The Research of XML-based Extensible Device Description for FF Fieldbus System

Zhongfeng Wang 1, Haibin Yu 2, Hong Wang 2, Aidong Xu 2, Yue Zhou 1
1 Zhongfeng Wang, Yue Zhou, Shenyang Institute of Automation, Graduate School of Chinese Academy of Sciences, China
email: (wzf,zhouyue)@sia.ac.cn
2 Haibin Yu, Hong Wang, and Aidong Xu, Shenyang Institute of Automation, Chinese Academy of Sciences, China
email: (yhh,wang,xad)@sia.ac.cn

Abstract—Fieldbus technology has been widely used in industrial control system. The coexistence of different kinds of fieldbus standards brings the problem of fieldbus interoperability. This paper proposes a solution named extensible device description based on XML for fieldbus interoperability. With XML, we can define standardized fieldbus device description language and uniform interface between configurator and device description source files. The DTD for XML-based fieldbus device description files for fieldbus control system is given in this paper. The interfaces for parsing device description source files are also defined.

I. INTRODUCTION

Fieldbus technology has been widely used in industrial control system. Considering application-specific requirements and system cost, users may use different devices from different vendors in one control system and may use different fieldbuses. In the case, the problem of fieldbus interoperability comes forth. Interoperability is the ability of a system or a product to work with other systems or products without special effort on the part of the customer [1]. The problem of interoperability is related to both the communication scheme of control network and the definition of control application [2].

Each fieldbus technology provides specific solution to satisfy the requirement of interoperability. For example, FF (Fieldbus Foundation) provides DD (Device Description) as a tool to achieve interoperability of field devices. PROFIBUS introduces GSD (General Slave Data) file to describe field devices’ attributes and parameters. DeviceNet and ControlNet define EDS (Electronic Data Sheet) as its device description format. These device description specifications bring confusion to end users. So a uniform device description specification should be defined to lessen the trouble forced on end users. Along with the development of Ethernet technologies, the Extensible Markup Language XML provides a promising solution for a general-purpose description. XML is a powerful description language independent of platform and has well interoperability. XML is an ideal candidate for the representation of complex structured data. Furthermore, the Document Object Model (DOM) makes it easy develop interrelated tools such as interface between configurator and device description source files. D.Buhler proposed CANopen markup language for CAN fieldbus system [3].

M.Wollschlaeger introduced concepts, description methods, software tools, and migration paths in XML-based frameworks [4].

This paper proposes a solution of extensible device description based on XML. The FOUNDATION fieldbus system is chosen as an example to show the feasibility of XML in the field of industrial automation. The research is to describe device functionality based on FF function blocks in XML and to define standard interface between configurator and device description source files.

II. FF FUNCTION BLOCKS

In FOUNDATION fieldbus system, each field device performs a portion of the total system operation by implementing one or more time-critical application or portions of an application, such as sensor data acquisition and control algorithm processing. Each application is composed of a set of elementary field device functions, modeled by function blocks. So function blocks are the key component for implementing control and monitoring function. Function blocks provide a general structure for processing data and event.

As shown in Fig.1, function blocks are defined by their input, output, contained parameters, and algorithms operating on these parameters. Data from other function blocks or sensors is snapped firstly by input snap procedure. After checking the status of this data, function blocks processes the input data by executing processing algorithm. At last function block publishes output value and status.

![Fig.1 FF Function Block Model](image-url)
III. A. The Structure of XML-based Device Description File

In order to describe device function, we use XML as description language to describe function blocks' information including input parameters, output parameters and contained parameters. This solution can describe devices perfectly. It provides a DTD (Document Type Definition) for developing XML-based device description source files.

The structure of field device description file based on XML is shown in Fig.2. Following section will give a detail introduction to all elements shown in Fig.2.

