Available online at www.sciencedirect.com

ScienceDirect

Procedia Environmental Sciences 2 (2010) 1525–1533

Procedia Environmental Sciences

International Society for Environmental Information Sciences 2010 Annual Conference (ISEIS)

Design for the Heterogeneous Environment Monitoring Network Component-base Management System

K.R.Zhao[b]*,Q.Q.Ma[a], X.Y.Zheng[c], K.C.Lu[c]

[a]*Buueau Geology of Guangdong Province, Guangzhou, China*
[b]*South China Institute of Environmental Sciences, MEPA, Guangzhou, China*
[c]*South China University of Technolog*

Abstract

Based on the current research on the technology of the environment monitoring informatization at home and abroad and its application, a component-based heterogeneous environment monitoring system was constructed, using structure analysis of environment monitoring system and component-base technology. The system includes heterogeneous environment monitoring network component-base design standard, component library, framework as well as the environment facilities driver library. The platformization, standardization and reusability of environment monitoring system development could be carried out effectively in this system, which at the same time improves the combination property of the environment monitoring system and meets the integrating and upgrading need of the existing heterogeneous environment monitoring network.

***Keywords*:** Environment monitoring, Heterogeneous integration, Component-base technology;

1. Introduction

Over the last 20 years, with the rapid development of economy, the environment protection becomes more and more urgent. In response to this situation, the development of the envirenment monitoring system in China successively underwent three phases, namely, passive monitoring, active monitoring and automatic (online) monitoring. Currently, research in environment automatic monitoring information technology has become one of the most hot domains in global environment monitoring technology. During the eleventh five year plan, development on environment monitoring system was accelerated in China. At the moment, environment monitoring system of certain scale has been formulated, which lays a solid foundation for accurately and punctually acquiring

* Corresponding author. Tel.: +86 20 85532915; fax: +86 20 85532915.
E-mail address: zhaokunrong@scies.org.

1878-0296 © 2010 Published by Elsevier Open access under CC BY-NC-ND license.
doi:10.1016/j.proenv.2010.10.165

environment monitoring data, objectively looking into the overall environment quality conditions and changes, clarifying the emission situation of principal pollutant from key polltant sources, and scientifically and effectively dealing with sudden events concerning environment[2] . Ever since a long time ago, the environment monitoring informatization and ability to process the environmental information in China have been failing to meet the increasing demand of environment protection cause. Although currently, cenvironment monitoring system of certain scale has been formulated, the heterogeneous property of the environment monitoring network keeps showing up, such as, the isomerism in the communication medium and communication protocol of the environment monitoring facilities, and the heterogeneity in the transmission mode of the content to be monitored. Therefore, it becomes the crucial technical needs, to improve the standardization, modularization and universalization of the environment monitoring network, for both integrating and upgrading heterogeneous environment monitoring network as well as the development of the informatization of environment monitoring system in China. This paper aims to design the heterogeneous environment monitoring network component-base management system by employing component-base technology so that the platformization, standardization and reusability of environment monitoring system development could be carried out effectively.

Component-base technology and software reuse technology are essential in the process of the industrialization of software products[3]. The “component” here is similar to the standard component in conventional industry. Combined with corresponding business logic, different application systems can be produced by assembling the components produced on the basis of agreed rule. Every component defines some interfaces, through which the information is exchanged with the outside world. So does the software component. Centering on the object-oriented, plug-and-play software component, software component technology establishes the applied technology system and formulates different application system through the assembly of the components. Currently three types of distributed component technology, namely, COM/DCOM、Java Beans and CORBA, are popular around the world. Component base management system is configured for the sake of the establishment, application and maintenance of the component base. With this system, the component base is managed and controled systematically, and the users can easily and efficiently search and quote components in the component base. Besides, this system makes it easier to maintain the safety and integrity of the components therein.

2. Framework for the Heterogeneous Environment Monitoring Network Component-base Management System

The environment monitoring network component-base management system designed in this paper mainly includes four parts—environment monitoring facilities component-base design standard, component library, extension framework as well as the environment facilities driver library. Figure 1 shows the framework for the heterogeneous environment monitoring network component-base management system.

