Research on OPC UA based on Electronic Device Description[①][②]

Huang, Renjie   Liu, Feng
Faculty of Computer & Information Science, Southwest University, Chongqing China 400715
cqhrj@163.com

Abstract- **On the basis of introducing the characteristics of OPC Unified Architecture and EDD (electronic device description), a framework of OPC UA server based on EDD is proposed to map field devices as nodes in address space of OPC UA server. In the framework the device object model which accords with the information model of OPC UA is presented by extracting the information from device description document. Then aimed at the characteristics of EDDL in HART, the method for transforming DD document into node object is discussed in detail. According to the method above devices of different manufacturers can be integrated into OPC UA server, and the OPC UA server can get more device management information through EDD.**

I. Introduction

In the last ten years, OPC technology is widely applied in the industry automation as factual standard. Recently the development and application are hindered by its following technology characters [1] [2]:

1) Basing on COM/DCOM.
2) No contacts between OPC DA, OPC HDA and OPC A&E.
3) No ability to provide device management information for more applications such as ERP and AMS (asset management system).

With the goal to overcome the above disadvantages, the OPC Foundation released OPC Unified Architecture (UA) specifications. By defining abstract services, it provides a Service-Oriented Architecture (SOA) for industrial applications from factory floor devices to enterprise applications. OPC UA integrates the different flavors of the former OPC technology into a unified address space accessible with a single set of services. In fact, OPC UA is only a general framework for industrial applications. It can be realized by different measures. This paper mainly discusses how to realize the OPC UA server based on EDD.

II. OPC UA AND EDD TECHNOLOGY

A. *OPC UA architecture*

The OPC UA systems architecture models OPC UA *Clients* and *Servers* as interacting partners. *Clients* communicate with *Servers* by C/S request and response, and *servers* can publish notifications to *clients.* Figure 1 illustrates the architecture [3].

OPC UA *client* Communicates with *server* through OPC UA *communication stack* which is used on client-side and server-side to encode and decode message requests and responses. OPC UA API is a link between *communication stack* and the *server* or *client implementation*. Both can be realized by the different technologies and program languages in system independent platform [4].

The *Client* and *server* implementations take charge of OPC UA applications. The client implementation sends request and receives data over OPU UA client API, The server implementation receives OPC UA server API invocation and returns data from address space in which nodes are maps of real objects.

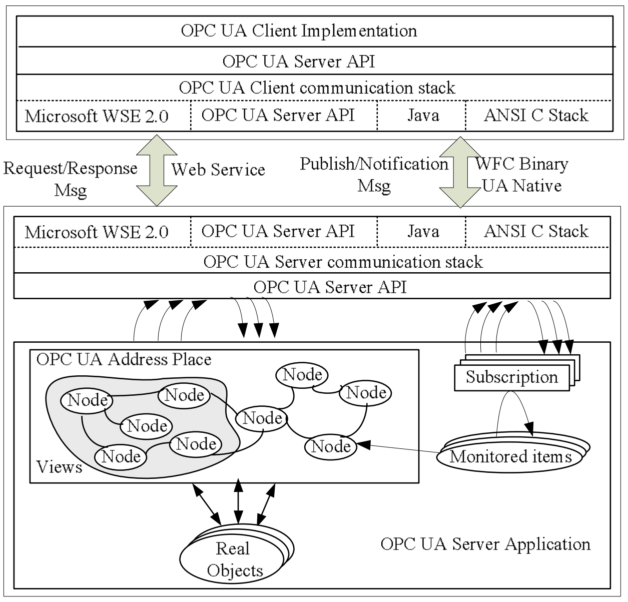

Fig. 1.OPC UA Server Architecture

Address space and service are the key concepts in OPC UA. In address space there are basic objects such *Variable, Object, View and Reference* etc to describe and manage data from underlying system. Fox example a *Variables object* can be used to describe a parameter value of device or process variable value, *Objects* are used to represent systems, system components, real-world objects and software objects. *References* are used to describe the relation between *Variables, objects and views. Views* are used to describe the logical group which can be viewed by client.