Fig.2 Structure of XML-based Device Description File

The XDDLDeviceDesc element is the root element of device description file. It has two different child elements: XDDLDeviceInfo and DeviceDescription. The preceding declaration can thus be written as follows:

```xml
<ELEMENT XDDLDeviceDesc (XDDLDeviceInfo, DeviceDescription )>
```

The XDDLDeviceInfo element represents device description file itself information and has two child elements: XDDLDocCreateDate which reveals the date and XDDLDocRevision representing the revision of device description source files. Follow declaration specifies the relation of these three elements.

```xml
<ELEMENT XDDLDeviceInfo(XDDLDocCreateDate, XDDLDocRevision )>
<ELEMENT XDDLDocCreateDate (#PCDATA )>
<ELEMENT XDDLDocRevision (#PCDATA )>
```

The DeviceDescription element is the starting markup of device description. It has two child elements: DeviceIdentify and BlockDescription. The DeviceIdentify element describes the information of fieldbus device about manufacture and type. The BlockDescription element is the starting markup of function block description. It has one BLOCK child element at least and no other attribute. The declaration can be written as follows:

```xml
<ELEMENT DeviceDescription( DeviceIdentify,BlockDescriptions )>
<ELEMENT DeviceIdentify (ManufacturerID, DeviceType )>
<ELEMENT ManufacturerID (#PCDATA )>
<ELEMENT DeviceType (#PCDATA )>
<ELEMENT BlockDescriptions ( BLOCK+ )>
```

The plus sign (+) indicates that the BlockDescription element must contain one or more copies of the BLOCK child element.

Fig.3 shows what the document of the device description should look like.

Fig.3 Structure of Document

IV. Function Blocks Description Elements

A. BLOCK element

Each BLOCK element represents one function block respectively and each BLOCK element has only one attribute and one child element. The declaration of the BLOCK element can be defined as follows:

```xml
<ELEMENT BLOCK ( BLOCKPARA )>
<ATTLIST BLOCK BlockType NM_TOKEN #REQUIRED>
```

The BLOCKPARA element is the only child element of the BLOCK element and is the starting markup of all parameters' description.

The BlockType attribute shows the type of function block such as AI, PID and so on.

B. BLOCKPARA element

In FF Function block application process specification, all parameters of function block are divided into three classes: simple variable, record and array. So three values: VARIABLE_TYPE, STRUCT_TYPE and ARRAY_TYPE, are defined to represent parameter's type respectively.

The BLOCKPARA element is the starting markup of all parameters description. Its sub-elements include S_PARA, V_PARA, and A_PARA. The declaration of the BLOCKPARA element can be written as follows:

```xml
<ELEMENT BLOCKPARA (V_PARA+, S_PARA+, A_PARA+)>
```

C. V_PARA element

The V_PARA element is used to describe simple variable. It has only one ParaType attribute which is fixed with VARIABLE_TYPE. We define the declaration of V_PARA element as follows:
The \texttt{S\_PARA} element is defined to describe record parameters. This element has only one \texttt{ParaType} attribute which is fixed with \texttt{STRUCT\_TYPE}. The declaration of the \texttt{S\_PARA} element is defined as follows:

\begin{verbatim}
<ELEMENT S\_PARA ( name, help, numofmembers, Members ) >
<\!ATTLIST S\_PARA ParaType NM TOKEN \#FIXED "STRUCT\_TYPE" >
\end{verbatim}

This declaration specifies that the \texttt{S\_PARA} element can include a \texttt{name} child element, a \texttt{help} child element, a \texttt{numofmembers} child element and a \texttt{Members} child element. The \texttt{name} child element and the \texttt{help} child element of the \texttt{S\_PARA} element are same to those of the \texttt{name} child element and the \texttt{help} child element of the \texttt{V\_PARA} element.

The \texttt{numofmembers} element gives the number of members of the described record parameter.

\begin{verbatim}
<ELEMENT numofmembers ( #PCDATA ) >
\end{verbatim}

The \texttt{Members} child element of the \texttt{S\_PARA} element is the starting markup of all members' description of the record parameter. All members are simple variable, so the \texttt{Members} element has two \texttt{V\_PARA} child elements at least. Its declaration is written as follows:

\begin{verbatim}
<ELEMENT Members ( V\_PARA+ ) >
\end{verbatim}

\textbf{E. A\_PARA element}

The \texttt{A\_PARA} child element of the \texttt{BLOCKPARA} element is used to describe array parameters. This child element has only one \texttt{ParaType} attribute which is fixed with \texttt{ARRAY\_TYPE}. Its declaration is defined as follows:

\begin{verbatim}
<ELEMENT A\_PARA ( name, help, numofmembers, Members ) >
<\!ATTLIST A\_PARA ParaType NM TOKEN \#FIXED "ARRAY\_TYPE" >
\end{verbatim}

The declaration specifies that the \texttt{A\_PARA} element can include a \texttt{name} child element, a \texttt{help} child element, a \texttt{numofmembers} child element and a \texttt{Members} child element. The meaning of the \texttt{name} child element and the \texttt{help} child element of the \texttt{A\_PARA} element are same to those of the \texttt{name} child element and the \texttt{help} child element of the \texttt{V\_PARA} element.