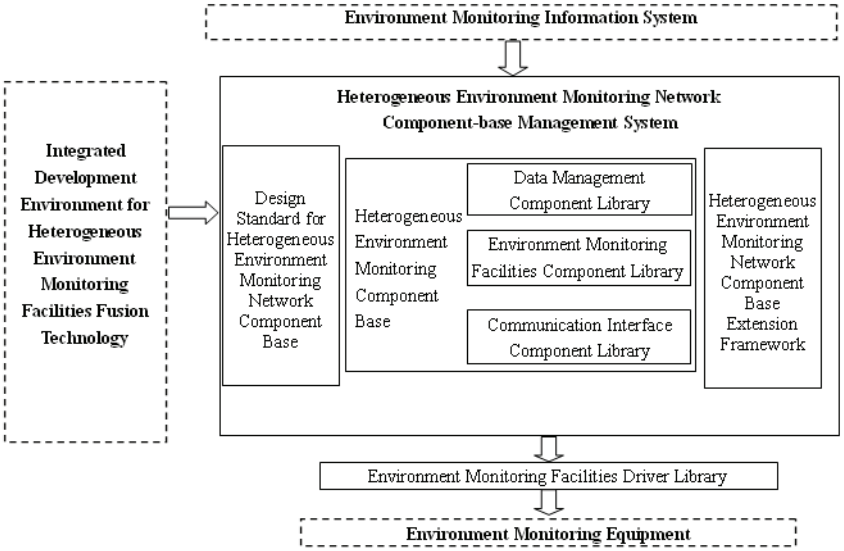

Figure1 The framework for the heterogeneous environment monitoring network component-base management system

3. Detailed Design for the Heterogeneous Environment Monitoring Network Component-base Management System

3.1. Design Standard for the Environment Monitoring Facilities Component Library

Application components of the environment monitoring platform provide a set of application oriented functional interface. Each component involves the fulfilling of one particular function. Being a packaged service provider, each component cooperates with other components through favorable interfaces. From the common model structure, it can be inferred that the components realize and provide a series of function service. However, details of the fulfilling of these functions, that is, the internal processing logics of the components, are invisible to the outside world; components only provide the outside world with interfaces corresponding to the service, which include interfaces providing services and receiving service. The standard specifications for the application components concerning heterogeneous environment monitoring facilities are as follows:

COMPONENT<class name>{
DESCRIPTION<function, capability, object,...> // component description
PARAMETERS<parameters list>// parameters list of the component
INTERFACES<interfaces list>// interfaces list
INTERFACE1<interface name>// name of interface 1
FUNCTION1<function description>// function description of interface 1
DIRECTION1<input or output>// input and output description of interface 1
PARAMETERS1<parameters list>// parameters list of interface 1
LOGICAL FLOW 1<processing>// logic flow processing of interface 1
MEMORYALLOC1<allocation>// memory allocation of interface 1
ALGORITHM1<algorithm>// algorithm of interface 1
REMARK1<attended problems>// related restrictions on interface 1
INTERFACE2<interface name>name of interface 2
...
INTERFACE n<interface name>name of interface n
INTERFACE END
}COMPONENT END

In order to describe the demand analysis and conceptual design of the components more clearly and carry out the architecture of the component base, this paper defines simple characters and character string so as to comprise descriptive tools and give a formal description to the component design. The sense of the formal symbol is as follows.

Table1 Component formal symbol description Table

Symbol	Sense
: : =	Defined as
{ }	Function aggregate
^	On account of
||	or

…	A couple of

3.2. Heterogeneous Environment Monitoring Network Component Library

Based on the existing heterogeneity, isomerism and complexity of the existing environment monitoring network, this paper makes a systematic exploration into the design model of all functional modules, constructs component bases of bottom layer and integrates data management, communication interface and bottom-layer heterogeneous environment monitoring facilities of environment monitoring instrument. In addition, this paper effectively shields the isomerism and heterogeneity of the bottom-layer hardware platform, and successfully separates the environment monitoring data collection system and the bottom-layer hardware platform. The structure of the heterogeneous environment monitoring network component library is as follows.