An OPC UA service is defined by its request and response messages, thus, it is on the same level called operation in WSDL [5]. OPC UA groups their services into service sets.

① The paper is granted by the project of ChongQing in China (No.CSTC-2007AB2031).
② The paper is granted by the project of university in China (No. SWUQ2006016).

In OPC UA there are the following service sets such as Secure Channel Service Set, Session Service Set, Node Management Service Set, View Service Set, Query Service Set etc. For example, the node management service set allows adding and deleting nodes and references in the address space.

B. EDD and EDDL

EDD was firstly presented in HART protocol in 1992. It was adopted by FF, PROFIBAS and other fieldbus protocols in a short time because of its powerful description function and system independence. And IEC released the IEC 61804 series standards to unify the EDD in 2002. Now EDD is widely applied for field device integration in industrial automation.

The technology to describe an EDD is called EDDL (Electronic Device Description Language). It is a structured and interpretative language for describing device properties [6]. Also the interactions between the devices and the EDD runtime environment are incorporated in the EDDL. It provides a set of language elements such as VARIABLE, MEHTOD and MENU etc for these purposes. An EDD document is as following Figure 2:

DeviceID	MANUFACTURER: company DEVICE_TYPE::PRESSURE
VersionInfo	DEVICE_REVISION:1 DD_REVISION:2
DataDescription	VARIABLE …
CommunicationMapping	COMMAND 1 …
DiaplayStructure	MENU: ShowMeasurementValue …
Method	METHOD: Reset_Device …

Fig. 2 Content and structure of EDD

In the EDDL the elements such as MANUFACTURER, DEVICE_TYPE AND DEVICE_REVISION are used to describe the basic information of device. VARIABLE is used to describe the parameters of device, and alarm information is described by VARIABLE also. COMMAND is used to describe the communication mapping. It relates the VARIABLE to factual command used to read the variable. And the relation between variables can be described by the elements such as UNIT, WRITE_AS_ONE etc. The MENU is used to organize the VARIABLE and METHOD and describe their display structure, by the means they are expediently viewed in the DD application. METHOD is used to describe the configuration and diagnosis functions, EDD application can diagnose and maintain device by calling the METHOD. Moreover In EDDL there are other elements such as COLLECTION and ARRAY to organize a lot of variables and methods.

Usually manufacturers develop the DD in DD IDE (integration development environment) with EDDL. At last the DD is compiled into binary document. It can be embedded in device, or it can be provided to user with device. In conclusion, the EDD is the system independent technology with powerful description functions. It covers the following aspects [7]:

1) Description of the device parameter,
2) Support of parameter dependencies,
3) Logical grouping of the device parameters,
4) Selection and execution of supported device functions,
5) Description of the device parameter access method.

C. Combination of OPC UA and EDD

Both EDD and OPC UA are the system independent technologies for system integration. Just EDD is used for horizontal integration of field devices and OPC UA is used for vertical integration from system floor to enterprise application. But their information models describe the device information similarly. OPC UA specifies a general framework for system integration, and it provides a data exchange across independent interfaces and services. EDD is the technology used to field device integration. It can map the field devices of different manufactures to application in control system. Accordingly it is profitable to apply EDD technology to OPC UA server design and development. Field devices can be expediently integrated into OPC UA server. Moreover EDD technology can provide not only the process variable values and alarm information but also rich device management and maintenance information for OPC UA client. It is a solution that provides open standards to extend interoperability from device to enterprise applications. Figure 3 shows the framework of plant automation system OPC UA and EDD is applied together.

