A Novel Framework for Building Distributed Data Acquisition and Monitoring Systems

Vu Van Tan, Dae-Seung Yoo, and Myeong-Jae Yi
School of Computer Engineering and Information Technology, University of Ulsan, Ulsan 680-749, Korea
Email: {vvtan, ooseyds, ymj}@mail.ulsan.ac.kr

***Abstract*— We propose a novel framework for building a distributed data acquisition and monitoring system. Our novel framework is mainly based on XML leverages and OPC (Openness, Productivity, and Connectivity) techniques. Correspondingly, our framework-based system allows easily aggregating more than the hundreds or thousands of the existing OPC DA (Data Access) products in use today with new OPC XML-DA products in a flexible system that can support to exchange the complex data between the OPC products in both the vertical and horizon effectively. In addition, the high performance and abilities of reading and decoding any type of data from hardware I/O devices on the plant floor, which are truly important factors in application design, are also adequately proposed in this framework. To provide a framework for design and implementation of the control and monitoring systems, our framework is proposed as the throughout of standards for interfaces, architectures, and functionalities. This framework is significantly easy to reuse, maintain, and deploy to real industrial applications. In addition to guarantee security problems, we discuss the levels of security aspects in the control and monitoring systems considerately. These security aspects provide more security information for technical-level readers. Moreover, the evaluations of our framework and system performance analysis indicate that our framework-based system has a good design, an acceptable performance, and is feasible to apply to the area of control and monitoring systems today.**

***Index Terms*— Framework, XML leverage, distributed data, OPC, monitoring system, security, acceptable performance**

I. Introduction

With the rapid growth of Web technologies in e-business, it is also widely increasing important position in control and monitoring systems. In addition, XML is an open standard for exchanging the structured data on various platforms based on the Internet environment. This provides interoperability and data integration using the Internet. Recently, XML and Web technologies provide a new and powerful way of accessing and delivering the complex data from the hardware I/O devices on the plant floor, condition monitoring, etc. to users at the enterprise applications using Web-browsers or wireless devices [1]. Based on these leverages, the OPC Foundation formed the OPC XML-DA technical working group to define a new specification to move the same type of plant floor data as the existing OPC COM-DA. The OPC XML-DA provides vertical integration between the plant floor and condition monitoring system, maintenance system, and enterprise application using such as industrial standards, XML, and SOAP (Simple Object Access Protocol) [2], [3]. Moreover, the OPC Foundation has defined the OPC Complex Data for implementing both the OPC DA and OPC XML-DA. The OPC Complex Data working group is making enhancements to OPC Specifications based on requirements and feedback from other industry groups to address additional data types such as structures, binary, XML, etc. [4]. Thus, the design and implementation of the XML-DA Server that allows OPC Clients to read and decode any type of data from the hardware I/O devices (*i.e.*, measurement and control systems) are represented in [2]. Another proposal mentioned in [5], the authors proposed a data exchange protocol based on the OPC XML-DA. The OPC XML-DA has a big disadvantage that used the pure XML textual data representation. This causes much more network traffic to transfer data [6]. However, this protocol has focused on an acceptable performance by providing the several extensions. It was interested in the consistency with high level standards, multi-platform compatibility, and high performance.

Besides, the requirement of converting data between XML data and OPC Binary data make it to determine the required size of the memory buffer to hold the whole data. In addition, only the binary data are really optimal to transfer a large amount of data rather than XML data. Moreover, different platforms even different compilers of the same platform use different binary representations such as different floating point formats, different string formats, etc. [4], [6]. Therefore, some universal standards should be chosen to effectively transport data between different platforms.

In use today, there are more than the hundreds or thousand of the existing OPC DA products and new OPC XML-DA products that require easily aggregating into a system, which easily allows that the complex data are exchanged not only between the OPC Clients and OPC Servers, but also between the OPC Servers. Moreover,

This journal paper is based on "Modern Distributed Data Acquisition and Control Systems Based on OPC Techniques", by Vu Van Tan, Dae-Seung Yoo, and Myeong-Jae Yi, which appeared in the *Proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer Based Systems*, ECBS'07, Tucson, Arizona, U.S.A, March 2007.

this system should be flexibly configured by using remote configuration clients through the Internet.

This paper proposes a novel framework for building the distributed data acquisition and monitoring systems that based on the OPC and XML technologies. Our novel framework is significantly proposed as the standards for interfaces, functionalities, and architectures. Indeed, our framework-based system allows easily aggregating the hundreds or thousands of the existing OPC DA Servers in use today with the XML-DA Servers into a system for supporting the horizontal data exchange between them. Moreover, the security aspects are carefully discussed in our works. The framework discussion and performance analysis indicate that our framework has a good design and is feasible to apply to real industrial applications.

This paper is organized as the following sections: The next section shows the generic model of Web integration, overviews of related OPC techniques, and the problem statements. Section III proposes the framework for design and implementation of the Universal DA Server using in control and monitoring systems. Section IV investigates the data representation between the Universal DA Server and the OPC Clients with the results of the benchmarks. To provide an open way for expanding in the future, the connection between database and our system is discussed in Section V. In Section VI, the security aspects are also carefully discussed. The framework and performance discussions are exposed in Section VII. These analyses indicate that our system has a good performance and is expected applying in industrial systems, especially for control and monitoring systems. Finally, we mark some conclusions and future work in Section VIII.

II. BACKGROUD AND PROBLEM STATEMENTS

In this section, we briefly describe overviews of Web Integration Model and then we summarize the related OPC techniques, which are mainly used in our work. Finally, the problem statements are carefully discussed.