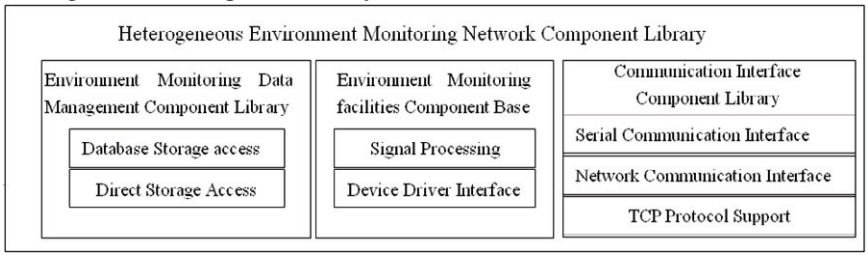

Figure2 Heterogeneous environment monitoring network Component Library Structure

3.2.1. Design for the Environment Monitoring Data Management Component Library

The environment monitoring data management component library provides the environment monitoring system with data management function, which includes opening and closing the database and the data table, as well as accessing, inserting, deleting and modifying data record. According to demands of the data management of the system, users can choose the bottom-layer database to establish corresponding database; they can also choose the operation of direct data storage access based on storage devices such as FLASH. The details are show in Figure 3.

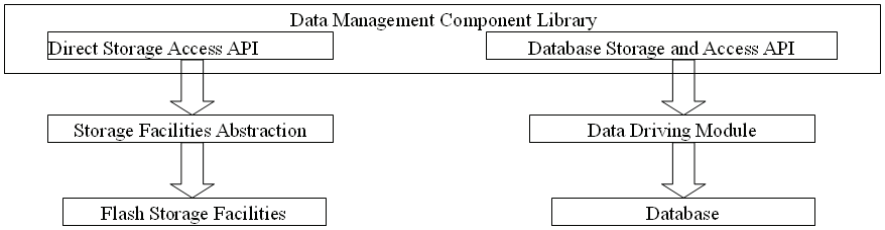

Figure 3. Data Management Component Library Structure

Formal description of the component functional needs is as follows:

Data management component library::={ database operation component ‖ direct data manipulation component}

Database operation component::={ connect ‖ disconnect ‖ open ‖ close ‖ insert ‖ delete ‖ modify ‖ extract}^ bottom-layer database

Bottom-layer database::= { database 1 || database 2 ||……}

Direct data operation component::= { open ‖ close ‖ establish ‖ insert ‖ delete ‖ modify ‖ extract}^ abstract base drive for the storage devices

3.2.2. Design for the Environment Monitoring Equipment Component Library

The environment monitoring equipment component library provides the application development of the embedded environment monitoring platform with environment monitoring instrument, algorithm for the monitoring indicator signal exchage management and environment monitoring driving library. These algorithms include linear

prediction, averaging calculation (arithmetric average, harmonic average, geometric average), and variance analysis. Its structure is shown in Figure 4.

Formal description of the component functional needs is as follows:

Data management component library::={ database operation component ‖ direct data manipulation component}

Database operation component::={ connect ‖ disconnect ‖ open ‖ close ‖ insert ‖ delete ‖ modify ‖ extract}^ bottom-layer database

Bottom-layer database::= { database 1 || database 2 ||……}

Direct data operation component::= { open ‖ close ‖ establish ‖ insert ‖ delete ‖ modify ‖ extract}^ abstract base drive for the storage devices

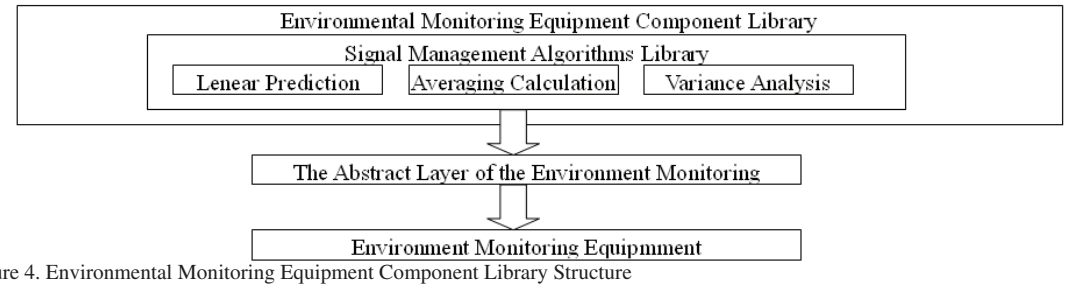

Figure 4. Environmental Monitoring Equipment Component Library Structure

3.2.3. Design for the Communication Interface Component Library

The communication interface component library provides the input and output operation towards the external equipment of the environment monitoring instrument. These operations include support for interfaces such as UART, Modem, GPRS and Ethenet etc. Besides, the component library also supports RTP/RTCP(Real-time Transport Protocol). Its structure is shown in Figure 5.