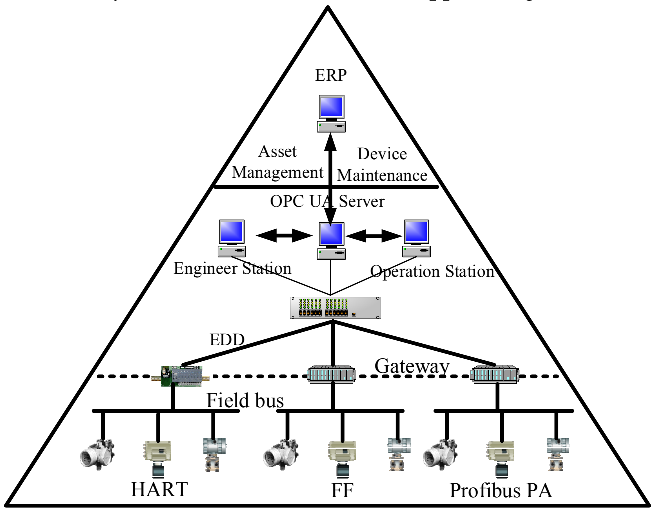

Fig. 3 System integration based on EDD and OPC UA

III. OPC UA SERVER BASED ON EDD

A. Goal

The goal of combination OPC UA and EDD is to provide a low-cost system integration solution by extracting DD document information and transform it to device object in OPC UA. When the OPC UA servers are equipped with

EDD interpreter, the OPC UA servers can access the filed device and get the information from field device. And the device object as node in OPC UA, its component and detailed information can be extracted form the DD document by designing a universal software module. It is not necessary to develop drivers to access data, and address space can be designed easily because node model can be extracted form DD document. By the means, development cost of OPC UA servers can be reduced, and development periods can be shortened. In the following sections this paper will present the transformation models and measures from DD to device object in OPC UA in detail.

B. Frameworks of OPC UA server based on EDD

Although IEC released the IEC 61804 series standards, the device description technologies of FF, HART and PROFIBUS differ from each other in some way. It has no effect on the application of OPC UA based on EDD for system integration, because OPC UA servers at first are independently developed for HART, FF and PROFIBUS, then an aggregating server aggregates them and provide their information. This paper will discuss how to mapping the field device to OPC UA server through EDD only in HART protocol.

The main objective of realizing the OPC UA server based on EDD is to transform the device information described by EDDL into OPC UA information model. Consequently, device information resides in the address space as a Node. client application can access device information and method. EDD interpreter is the key part in the process, it parses the device description document and gets the detailed information of device and the elements in EDD document, which mainly contain the base device information, VARIABLE and relation of VARIABLE, and METHOD. Then it transforms a DD document into *Node* composed of *Variable object, Reference, Method* and *Views* of OPC UA in term of rules, At last it is necessary to create and organize it into OPC UA address space by node management. Figure 4 shows the total process.

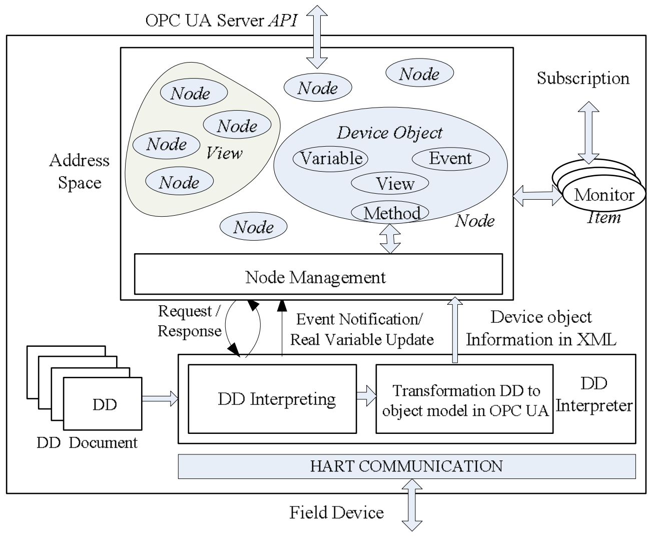

Fig. 4 Framework of OPC UA server based on EDD

In the framework when a DD is imported by user, the interpreter will interpret it and extract information from it, then encapsulate the information in XML format and send to node management, the node management can create device object with XML template. At the same time the node management sets the default values of variables. In fact *node objects* in Address Space only are the reflection of field devices, which is abstracted from DD document. DD only describes the information of variable such as parameter name, data type and help information, but DD doesn't contain factual variable value. Thus the interpreter must invoke the interrelated command to get the variable value and send the variable value to node management when node is created, then node management update the variable object value in OPC UA. And Node Management must monitor and manage address space, and send the request to DD interpreter when OPC UA server API sets parameter and invokes Method, DD interpreter communicates with field device by sending interrelated COMMAND through HART COMMUCATION.