A. Architecture of Web Integration

A general architecture of Web integration consisting of three layers is shown in Figure 1 [7]. The lower layer provides data or events from automation devices to controller level of an automation system. The upper layer

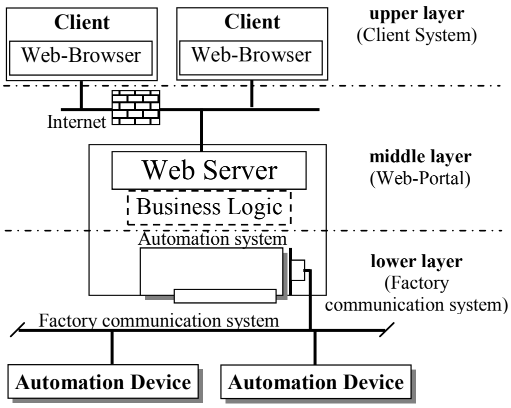

Figure 1. A generic architecture of Web Integration.

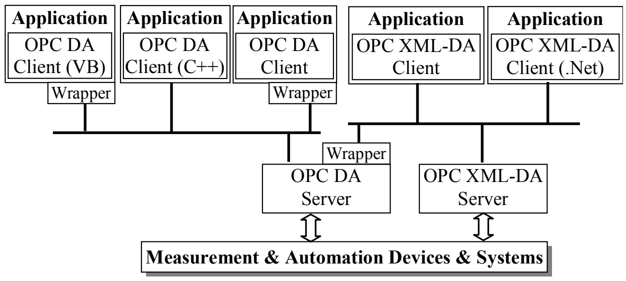

Figure 2. The OPC DA and OPC XML-DA provide plant floor to manufacturing enterprise integration.

is based on standard IT technologies such as Client-Server model, using Web Server as data source and Web browsers as clients. The middle layer normally contains the functionalities of business logic and performs as an application gateway between the upper clients and lower automation systems. The Web Server can be used to assign information from the automation and control systems to object models that can be accessed via COM/DCOM (Distributed Component Object Model). The most important problem is the clear data mapping between data and Web application because the data have different types and different semantic meaning. Using Web technologies for slow process control systems, it means integrating a multitude of different technologies.

B. Overviews of Related OPC Techniques

The OPC Foundation is an independence, non-profit, industry trade association comprised of more than 300 leading process-industry-companies worldwide. In this section, we provide some overviews of the related OPC technologies. Firstly, the OPC DA specification defines a set of standard COM objects, methods, and properties that specifically address interoperability requirements for the factory floor automation, process control, and condition monitoring applications, etc. [8]. The OPC DA leverages DCOM allowing the Client-Server applications to access the plant floor via an Ethernet network distributed across the manufacturing enterprise. But, OPC DA applications are only compatible with cooperating applications based on Microsoft Platforms.

Secondly, OPC XML-DA defines a new specification to move the same type of plant floor data as the existing OPC COM-DA based products [3]. This provides vertical integration between the plant floor and the condition, monitoring, maintenance, etc. by using XML, HTTP, and SOAP industry standards. The OPC XML-DA provides better connectivity and interoperability for production management and enterprise applications such as MES, ERP, CMMS, EAM, and plant optimization that need to access the data from the plant floor. In addition, it is complementary with products based on existing OPC DA Specification. It was specifically designed to allow the existing OPC COM-DA based products to be wrapped by the OPC XML-DA Interfaces and in effect support both interfaces from the same OPC Server. It is as a standard Web Service Interface for reading and writing data from and to plant floor automation systems [3]. The OPC DA and OPC XML-DA provide plant floor to manufacturing enterprise integration as shown in Figure 2.

Thirdly, OPC Complex Data will provide a full way for the OPC Clients to read and decode any type of data from the hardware I/O devices on the plant floor [4]. Complex data mean that an OPC Item is defined as a structure. The item includes read-only information, run-time status and writeable control points. Actually, the complex data consist of complex data items that can include non-structured items, structure items, XML data, OPC Binary, integer, etc. [4]. The OPC Complex Data specification defined two type systems that provide this level of capability, *XML Schema* and *OPC Binary*.

Finally, OPC DX (Data eXchange) has well-defined objects that are based on OPC COM-DA and XML-DA objects [12]. The OPC DX basically addresses the simple mechanism for moving data between source and target (*i.e.*, destination). It sets out the rules associated with the when, what, and how of moving the data between end points. Moreover, it defines for dealing with exception conditions, the data value to be written to the target or maintained at the target when good data are not available from the source. The OPC DX adds some key extensions by leveraging the OPC DA and the OPC XML-DA standards to exchange data horizontally between peer level OPC applications. The OPC DX also extents data access to enable server-to-server data exchange during run-time and independent of the real-time applications supported by the Ethernet networks [13].

C. Problem Statements

As a big step for process control running on various platforms, the OPC Foundation has defined a new XML-DA to allow wrapping the existing OPC DA that is widely successful standard based on COM/DCOM. Thus, PLC (Programmable Logic Control), DCS (Distributed Control System), HMI (Human Machine Interface) and other factory-floor software vendors use the OPC standards to compatibility, reliability, and interoperability.

Nowadays, there are more than the thousands of the existing OPC DA products and the new OPC XML-DA products that need to fully aggregate into a system, which allows the complex data are exchanged in vertical and horizon. In addition, this system should be configured by using remote configuration clients through the Internet. Another requirement is to allow that the OPC Clients access to both the DA Servers and XML-DA Servers.

The OPC XML-DA solution provides more advantages that have been represented in Section II-B. However, the OPC XML-DA solution has a big disadvantage that is to use the XML textual data representation [3], [6]. The XML textual data representation causes much more network traffic to transfer data. The OPC XML-DA requires XML messages to be very descriptive about the data being transferred. For example, instead of "0.555" the record *<value xsi:type = "xsd:float">0.555</value>* is sent, meaning that the bandwidth is increased about six times or even more [6]. Moreover, data alignments are often required to transport data in native representation. The OPC Complex Data based on using the XML data presentation also requires big amount of memory and high intensity of memory management operations. In addition more CPU resources for transformation between native data representation and XML are also required [9]. Besides, the requirement of converting the data between the XML textual data representation and OPC Binary data representation make it to determine the required size of the memory buffer to hold the whole data. A further issue is the present unavailability of XML versions of the OPC HDA (Historical Data Access) [10] and the OPC AE (Alarms and Events) Specifications [11] that are typically required in scientific experiments.