The \texttt{numofmembers} child element of \texttt{A\_PARA} element gives the number of members of the described array parameter.

The \texttt{Members} child element of \texttt{A\_PARA} element is the starting markup of all members' description of the described array parameter. All members are simple variable. Its declaration is defined as follows:

\begin{verbatim}
<ELEMENT Members ( V\_PARA+ ) >
\end{verbatim}

\textbf{V. INTERFACE BETWEEN CONFIGURATOR AND DEVICE DESCRIPTION FILE}

The Document Object Model defined by the W3C is a set of abstract interfaces which provides access to the logical tree structure of XML document. We can create, modify and explore the trees of XML document. The implementations of Document Object Model can be achieved with a variety of programming languages such as C++, Java and so on.

<table>
<thead>
<tr>
<th>#</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>char * XDDLFileSearch()</td>
</tr>
<tr>
<td>2</td>
<td>void XDDLGetBlockPara()</td>
</tr>
<tr>
<td>3</td>
<td>void XDDLGetBlockSimpleVarPara()</td>
</tr>
<tr>
<td>4</td>
<td>void XDDLGetBlockStructPara()</td>
</tr>
<tr>
<td>5</td>
<td>void XDDLGetBlockArrayPara()</td>
</tr>
<tr>
<td>6</td>
<td>void XDDLGetBlockParaEnumValue()</td>
</tr>
</tbody>
</table>

\begin{table}[h]
\centering
\begin{tabular}{|c|}
\hline
\textbf{Function}  \\
\hline
1  & char * XDDLFileSearch()  \\
2  & void XDDLGetBlockPara()  \\
3  & void XDDLGetBlockSimpleVarPara()  \\
4  & void XDDLGetBlockStructPara()  \\
5  & void XDDLGetBlockArrayPara()  \\
6  & void XDDLGetBlockParaEnumValue()  \\
\hline
\end{tabular}
\caption{Functions List}
\end{table}
The functions defined for configurator to parse XML-based device description file are listed in Table 1.

The first function is used to search the directory of XML-based device description file. The second one is used to search the description of the function block which resides in field device. The last four functions in Table 1 are called by the second one to get the descriptions of simple variable, array, record, enumerated name and value of simple variable respectively.

VI. AN EXAMPLE OF XDDS DEVICE DESCRIPTION FILE

According to the DTD mentioned above, we define a device description file for FF I/O device manufactured by Shenyang Institute of Automation and applied it successfully in our test system. The XML-based device description source file is shown in Fig.4.

```xml
<?xml version="1.0"?>
<XMLDeviceDesc>
  <DeviceDescribe>
    <DeviceIdentify>
      <ManufactureID>002</ManufactureID>
      <DeviceType>10</DeviceType>
    </DeviceIdentify>
    <BlockDescriptions>
      <BLOCK BlockType="Z">*
        <S_PARA>
          <ParaType="STRUCT_TYPE">*
            <V_PARA>
              <ParaType="VARIABLE_TYPE">*
                <name>ST_REV</name>
                <help/>
                <class>CONTAINED</class>
                <type>UNSIGNED</type>
                <length>2</length>
                <operation>READ</operation>
              </V_PARA>
            </S_PARA>
          </ParaType="STRUCT_TYPE">*
      </BLOCK>
    </BlockDescriptions>
  </DeviceDescribe>
</XMLDeviceDesc>
```

Fig.4. An Example of XML-based Device Description Source File

VII. SUMMARY

The XML-based device description solution proposed in this paper provides a means to describe process data in industrial control system. The DTD is defined to make it easy to create a device description file. The XML-based device description file is very easy to modify according to the requirements of control system. Since XML is an extensible description language independent of operating system, the data descriptions in XML-based device description file are available in a standardized way. We think that XML will be used widely in industrial control system in the future.

VIII. REFERENCES