnikov, IHEP, Russia

Abstract

The construction of the LHC experiments' control systems will require the integration of a wide range of COTS (Component Of The Shelf) and custom components: hardware such as instruments, controllers, fieldbuses and sensors as well as applications, for example, for operator control and visualization or for sub-systems supervision.

This integration may require a non-negligible effort if standard interfaces or integration mechanisms are not applied.

OLE for Process Control (OPC) is a recently defined set of interfaces designed to allow Windows applications to access control data. OPC is based on Microsoft DCOM and is developed by the OPC foundation [1].

This paper presents the result of an evaluation done in the context of the Joint Controls Project (JCOP) [2]. The aim of this evaluation was to study the usability of OPC for the detector control of the LHC experiments. In particular it presents the benefit and limitations of the specifications, the availability of OPC compliant COTS and the usability of OPC development kits to develop applications accessing custom devices or non-Windows platforms.

1 OPC OVERVIEW

1.1 Introduction

OPC™ (OLE for Process Control) defines a set of interfaces, based on OLE/COM and DCOM technology, for truly open software application interoperability between automation/control applications, field systems/devices and business/office applications (See Figure 1).

OPC is managed by an independent body, the OPC Foundation, which counts more than 200 companies and institutes as members.

The foundation has released two sets of interfaces:

- OPC Data Access (V2.0, October 1998);
- OPC Alarms and Events (V1.0, December 1998).

A third one is in preparation:

- OPC Historical Data.

As these specifications are rather recent, we focused our evaluation on the OPC Data Access, which is the only one actually supported by industry.

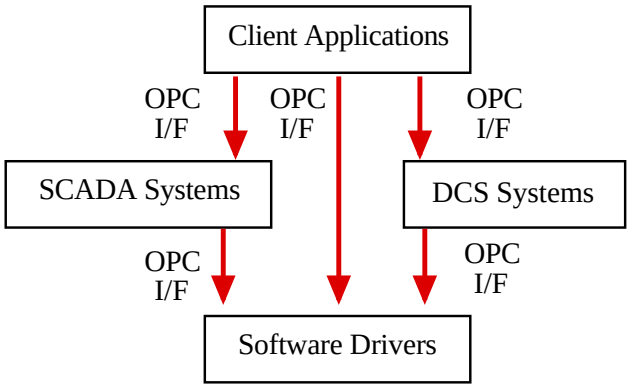

Figure 1: Example of where OPC may be applied[1].

1.2 Concepts

1.2.1 A Client/Server architecture

OPC is a client/server system. *OPC servers* hold process data for *OPC clients*, OPC clients read, write and subscribe to the OPC servers' process data. Relations are n-to-m: an OPC client can simultaneously interact with several OPC servers and several OPC clients can access the same OPC server.

OPC servers optionally offer introspection facilities to allow clients to browse the available process data.

1.2.2 OPC groups and OPC items

OPC servers make process data available by means of *OPC items*. An OPC server creates OPC items on behalf of an OPC client. The client's OPC items are organized in *OPC groups*. OPC clients can only access their OPC items through their respective OPC groups[2]. The Figure 2 shows two OPC clients accessing process data managed by an OPC server. The first OPC client accesses its OPC items through two groups, the second one has only one group.

[1] DCS stands for Distributed Control System.
[2] See the OPC specification [3] for details about OPC groups and OPC items.