Alternatively, the abilities of reusing and upgrading the systems are important factors. They will make the cost of an application is reduced as well as possible. In addition, the compatibility and interoperability of new system with the existing systems and middleware are strictly required. Thus, the clear design and implementation as frameworks also require us to guarantee in the application design.

III. Design and Implementation

In this section we considerately introduce a novel framework for design and implementation of distributed data acquisition and monitoring system (*i.e.*, *Universal DA Server*). This novel framework is really flexible when applying to real industrial applications. The frameworks for the aspect design and the module implementation of Universal DA Server are represented in Section III-A and Section III-B, respectively. Finally, the discussions and evaluation of our system are exposed in Section III-C.

A. Framework for Aspect Design

In control and monitoring system today, there are more than the hundreds or thousands of the existing OPC DA products and new OPC XML-DA products; OPC XML-DA was specially designed to allow the existing OPC DA products to be wrapped by OPC XML-DA Interfaces. So we are expecting a gateway application for the OPC DA Servers, OPC XML-DA Servers, and other OPC-based gateways that this gateway application provides a sound base for new application and an easy migration path for the thousands of the OPC DA products in use today. Therefore, we propose a *Universal DA Server* based on

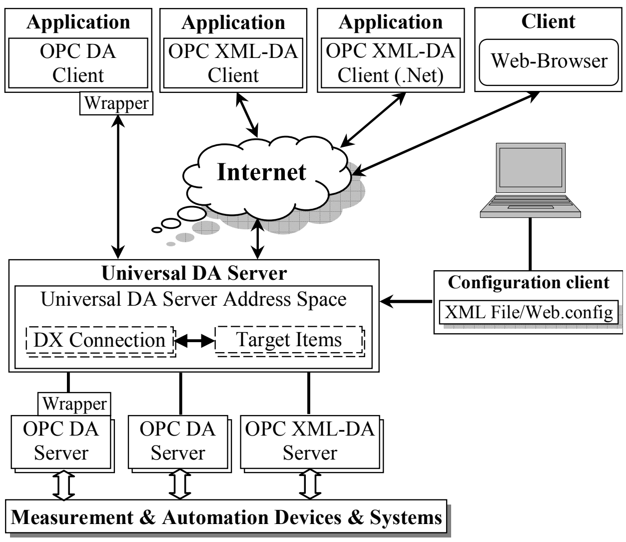

Figure 3. A model of distributed data acquisition and monitoring systems based on OPC techniques.

the OPC DX Specification that can allow integrating the OPC DA Servers, OPC XML-DA Servers, and Gateways to exchange data between them. And, the Universal DA Server also allows accessing the data from the XML-DA Clients. The roles of Universal DA Server to aggregate multi-OPC Server are shown in Figure 3. The Universal DA Server provides methods to allow that the XML-DA Clients easily access to both the OPC DA Servers and the XML-DA Servers. Moreover, it is flexible to configure the access rights such as write-only, read-only, and read-write from the configuration client (*i.e.*, remote client).

With the Universal DA Server, the multiple OPC DA Servers and OPC XML-DA Servers can be mapped as data sources as shown in Figure 3. Data from source servers are handled according to the OPC DX Standards. By basing on configuration clients, we can configure the Universal DA Server to allow clients to read/write data from/to the OPC Source Servers (*i.e.*, source servers).

In addition, the data from the hardware I/O devices on the plant floor are complex data types, so that OPC Items of all configured servers are mapped into the Universal DA Server as complex data items based on the solution mentioned in [2]. Corresponding to this requirement, we design and implement the Universal DA Server that can support the OPC Complex Data Specification. Hence, the Universal DA Server not only resolves the problems to aggregate the OPC DA and OPC XML-DA Servers, but also permits the OPC Clients to read and decode any type of data from the hardware I/O devices.

The Universal DA Server is responsible for managing the data on a DX Connection from source to target. In general, it subscribes to data from the source and copies it to the target when it is received. The flow diagram of updating data to target items from source servers is shown in Figure 4. In addition, our framework-based system is responsible for maintaining the connection to source servers. When the connection to a source server is lost, either during startup or runtime, the Universal DA

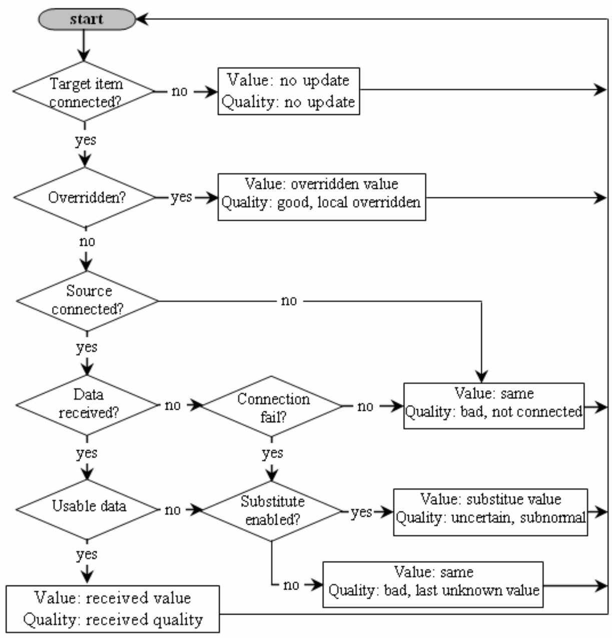

Figure 4. The flow diagram of updating data to target items.