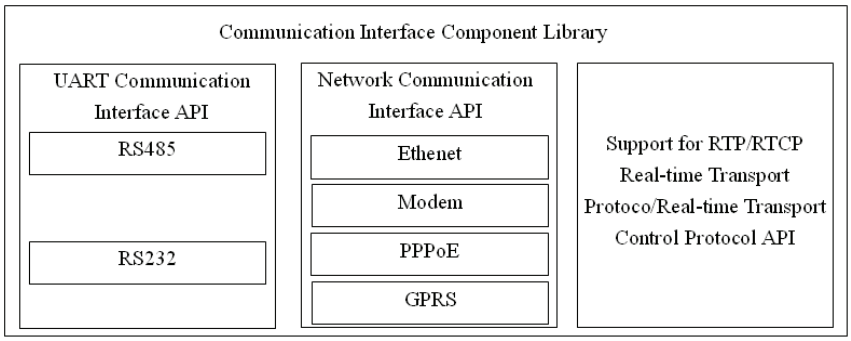

Figure 5. Communication Interface Component Library Structure

Formal description of the component functional needs is as follows:

Communication Interface Component Library::={UARTI||Internet||RTP/RTCP}

UART

Directed mainly towards the communication support for interfaces such as RS232, RS485 etc.

The Internet

Directed mainly towards support for the internet communication interfaces such as Ethemet, PPPoe, GPRS, Modem etc.

RTP/RTCP Real-time Transport Protocol

This component library supports the Real-time Transport Protocol/ Real-time Transport Control Protocol (RTP/RTCP), reads and writes RTP data, acquires the SSRC (Sychronization souree) of RTP conversation, acquires and sets the RTP header and header length, appoint the acquisition and setting of loadtype RTP header and header length, acquires SSRC of RTCP conversation, obtains the address of the opposite party in the RTP conversation, receives and sends RTP data and give notice of the RTCP etc.

3.3. Design for the Extension Framework of the Environment Monitoring Facilities Component Library

Except for several kinds of compoment library mentioned above, the environment monitoring facilities application component can also adopt new compoment libraries according to the development needs. This can be realized mainly through increasing extension points program in the graphic IDE. IDE designed in this paper is based on the concept of extension point plugins. The extension point is the strictly defined point in the system, and at the same time tools provided by the IDE (such as new component library)can also be added by using this point. In the IDE plugin system, the extension point provides interfaces for the plugin, and each plugin is developed based on the existing extension points while retaining its own extension point for the convenience of redevelopment. Related information such as name of the plugin, version number, name of the provider and class name should be provided for each plugin. With plugins, tasks to be fulfilled at the start-up stage of the core of monitoring platform will become simple: to start the base component of the platform and search the system plugin. With regard to the newly increased component library, the corresponding information can be found, initialized and displayed on the platform at the start-up stage. The structure of the whole system is like a jigsaw, to which plugins can be constantly added. The lucid XML file is chosen for plugin maintenance. As to the extension of component library, there is a corresponding XML file, that is phigin.xml. Information about the file phigin.xml reflected by adding a new component library is as following:

```
<extension
    …
  point=”org.zjuIslab.baseComponent.ui”>
  <category
     id=”org.zjuIslab,baseComponent”
     name=”baseComponentWizardCategoryName”
  </category>
  <Component
   id=”org.zjuIslab.baseComponent.networkComponent”
   name=”networkComponent”
   label=”newComponentName”
   class=”org.zjuIslab.baseComponent.networkComponentWizard”
   category=”org.zjuIslab.baseComponent”
   project=”ture”
   finalPerspective=”org.eclipse.cdt.ui.C/C++Perspective”
   preferredPerspctives=”org.eclipse.cdt.ui. C/C++Perspective,org.eclipse.cdt.ui”
   C/C++BrowsingPerspective,org.eclipse.cdt.ui.C/C++HierarchyPerspective
   Icon=”icons/full/ctool16/newjprj_wiz.gif”
```

```
    <description>%NewC/C++Project.description</description>
  </Component>
…
</extension>
```