In conclusion, the DD interpreter plays an important role in the framework. it not only maps DD into device object, but also updates the value of device object. Data exchanges between address space and interpreter can divide into two classes. One is the request/response mode when the variable is set or method is invoked. The other is the event notification and real variable updating mode. For the request/ response mode it can be realized by callback concept. For the other mode it can be realized by calling the universal COMMAND 3 periodically. Fox example, when a Variable object of *Node* is set, node management will send request to the interpreter, then the interpreter write the parameter into the device by calling the interrelated commands.

C. Model and Realization

From the above framework we can find that the interpreter plays the role of transforming DD into information model of OPC UA. In order to realize the transformation, Aimed at the content and structure of DD, the device object model which is in accord with the address space model of OPC UA is defined as following Figure 4 [8].

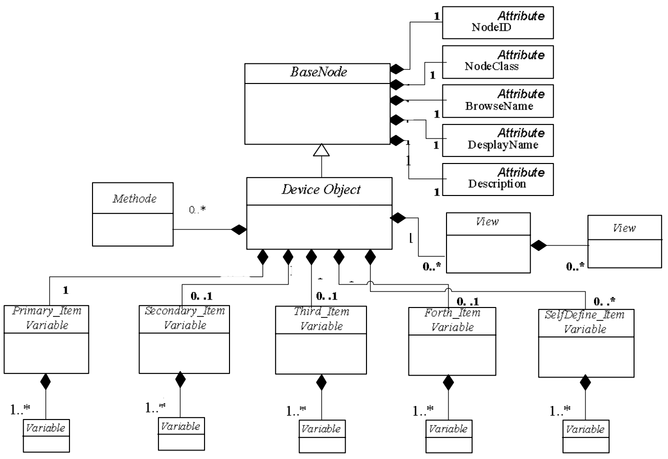

Fig. 5 Model of device object in OPC UA

In EDDL of HART, in order to organize and manage the variables, the element COLLECTION is used to describe the variable group, thus we define the several complex *Variable object* in the model, the complex variable is in accord with COLLECTION of DD. Complex variable Primary_Item is in accord with primary variable group which can contain several variables such as parameter unit, upper_limit and lower_limit of range etc which are correlative with primary variable. Secondary_Item is for secondary variable group and so on. And the model also contains the self-defined variable group. View and its *Hierarchical Reference* are in accord with the MENU and sub-MENU in EDDL. Method objects are in accord with METHOD in EDDL. The above model can contain all information and structure of DD document. Moreover the attributes such as *NodeID, BrowseName, DisplayName* and *Description* can be obtained from the basic information of device in DD. For example the LABEL of MENU root_menu can be used as *DisplayName* attribute.

In EDDL VARIABLE is the basic element, it can describe the device parameter, its unit and alarm information. It is described as following:

```
VARIABLE flow_value
{
LABEL "flow_value";
HELP "process value of flowmeter";
CLASS   DYNAMIC & INPUT;
TYPE   FLOAT
{
DISPLAY_FORMAT "5.2f";
}
HANDLING   READ;
}
```

A Variable object in OPC UA contains Value and the attributes such as *DataType, ArraySize, AccessLevel, UserAccessLevel and MininumSamplingInterval* except for the attributes *inherited from BaseNode* such as *NodeID, NodeClass, BrowseeName, DisplayName* and *Decription* Moreover the relation between Variable objects can be described by *Reference*.