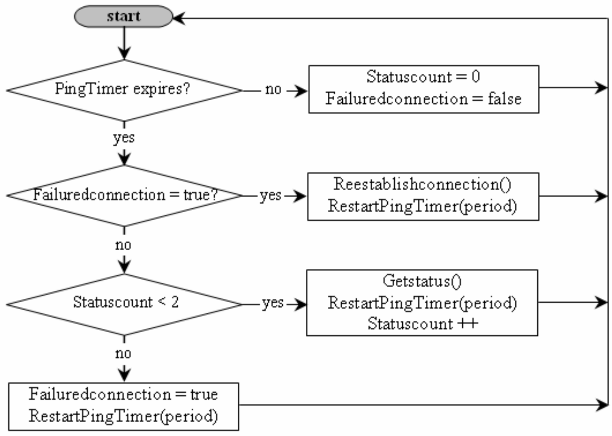

Figure 5. The flow diagram of recovering a connection failure between the Universal DA Server and source servers

Server tries to reestablish the connection in the minimum time between two attempts to reestablish a connection. If Universal DA Server has not received data from the source server for any DX Connection in a period, it checks the status of connection to the source server using *Getstatus* method, depending on the type of the source servers. After three times of expiration without receiving the data or a *Getstatus* respond from source server, the Universal DA Server assumes that the connection has failed and attempts to reestablish it. The flow diagram of recovering a connection failure between the Universal DA Server and source servers is shown in Figure 5. By carefully exposing the system aspects as framework, this framework provides more architecture, functionalities, and interfaces for understanding and building distributed control and monitoring systems easily.

B. Framework for Module Implementation

The Universal DA Server can access to multiple OPC DA Servers and OPC XML-DA Servers and it can be dynamically configured according to the OPC DX. The OPC Items of all configured OPC Servers are mapped into single address space that can be structured in any suitable way. Both *read* and *write* accesses are supported by the Universal DA Server. As we aforementioned, the Universal DA Server aggregates multiple OPC Servers as data sources that allow exchanging the complex data among them. The data are acquired from the source server depends on the types of the source servers such as OPC DA and OPC XML-DA. In general, the mechanism is referred as a subscription. Subscribing to the data is supported by the *Callback* method in OPC COM-DA and by the *Subscribe* service in OPC XML-DA.

The Universal DA Server creates one ore more groups of each source server for OPC DA case; and with defined groups it adds data items to them to indicate the source server which items to access. In OPC XML-DA case, the Universal DA Server uses *Subscribe* and *Subsciption-PolledRefresh* services to acquire sources' data. The *Subscribe* service allows the client to define a single operation and a set of items for source server to access.

The design and implementation of the Universal DA Server consisting of three components are shown in

Figure 6. To provide the module implementation as a framework, we represent the following components.

1) COMModule. This module exposes OPC DA-Interface and DX-Configuration Interface implemented as COM-Interface. It provides methods and interfaces for communication between the COM-based OPC Clients and the Universal DA Server.

2) SOAPModule. This module handles SOAP requests and exposes both the OPC XML-DA Interface and DX-Configuration Interface based on the use of WSDL (Web Service Description Language) [14].

3) DXModule. This module contains major part of DX Server's functionalities and it marshals data to and from the OPC Servers. It can create new connections, remove connections, and modify the status of each connection. The Universal DA Server requires that DX-Connection can be modified and controlled by the SOAP and COM Clients simultaneously. Therefore, it requires solving the inconsistencies and synchronization problems.

The design of the modules for the Universal DA Server to aggregate the multiple OPC Servers into the system that can exchange the complex data between OPC DA Servers and OPC XML-DA Servers is shown as in Figure 7. To support the OPC Clients to read and decode any type of data from the plant floor, *OPCComplexData* module is used in the Universal DA Server. This module

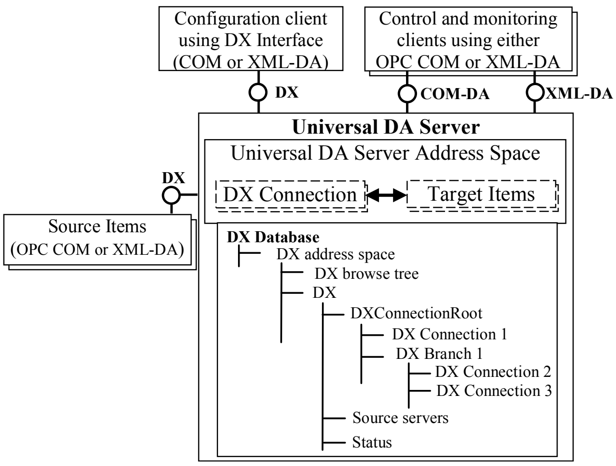

Figure 6. The design of Universal DA Server to support multiple OPC DA Serverss and OPC XML-DA Servers.

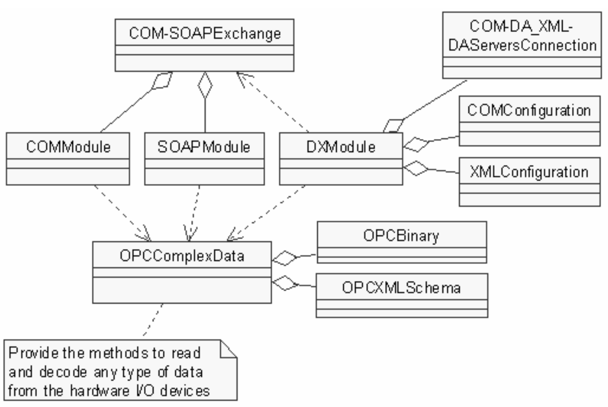

Figure 7. The framework for module design of Universal DA Server.

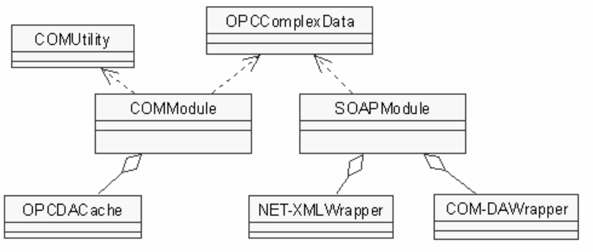

Figure 8. The class diagram of the COM and SOAP modules.

provides methods to convert and represent the complex data that complex data types are defined as dictionaries.

Besides, the Universal DA Server based on OPC DX Specification that provides a means to exchange the data between OPC DA Servers and OPC XML-DA Servers. Thus, as shown in Figure 7, the *COMSOAPExchange* aggregates the *COMModule* and *SOAPModule*, and it is used by the *DXModule*. This module handles SOAP requests from XML-DA Clients and it also supports to exchange data between OPC DA Server and OPC XML-DA Server.