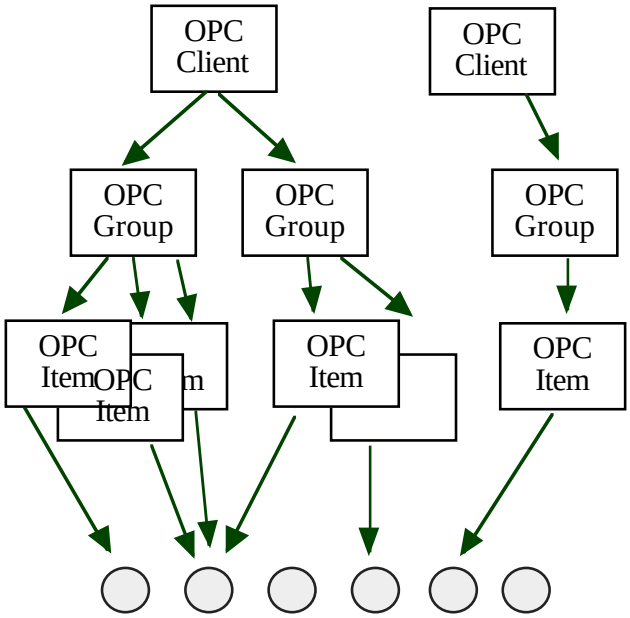

Figure 2: The OPC concepts

1.2.3 Data access mechanisms

Four communication mechanisms have been defined for OPC™ Data Access: synchronous and asynchronous read/write, refresh and subscription.

When a client issues a synchronous read call, the server does not return the control to the calling thread until sending back the requested values. When a client issues an asynchronous read call, the server immediately returns control to the calling thread and later, using a call-back mechanism, sends the requested values.

The synchronous and asynchronous calls require that clients specify the list of process data that have to be read or written.

Refresh and subscription are callback mechanisms. They are used to access predefined sets of process data. Refresh is a "pull" mechanism, subscription is a "push" one. When an OPC client issues a refresh call, the OPC server asynchronously returns the current values of the predefined set of data points using a predefined communication path. Subscription is an event-based mechanism. OPC servers notify clients when significant changes (i.e. bigger than the dead-band defined for the OPC group) occur within a predefined set of process data.

2 MOTIVATIONS

The main purpose of the JCOP evaluation was to decide if OPC could be recommended as the standard interface to the various devices to be controlled in the LHC experiments, as such an interface would obviously ease the integration of both industrial equipment and home made devices.

The use of OPC as middleware for the control of the LHC experiments or to interface high level applications was also considered.

To assess this, a set of criteria has been identified among which are the adoption of OPC by the market, OPC openness and flexibility, performance and reliability of OPC implementations.

3 EVALUATION

3.1 Capabilities

Although OPC is not fully object-oriented - process data are accessed through OPC items (i.e. tags) - we appreciated the four communication mechanisms of OPC. One can use both the read/write calls and the subscription to adapt the communication to the access speed of a device.

Timestamps and quality flags that can be attached to the retrieved values are useful features for distributed control systems.

However, we found weaknesses in the OPC capabilities; they are mainly due to the tight coupling of OPC with DCOM. DCOM does not offer any naming service like the one proposed by CORBA. The deployment of a distributed control application is therefore more difficult with OPC and DCOM than with CORBA. The location of OPC servers has to be supported by a custom OPC client configuration or to either be contained in the OPC client source code or in the Windows registry of the OPC client PC.

OPC security as well is entirely based on the NT security. It is not easy to use and the granularity is not fine enough. One can only specify access rights for OPC servers. We would have appreciated to be able to set these rights at the level of the process data. If this feature is not improved with the next releases of the specifications, the access control of our applications will have to be implemented at a higher level up in the SCADA system.

3.2 Market

As demonstrated by the OPC catalog[3] published by the OPC foundation, OPC has been widely adopted by the market. A lot of OPC servers are available to access fieldbuses and Programmable Logic Controllers (PLC). Most of the SCADA (Supervision Control And Data Acquisition) systems have an OPC Data Access client driver and sometimes are OPC Data Access servers. The others plan to support OPC in the near future. High level applications such as expert systems are also available as OPC clients.

When OPC servers are not available for a specific device, toolkits can be used to develop the required ones.

[3] The catalog is accessible from [1] and [4].

More than 30 toolkits to develop clients and servers are referenced in the OPC catalog.