3.4. Environment Monitoring Equipment Drive Library

In order to satisfy demands of environment monitoring information system on property, stability and extendanility of the equipment driver, this paper adopts design methods for equipment driver procudures basing on dynamic link library technology, transforming various equipment driver into corresponding dynamic link libraries. The environment monitoring information system defines unified protocol interfaces for these DLLs. Before accessing a particular I/O equipment, the environment monitoring information system is required to first load the equipment driver DLL so as to get the handle of a DLL module, and then call the Get Proc Address function in order to get indicator for the output function. After that, according to the unified equipment driver interface, the system needs to call each function defined in the interface on the basis of established procedure so as to achieve the communication between the environment monitoring system and the equipment. These dynamic link libraries are established in the way of in-process service, operating in the same process with the environment monitoring information system. In this way, an effective and seamless communication linkage between the environment monitoring information system and the equipment driver is established. Moreover, simply by providing corresponding equipment driver DLL which conforms to requirements on unified interfaces the new equipment can be smoothly interposed into the environment monitoring information system.

3.4.1. Interface Function of the Equipment Driver Library

For different types of I/O equipment, functions with the same function of their equipment driver library are of the same name and parameter. In other words, different types of equipment have the same interface function, which contributes to unify the interface specifications and realize the compatibility of environment monitoring information system and various equipment drivers.

On the basis of the communication procedure between the environment monitoring information system and I/O equipment as well as requirements on the subcontracting of data item in the real-time database, the following interface functions are designed for the environment monitoring component library:

int OpenDevice(const char *addr)

Function declaration: open the equipment and go back to the device descriptor

int TryConnect(int devid, int trycount=1, CONNECT_CALLBACK=0, DWORD data=0)

Function declaration: connect the equipment, in the event of blocking linkage, returning the number 0 signifies success, otherwise means failure; in the event of asynchronous linkage, returning the number 0 signifies success. The results are notified in the callback function.

BOOL GetRegisters (char *szDeviveName,LPVOID **ppReg,int *p RegNum)

Function declaration: obtain the name and number of the register of the equipment determined by the sz Device Name. If the sz Devive Name takes effect, TRUE is returned; otherwise the FALSE is returned.

BOOL CheckConfig(DbItem*lpDbItem)

Function declaration: conduct a validity check on users' configuration information; if the function is legal, then TRUE is returned; otherwise error information is configured, and FALSE is returned.

int GetDevice(int devid, int *deviceID=0, int *deviceNum=0)

Function declaration: read the equipment parameter, inquire information and ID number of the equipment; if the number 0 is returned, then succeed.

(6) BOOL AddVarToPacket (LPVOID lpVar, int nVarAccessType,

LPVOID lpPacket)

Function declaration: verify whether the variable can be collected together with other variables of certain package so as to pack variables. If true, then modify the start and stop address and TRUE is returned; otherwise FALSE is returned.

(7) int ProcessPacket(LPVOID lpPacket)

Function declaration: deal with the data in the package according to the protocol and the condition of the package.

(8) const char GetLastErr(int devid = -1)

Function declaration: gain relevant information on errors recently committed by the equipment.

(9) int CloseDevice(int devid)

Function declaration: close the equipment; the equipment descriptor becomes unavailable after the equipment being closed; the return of the number 0 indicates success.

3.4.2. Working Procedure of the Driver and the Enforcement

In the following part, taking drivers aiming at TCP/IP communication protocol for example, this paper demonstrates the working procedure of the driver. For other communication protocols, the same work procedure can be realized simply by modifying the corresponding code in the interface functions, with the same driver structure. Typical environment monitoring information system includes development environment and operating environment. In the following sections, the working procedure and implementation procedure of drivers in the above protocols are introduced.

(1) Driver under the development environment of the environment monitoring information system

1) When the user defines the data item, the development environment will call Get Registers function to acquire register list (based on the hardware features of the target I/O equipment, the driver defines corresponding registers for each input and output channel) defined in the driver as a choice for the user.

2) After the user defines a data item, the development environment will call Check Config function and conduct a validity check on users' configuration information. If the information is legal, the development environment will keep the new data item into the real-time database of the environment monitoring information system; if there is any error in the configuration information, FALSE is returned.