Two modules such as *COMModule* and *SOAPModule* as shown in Figure 8 provide mechanisms and methods for processing the COM Objects and the SOAP Objects depending on the OPC Clients (*i.e.*, either COM-DA Client or XML-DA Client). The *SOAPModule* is used not only for the XML-DA, but also for the existing COM-DA to be wrapped by using the wrapper. Therefore, this module is composed of the *NET-XMLWrapper* and the *COM-DAWrapper*. The *OPCDACache* that provides a means to resolve the large control network problems is aggregated in the *COMModule*.

Thereby exposing the implementation of the modules as framework, this provides a bottom-to-top way for understanding and implementing the system. In addition, our framework also provides a seamless implementation for readers or programmers. It is easy to reuse, maintain, upgrade, and deploy to any general application based on OPC techniques. Hence, our framework-based system is clearly designed as Web Services, which allow the clients running on the Microsoft Platforms and non-Microsoft Platforms to access by using the XML leverages.

C. Discussions of Universal DA Server

In software engineering, the system should be designed as frameworks that seamlessly provide a complete way for reusing, maintaining, improving, and deploying to real applications are needed. This framework based on the clear design of aspects, architectures, functionalities, and components. The application framework normally used content model to request data source and data formats for their values that have to be read out of the factory communication systems and their components by basing on several standards. So, the XML-based content model in [15] that consists of a distributed set of XML files, schemas, and transformation rules is an example of a framework. In this paper, we introduce a framework for building the distributed data acquisition and monitoring systems that are widely used in industry today. Our framework is proposed as the outlines of the interfaces, architectures, and functionalities. Corresponding to this goal, the framework for system aspect design and module implementation is constructed. With our framework, the abilities of reusing, upgrading, and deploying to industrial applications are proposed and it will decrease the cost, time, etc. for developing and deploying.

According to our design and implementation, the Universal DA Server provides a bottom-up mechanism to aggregate more than the hundreds or thousands of the existing OPC COM-DA products in use today with new OPC XML-DA products into a system and to exchange the complex data between the OPC Servers in horizon. This allows the OPC Clients to read and decode any type of data from the hardware I/O devices on the plant floor.

In addition, our Universal DA Server has an advantage is ability of running on various platforms (*i.e.*, Microsoft Platforms and non-Microsoft Platforms), meaning that it is defined as Web Services. Moreover, it also ensures the OPC COM-DA components that have been successfully developed and tested to be reused as effectively as possible. Finally, the abilities of easily configuring and monitoring our systems based on the Web Service Interface and the COM Interface are provided.

IV. Data Representation

When determining our distributed data acquisition and monitoring system for running on various platforms, data transferred on the Internet will be packaged in the SOAP message. However, the SOAP messages with pure XML textual data representation normally have a big size, so that they make a system is not optimal. In addition, the *Subscribe* and *SubscribePolledRefresh* mechanisms of data subscription used in OPC XML-DA are inefficient in utilizing a full network bandwidth. Thus, the proposal of using the binary data representation is exposed in Section IV-A. The *XML Libraries*' comparisons are continuously discussed in Section IV-B.

A. Binary Data Representation

The only fast possibility to transmit the large amounts of data is to use the binary data representation rather than the XML textual data representation as aforementioned in Section II-C. In addition, different platforms and even different compilers of the same platform use the different binary representations such as different bytes order for representing multi-byte data, different floating point formats, etc. Thus, universal standards should be chosen to transport data between different platforms. Therefore, we must investigate and compare the current available binary encoding standards. The fundamental approach to resolve the bandwidth problem is using the binary data representation, which is integrated into XML such as BXML [16], BXSA [17], etc. Several proposals are available to satisfy these conditions such as SOAP message [18] with attachment or the HTTP message using XLink (XML Linking Language) [19]. The Universal DA Server will respond to the XML-DA Clients the SOAP message with all the data replaced by XLink references. To incorporate binary data into the SOAP message, the WS-Attachment technology [20], [21] is used. In addition, XLink is used to link different parts of compound message together. Also, addressing of multi-group can be done in XLink.

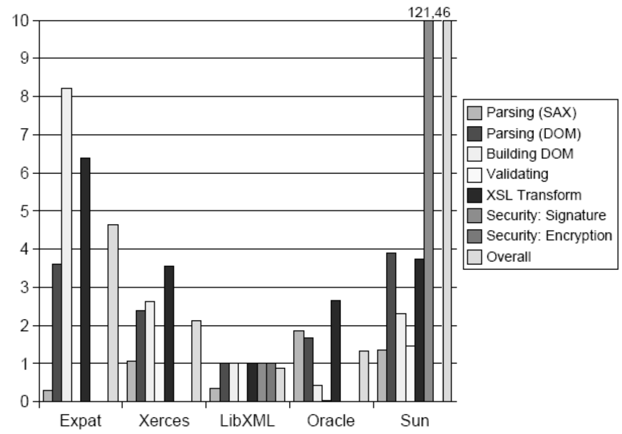

Figure 9. Benchmark results of five different library combinations. The right most bar shows the overall performance of all processing stages. A blank bar indicates that the service is not available.

Our Universal DA Server is proposed with its two contrary goals: (1) maximizing the compatibility with different architectures, operating systems, and third-party applications and (2) achieving the best possible performance when exchanging big amounts of data over the Internet. Our framework-based system is accomplished without any restrictions based on various platforms to guarantee the first goal. Moreover, high data rates are supported by using the SOAP messages with attachment of the binary data representation. Thus, the performance of our system is better than with the pure OPC XML.