From characteristics of VARIABLE in EDDL and *Variable object* in OPC UA we can see that the VARIABLE in EDDL is similar to *Variable object*. Thus the information of VARIABLE can directly be extracted as attributes of *Variable object*. In DD document IDs are assigned for each variable in compiling process, the ID can be used as *NodeID* of *variable object*. The name and LABLE of VARIABLE can be transformed respectively as the *BrowseeName,* and *DisplayName* of *variable object*. The content described by CLASS can refer to a *DataType object*. When a variable belongs to the DYNAMIC class, it indicates that the variable is a real variable, we must set *MininumSamplingInterval* attribute according to sampling periods of the variable. The TYPE can be transformed to *DataType* Attribute of *Variable object*. HANDLING can be transformed to the *AccessLevel* of *variable object*. And the *UserAccessLevel* of *variable object* can be ascertained by taking into account of the device state and *AccessLever*. For example a parameter is writable when device is in configuration state, but it is not writable when device in running state. Updating variable value is done by calling the interrelated COMMAND in HART.

The elements such as REFRESH, UNIT and WRITE_AS_ONE are used to describe the relation of variables in EDDL, for example UNIT is used to relate a parameter VARIABLE to a unit VARIABLE. WRITE_AS_ONE is used to describe that several interrelated variables are once written to device at one time. All the elements can be described by the *Reference* concept of OPC UA.

METHOD description in EDDL has different description measures and syntax compared with the other program languages, so it is comparatively complex to transform the METHOD in EDDL to *Method object*. METHOD has no *inputargument* and *outputargument*, user enter parameter on the visual interface when MEHTOD executes if it require parameters. But for the simple METHOD without parameter, we can pass its call entry from EDD interpreter to Method object. Factual execution of method is done by EDD interpreter when method in OPC UA is invoked by OPC UA client. The complex METHOD with parameters must be done by disassembling.

MENU in EDDL and *Views* in OPC UA have the similar functions, both are used to organize the variables and methods, and have the same hierarchical structures also, hence MENU can be directly transformed to *Views*.

The event contact with variables in EDD commonly, for example an alarm event is trigged when variable value is out of its range. Thus event is in accord with the *Reference GeneratesEvent* of *Variable object.* The interpreter sends the message to address space when it finds that alarms occur.

According to the transformation methods above, a device can be mapped as a *Device Object Node* by extracting the information from DD document. In the same way the devices in FF and PROFIBUS protocol can be mapped into OPC UA by their EDD technologies.

IV. CONCLUSTION AND FUTURE WORKS

In our framework, OPC UA server based on EDD has the virtue of OPC UA and EDD. By the means to design and develop OPC UA server, device integration can be enhanced, and system independent complex diagnostic packages can be applied to more applications. For manufacturer and user, the former investment in EDD can be preserved because their devices can be integrated into OPC UA application through EDD technology. But what we present is only a framework and model and it must be verified in practical application.

ACKNOWLEDGMENT

This work was supported by a grant form the Science and Technique Programs of Science and Technology Commission Foundation of Chongqing in China (No.CSTC-2007AB2031).And this work was supported by Youth Foundation of Southwest University in China (No. SWUQ2006016).

REFERENCES

[1] Hadlich, Providing device integration with OPC UA. Industrial Informatics, IEEE International Conference. Singapore, pp. 263-268, Aug 2006.

[2] http://www.opcfoundation.org.

[3] OPC FOUNDATION, OPC UA Part 1-Concepts Version 1.00 July 28, 2006.

[4] Stefan-Helmut Leitner, Wolfgang Mahnke. OPC UA - Service-Oriented Architecture for Industrial Applications. http://www.zdnet.co.uk.

[5] OPC FOUNDATION, OPC UA Part 3-Address Model Space. Version 1.00 July 28, 2006.

[6] IEC 61804-2: Function Blocks for process control – Part 2: Specification of FB concept and Electronic Device Description Language (EDDL). PAS Pre-Standard, pp.75-77, Apr 2002.

[7] Dip1.-Ing. Rene Simon, Prof: Dr.-Ing. Field Device integration. ISIE, Pusan, KOREA, 2001.

[8] OPC FOUNDATION, OPC UA Part 5-Information Model. Version 1.00 July 28, 2006.