3) No matter for which function in the development environment, once FALSE is returned, the development environment calls Get Last Err function, and displays the error message.

(2) Driver under the operating environment of the environment monitoring information system

1) In the initial process of collecting variables linked list, the driver transforms the data item in the real-time database into corresponding Plc Var memory variables. These variables are deposited in the internal memory (composing linked list on the basis of equipment address, access frequency, register type, and access mode) in accordance with certain data structure (multiple linked list), which can obviously increase the speed of processing data item. This can be realized by the function BOOL CProject::RunQuene_init (C Point List &list Point).

2) If functions in the 1) are executed smoothly, then call the Open Device function. The initial communication equipment, in this case, is to load WinSock environment, so as to prepare for establishing TCP connect later.

3) The scanning in the operating environment is conducted by the multiple structures composed by PlcVar variables. When the collection time of any variables is up, call AddVarToPacket function. This function judges whether the variable can be interposed into the existing package for coprocessing: if the variable can be interposed (in this case, the package and the variable are on the same equipment, and the access type and register type are the same), TRUE is returned; otherwise FALSE is returned. The operating environment adopts corresponding actions according to the returned value: if TRUE is returned, it interposes PlcVar variable in the form of IdNo panel point into the data item linked list of the existing data package; if FALSE is returned, it establishes a new data package, takes the corresponding IdNo panel point of the existing PlcVar variable as the first panel point of the data item linked list in the new data package, and interposes the new data package into the package processing list in priority of package establishing.

4) Recursively call ProcessPacket function, deal with the data package taken from the package processing list, one at a time. Data transmission n the whole communication process is realized in this function.

BOOL ProcessPacket (PPACKET IpPacket)

```
{
if (not yet establish TCP connection with the equipment)
call TryConnect function to establish connection;
if (fail in establishing connection)  return FALSE;
}
if (estimate the access property of package ==read the package){
transmit command of collecting data to the equipment;
receive data returned by the equipment;
if (succeed in data receiving){
traverse data item linked list of the existing package, assign values for field PlcValue of each IdNo panel point;
} else {return FALSE ;}
}
else {// the access property of package is to read the package
traverse data item linked list of the existing package, get values form field PlcValue of each IdNo panel point, and then send the values to the equipment with command of writing package;
if (fail in writing package) { return  FALSE;}
}
retrun TRUE; //indicating the existing package is successfully processed
}
```

5) When the ProcessPacket function returns FALSE, call TryConnect to recover the connection; if the function returns TRUE, then turn to step(3); if in the limited number of attempts defined by the user to recover the connection the connection is not established, then the driver will not be communicated with the equipment.

6) When the operating environment quits, call CloseDevice function to stop communication.

4. Conclusion

Based on the component-base technology, in this paper, a component-based heterogeneous environment monitoring system was constructed. This system includes heterogeneous environment monitoring network component-base design standard, environment monitoring facilities component library, environment monitoring facilities extension framework as well as the environment facilities driver library. This system effectively integrates data management, communication interface as well as isomerism and heterogeneity monitoring facilities in the underlying layer of the monitoring instrument, improves the platformization, standardization and reusability of the environment monitoring network, shields the bottom-layer hardware platform, and promotes the integrating and upgrading of the existing heterogeneous environment monitoring network.

References

[1] Liu, Jianguo, Liu, Wenqing & Wei Qingnong. Environment Monitoring Technology and its Development Direction. Photoelectronic Technique and Information, 2001, p7-11.

[2] Academy of Environment Planning of State Environmental Protection Administration. Research Report on Environmental Protection Program during the State Eleventh Five-Year Plan. 2006, 8-11.

[3] Li Xiang, Zhou, Xionghui & Ruan, Xueyu. Solving Strategies on Collaborative Design Moving Constrain. Journal of Shanghai Jiaotong University, 2001, 35(7): p1008-1010.

[4] Andert R., Mendgen R. Modeling with constraints: theoretical foundation and application [J].Computer Aided Design, 1996, 2 8(3):p155-168.

[5] Jiang, Weijin & Xu, Yusheng. Research in Distributed Collaborative Design System Based on Component Technology. Machine Design and Research, 2004, 20(z1): p1-6.