B. Comparisons of XML Libraries

When designing and implementing an application, we have to carefully resolve the existing problems of weak ground. If there is new software available made by different manufacturers, the best thing to do is to use the benchmarks. To achieve a better basic for a decision to select the best XML library, several benchmarks were performed to evaluate the present XML libraries. There are many available XML library benchmarking projects such as XMark [22], XML Benchmark [23], SAX Parser Benchmark [24], etc. However, they are not very well applicable for OPC XML-DA. In addition, they tested only one or two aspects of XML processing, with some predefined sequences of XML data. To provide fast and reliable OPC XML-DA or Universal DA Server solution, a fast multiplatform XML toolkit is required to process different types of XML files at high speed. Therefore, we have to investigate some supports including XML-binary Optimized Packing [25], XML Encryption [26], XML Signature [27], [28], XML Transformation [29], and some other. The significant benchmark results of five different libraries are shown in Figure 9 [30]. On some platforms, multiple compilers are available and should be compared, since the performances of the XML Libraries are very important for the overall system performances. The performance tests can be divided into the following phases of XML processing such as Schema validation, XSL Transformation, XML Security, etc. As shown in Figure 9, the *LibXML* is effectively high performance and it is selected to apply to our system.

V. Connection Between Universal DA Server and Database

A framework for building distributed data acquisition and monitoring systems should guarantee the connection between framework-based system and database for extending in the future. To provide an open and flexible system for implementing and aggregating the OPC HDA [10] into our framework, the connection between the Universal DA Server and database is investigated and provided. Standard SQL (Structure Query Language) as basic query language for database searches in many cases is not efficient. Consequently, each database has its own extension to SQL such as PL/SQL for Oracle, PL/pgSQL for postage-SQL. However, with each SQL solution is not independent platform. In fact, we are expecting to achieve a solution with independent platform by using XQuery. But in large applications the XQuery is still too slow and too memory consuming as shown in Figure 10 [6]. When the Universal DA Server is defined as a Web Server and it supports to store and exchange the data from the Web Server to the OPC Clients, it should be designed for expanding the connection between the Universal DA Server and database. To build a full system based on OPC techniques, we will investigate the ability to extend and integrate our system's framework when needfully adding new OPC Specifications-based products or combining the existing OPC-based products. However, the situation of XQuery will change rapidly in near future due to efforts put into the further development of XML.

Thus, as shown in Figure 10, when aggregating the database to our framework-based system (*i.e.*, Universal DA Server) for supporting the ability of storing data or events from the hardware I/O devices on the plant floor and the ability of exchanging them to clients, our system needs to carefully investigate. In addition, our framework will be proposed an optimal procedure of XML to access to the database. This procedure should be to start with the product depending query language XSU and to migrate to XQuery.

VI. Security Consideration

The security issues are very important for application when using both Ethernet and the Internet. In order to provide the abilities of preventing all kinds of attacks from the Internet environments, six common goals such as *authentication*, *authorization*, *integrity*, *confidentiality*, *audibility*, and *availability* have to be strictly guaranteed. In order to achieve the required security criteria, the concepts and solutions developed for General IT system have to be applied.

Appropriate technologies like encryption, SSL (Secure Socket Layer) technology, HTTP-S (Secure HTTP), certificates and digital signature should be used. The security solution for our framework-based system has to guarantee the security to both the COM-DA and XML-

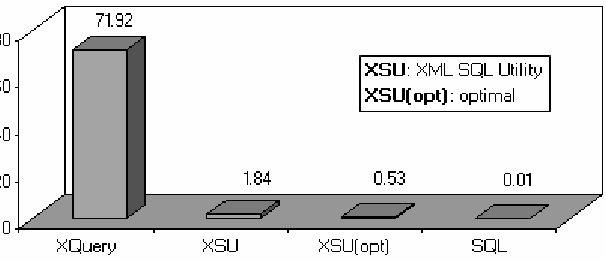

Figure 10. The comparison between query languages. SQL still is the fastest access method but XSU is feasible for querying in database.

DA. In addition to reduce the bandwidth problems, the described solutions of improving performance raise new problems that are security problems. In the standard cases, the HTTP-S can be used to protect data. However, for multicasting data connections HTTP-S is not available and some other mechanisms must be used. A proposal is to use an authentication server that will use SSL private/public keys for authorization and then generates symmetric session keys. Moreover, it is also proposed to use the internal XML security approach for the control connections instead of the HTTP-S protocol, being described in XML Encryption [26], XML Signature specifications [27], [28], and XML Decryption [31]. This will provide the following advantages over the HTTP-S approach:

- Better compatibility with third party Internet software.
- Some proxy servers lack of HTTP-S support.
- Some simple and fast Web servers lack of HTTP-S supports.
- More control over client authentication since access control is shifted from OPC Servers to Universal DA Server.
- Enhanced performance since only parts of an XML document may be secured.

The XML Distributed signature approach, which was represented in [32], should be used to guarantee the security with the OPC XML-DA. Because this solution is really optimal to reduce the data transfer between servers and clients, and ensures thin clients (*i.e.*, it is no necessary to build any special environment for XML Signature creation on the clients side). With the OPC COM-DA, the security based on DCOM Security that distinguished between connection security such as per call security and per packet security is exposed in [33], [34]. Moreover, access control and launch permission can be set for individual DCOM objects, groups of objects, or all objects on the OPC Servers. They are used for security in the COM-DA Interface. Another aspect, the security concepts with two sections, one for the Intra/Internet and one for the field level are proposed in [35]. As we already discussed, the security in our system has to ensure to absolutely guarantee for both the Ethernet and Internet. Consequently, several security solutions that are chosen to use in our systems have been discussed. However, there are no currently authentication protocol designs for the factory IT environments that strictly require security guarantees as well as expectation.

VII. Analyses of the Framework and System Performance

Our Universal DA Server is proposed as framework corresponding to the outlines of the architectures, interfaces and functionalities. These features are obviously advantages of our system when reusing, maintaining, customizing, and deploying to real industrial applications based on OPC techniques. With our framework, the developers and programmers can easily understand the architectures, interfaces, functionalities, components, and modules that need to build and implement.