Although the specifications are rather new, one can find industrial products supporting the second version of OPC Data Access and/or the first version of the OPC Alarms and Events.

3.3 Compatibility

While running the evaluation and developing small projects[4], several OPC servers and clients, mainly compliant with the OPC Data Access v1.0, were used concurrently. We did not experience any major problems as far the compatibility is concerned. The minor problems we encountered were due to different interpretations of the specifications. The OPC foundation is aware of this problem and has set up a compliance working group.

3.4 Openness and flexibility

To integrate very specific devices or devices controlled from non-Windows platforms (e.g. VME or PC with Unix) we need to develop our own OPC servers. As a consequence of OPC relying on DCOM, Windows NT is required to run any of the OPC components.

We found a large number of toolkits to support these developments, some of them are provided with evaluation versions. We developed rather easily OPC servers to access CAEN and Lecroy power supplies and to access devices through CORBA[5]. The use of these toolkits usually requires knowledge in C++. The most advanced toolkits shield entirely developers from DCOM. This is indeed an advantage.

3.5 Reliability

The reliability of OPC obviously depends on DCOM (as the underlying technology), on the toolkits used to produce the OPC components and on the developers' skill.

The DCOM communications timeouts being rather long (several minutes), OPC clients (respectively servers) will be notified lately about the unavailability of the OPC servers (respectively clients) they are in communication with. A workaround consists in implementing watchdog mechanisms to control the quality of the communications between OPC components.

In the current DCOM implementation clients allocate memory in the server address space, they have then to free this memory. This approach can lead to memory leaks in case of problems with clients. This limitation could disappear with the next major DCOM release.

3.6 Performance

White papers about OPC performances are publicly available on the Web [5]. In order to verify these numbers we ran our own performance tests. Our results[6] were close to the white papers ones.

To run these tests, we developed an OPC server that generates test values and OPC clients that consume them.

We basically tested the synchronous writing of a set of OPC items in a single OPC server. We also tested the subscription to a set of items from a single OPC client.

To read n (measured from 1 to 10000) OPC items, a client needs 515 + (85 * n) µs in synchronous mode.

When all n OPC items (we measured with n=500 to 20000 items) of a group are modified, the minimum update rate is 80 * n µs.

These results seem acceptable for the control of the LHC experiments where the majority of process data is not expected to simultaneously change significantly.

4 CONCLUSIONS

Almost all SCADA systems are already or will be soon OPC clients and OPC servers. OPC can then be used at a higher level to interface SCADA with other COTS or high level applications (e.g. Finite State Machine).

Although OPC has some limitations, mostly inherited from DCOM, it is a reasonable solution for the integration of commercial or homemade devices in the control systems of the LHC experiments. The use of appropriate toolkits shields developers from DCOM and reduces the impact of the evolution of DCOM on the maintenance of the developed OPC components.

With the emergence of the new OPC interface specifications (Alarms and Events, Historical Data), and if DCOM is extended with naming services, OPC could be used as middleware for component-based control systems.

OPC is a de facto industrial standard that is likely to have its place in our future solutions.

5 REFERENCES

[1] The OPC foundation: http://www.opcfoundation.org/

[2] The Joint COntrols Project: http://itcowww.cern.ch/jcop/

[3] The OPC specifications: http://www.opcfoundation.org/specs.asp

[4] The JCOP OPC sub-project: http://itcowww.cern.ch/jcop/subprojects/OPC/

[5] White papers about OPC performances & security http://itcowww.cern.ch/jcop/subprojects/OPC/NewOPC/Status/

[6] JCOP workshop II presentations; http://itcowww.cern.ch/jcop/JCOPworkshop2/

[4] A list of JCOP projects where OPC servers have been used is available in the JCOP web site [4].

[5] Details about these OPS servers can be found in the JCOP OPC sub-project web site [4].

[6] Details about our evaluation process will be available on the JCOP OPC subproject web site [4].