As we mentioned our novel framework for building the distributed data acquisition and monitoring systems, *e.g. Universal DA Server*, the framework-based system allows easily aggregating the existing OPC COM-DA products in use today with new OPC XML-DA products into a system that the complex data are not only vertically exchanged between the OPC Servers and OPC Clients, but also horizontally exchanged between the OPC Servers (*i.e.*, DA Servers and XML-DA Servers). Thereby using the attachment of the binary data representation in the SOAP messages, the required size of the SOAP messages transferred over the Internet is reduced about more than six times, meaning that bandwidth of the network and system resources are truly improved much.

In addition, the ability of running on various platforms indicates that our system is very flexible to apply to the Internet environment where the industrial systems now are widely increasing growth. This also means that the interoperability of our system is compatible with different operating systems, middleware, etc. Moreover, the ability of supporting both the XML-DA Interface and COM-DA Interface makes that our system has strength of position in the control and monitoring systems today to resolve the complex requirements such as operating systems, multi-interface supports, compatibility with old versions, etc.

When thinking about the performance of our system, the fast real-time data updates are difficult to meet in the Universal DA Server that is based on XML leverages and the Internet technology; because real-time applications must receive timely data updates in milliseconds or second. However, the business applications such as MES, ERP, and CMMS require data updates that are at hourly frequency at best or several minutes, and usually update every shift. Furthermore, remote monitoring applications require every slow updates. Therefore, the performance of our framework-based system is to reach an acceptable performance for several real industrial applications.

Alternatively, with the development of the network technologies the high speed Ethernets (*e.g.*, 100Mbps, 1Gbps, etc.) are widely used in LAN, WAN, and the bandwidth resources are increasing fast because of the development of the switch technologies. Thus, the system performance will be improved as well as possible.

In brief, our system is designed and implemented as framework that is easy to reuse, upgrade, and deploy to real applications. By using the SOAP messages with the attachments of the binary data representation, so that the performance of our system is significantly improved. The most important factor of our system is to allow easily aggregating more than the hundreds or thousands of the existing OPC DA products and new OPC XML-DA products into the Universal DA Server, which permits the complex data exchange in vertical and horizon. However, our system should be carefully investigated about the compatibility, interoperability, reliability, and acceptable performance before applying to industrial systems with a very large set of data and a short period of time.

VIII. Conclusions and Future Work

In this paper, we have already introduced a novel framework for design and implementation of distributed data acquisition and monitoring systems based on OPC techniques. Our framework-based system allows easily aggregating the existing OPC COM-DA and OPC XML-DA products into a system that permits to exchange the complex data between OPC Servers (*e.g.*, DA Servers, XML-DA Servers) also, the OPC Servers and the clients. This solution provides a sound base for new applications and an easy migration path for the hundreds or thousands of the existing OPC DA products in use today.

Thereby exposing our system as framework consisting of several sub-frameworks for the aspect design, module implementation, data representation, and security issues, our framework provides a complete bottom-up way that is easy to reuse, maintain, upgrade, and deploy. With our framework, the developers or programmers can easily understand the architectures, functionalities, modules, and requirements to develop their systems.

To overcome the performance limitations of the XML textual data representation, we have proposed to use the binary data representation in the Universal DA Server. Besides, we have investigated and compared the *XML Libraries* to choose XML technology as a candidate in our system. So the *LibXML* should be used to resolve the system requirements (*e.g.*, any platform, an acceptable performance, etc.). Moreover, our system can run on various platforms (*i.e.*, Windows Platforms and non-Windows Platforms). To configure the Universal DA Server, we can use remote clients that really ensure flexibility and optimization of the system. In addition, the security aspects are carefully discussed in our work for choosing good security solutions. These aspects provide seamless security solutions for systems transferring data over the Internet and Ethernet. They also provide more security information for technical-level readers.

In addition, the analyses of our framework and the system performance are also considerately discussed. These analyses indicate that our framework has a good design and the performance of our system is acceptable for many control and monitoring systems nowadays. Therefore, our system is truly feasible to apply in process monitoring and control systems. However, the system performance based on the SOAP message with the XML leverages is significantly dependent on the size of a SOAP message transmitted on the network. Therefore, a variability of our system performance related to the size of the data in a SOAP message with the attachment of the binary data representation will be carefully investigated in the future work.

ACKNOWLEDGMENT

The authors would like to thank the Ministry of Commerce, Industry, and Energy and Ulsan Metropolitan City, which partly supported this research through the NARC (Network-based Automation Research Center) at the University of Ulsan. The authors wish to thank the reviewers for their valuable comments and suggestions.

REFERENCES

[1] D. W. Holley, "Understanding and Using OPC for Maintenance and Reliability Applications," *Journal of the IEE Computing and Control Engineering*, pp. 28-31, February-March 2004.

[2] V. V. Tan, D. S. Yoo, and M. J. Yi, "Design and Implementation of Web Service by Using OPC XML-DA and OPC Complex Data for Automation and Control Systems," *Proceedings of the 6th IEEE International Conference on Computer and Information Technology*, CIT'06, p. 263, September 2006.

[3] The OPC XML Data Access Specification version 1.01, [Online]. Available: http://opcfoundation.org/, December 2004.

[4] The OPC Complex Data Specification version 1.0, [Online]. Available: http://opcfoundation.org/, December 2003.

[5] S. Chilingargyan and W. Eppler, "High Speed Data Exchange Protocol for Modern Distributed Data Acquisition Systems based on OPC XML-DA," *Proceedings of the Real-time Conference, 14th IEEE-NPSS*, pp. 352-356, June 2005.

[6] W. Eppler, A. Beglarian, S. Chilingarian, S. Kelly, V. Hartmann, and H. Gemmeke, "New Control System Aspects for Physical Experiments," *IEEE Transaction on Nuclear Science*, vol. 51, no. 3, pp. 482-488, June 2004.

[7] M. Wollchlaeger, P. Nweumann, and Th. Bangemann, "Web Service for Remote Maintenance of Fieldbus based Automation Systems," *Proceedings of the IEEE International Conference in Africa*, vol. 1, pp. 247-252, October 2002.

[8] The OPC Data Access Specification, [Online]. Available: http://opcfoundation.org/, December 2004.

[9] F. Bustamante, G. Eisenhauer, K. Schwan, and P. Widener, "Efficient Wire Formats for High Performance Computing," *Proceedings of the ACM/IEEE Conference on Super Computing*, p. 64, November 2000.

[10] The OPC Historical Data Access Specification, [Online]. Available: http://www.opcfoundation.org/, December 2003.

[11] The OPC Alarms and Events Access Specification [Online]. Available: http://www.opcfoundation.org/, October 2002.

[12] The OPC Data eXchange Specification version 1.0. [Online]. Available: http://www.opcfoundation.org/, March 2003.

[13] H. Xiaohong and H. Sunhong, "OPC DX and Industrial Ethernet Glues Fieldbus Together," *Proceedings of the Conference on Control, Automation, Robotics, and Vision*, vol. 1, pp. 562-567, December 2004.

[14] Web Services Description Language (WSDL) version 2.0, March 2006, [Online]. Available: http://www.w3.org/TR/wsdl20/

[15] M. Wollchlaeger and T. Bangemann, "XML based Description Model as a Platform for Web-based Maintenance," *Proceedings of the IEEE Conference on Industrial Informatics*, pp. 125-130, June 2004.

[16] C.S. Bruce, Cubewerx Position Paper for Binary XML Encoding, [Online]. Available: http://www.cubewerx.com/main/HTML/-Binary_XML_Encoding.html

[17] K. Chiu, T. Devadithya, W. Lu A. Slominski, "A Binary XML for Scientific," *Proceedings of the 1st International Conference on e-Science and Grid Computing*, 2005.

[18] SOAP Version 1.2 Recommendation 2003/6/24. [Online]. Available: http://www.w3.org/TR/SOAP

[19] XLink (XML Linking Language) version 1.0, 2001/6/27. [Online]. Available: http://www.w3.org/TR/xlink/

[20] SOAP message with Attachments, December 2000, [Online]. Available: http://www.w3.org/TR/SOAP-attachments

[21] J. H. Gailey, "Using Web Services Enhancements to Send SOAP Messages with Attachments," February 2003. [Online]. Available: http://msdn2.microsoft.com/en-us/library/ms996944.aspx.

[22] XMark, an XML benchmark project, [Online]. Available: http://monetdb.cwi.nl/xml/index.html

[23] XML Benchmark [Online]. Available: http://www.sosnoski.com/opensrc/xmlbench/results.htm;

[24] SAX Parsers Benchmark [Online]. Available: http://piccolo.sourceforce.net/bench/html

[25] XML-binary Optimized Packing, W3C Recommendation, 2005/01/25, [Online]. Available: http://www.w3.org/TR/2005/REC-xop10-20050125/

[26] XML Encryption Syntax and Processing: W3C Recommendation 2002/10/10, [Online]. Available: http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/

[27] XML-Signature XPath Filter 2.0:W3C Recommendation, 2002/11/08. [Online]. Available:http://www.w3.org/TR/xmlenc-core

[28] XML Signature and Processing: Recommendation 2002/2/12. [Online]. Available: http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/

[29] XSL Transformation (XSLT) version 1.1, 2001/8/24. [Online]. Available: http://www.w3.org/TR/2001/WD-xslt11-20010824

[30] S. Chilingarian, Fast, Multiplatform XML Toolkits Comparison. Functionality, Speed and Memory Usage, [Online]. Available: http://prdownloads.sourceforge.net/xmlbench/features.pdf, http://prdownloads.sourceforge.net/xmlbench/benchmark.pdf

[31] Decryption Transformation for XML Signature: W3C Recommendation, 2002/10/10. [Online].Available: http:www.w3.org/TR/xmldenc-decrypt/

[32] Miyauchi, "XML Signature/Encryption – The Basic of Web Service Security," *NEC Journal of Advance Technology*, vol. 2, no. 1, pp. 35-39, Winter 2005.

[33] OPC Security Custom Interface Specification, [Online]. Available: http://opcfoundation.org/, October 2000.

[34] D. Dzung, M. Naedele, T. P. V. Hoff, and M. Crevatin, "Security for Industrial Communication Systems," *IEEE Journal*, vol. 93, no. 6, pp. 1152-1177, June 2005.

[35] A. Treytl. T. Sauter, and C. Schwaiger, "Security Measures in Automation Systems – a Practice-Oriented Approach," *Proceedings of the 10th IEEE Conference on Emerging Technologies and Factory Automation*, vol. 2, pp. 847-855, September 2005.

About the Authors

Vu Van Tan was born in Haiduong province, Vietnam, in 1981. He received the Eng. degree in Information Technology from the Hanoi University of Technology, Vietnam, in 2004. He has worked as design and analysis engineer in KhaiTri Software Company, Vietnam, for one year. He is currently a Ph.D student at the School of Computer Engineering and Information Technology, University of Ulsan, Republic of Korea. He joined in the Applied Software Engineering Lab, University of Ulsan, in 2005. His main interests are software engineering, software for automation systems, Internet technologies for industrial automation systems, and real-time communication systems.

Dae-Seung Yoo received the B.S. and M.S. degrees in Computer Engineering and Information Technology from the University of Ulsan, Republic of Korea, in 1998 and 2001, respectively. Now he is a guest professor in the School of Computer Engineering and Information Technology, University of Ulsan. He is a member of KIISE, KIPS, and IEICE. His main interests are software engineering, software for automation systems, and Internet technologies for industrial automation systems.

Myeong-Jae Yi received the B.S. degree in Computer Science from the Seoul National University, Republic of Korea in 1987. He also received the M.S. and Ph.D degrees in Computer Science from the Seoul National University in 1989 and 1995, respectively. He was a part-time lecturer at the Department of Computer Science of the Seoul National University and the Sookmyung Women's University from 1991 to 1996. He is currently a professor in the School of Computer Engineering and Information Technology, University of Ulsan, Republic of Korea.

Prof. Yi is also a vice-Director of NARC (Network based Automation Research Center) at the University of Ulsan and is a member of KIISE, ACM, and IEEE. His main interests are software engineering, software for automation systems, and Internet technologies for industrial automation systems.