Journal of Applied Mechanical Engineering

Lin and Feng, J Appl Mech Eng 2013, 2:2
DOI: 10.4172/2168-9873.1000119

Research Article Open Access

Reconfigurable Manufacturing Modular Structure Based on OPC Technology

Dong Lin* and Chu Xiang Feng

School of Mechanical Engineering, Shanghai University of Engineering Science, China

Abstract

To realize the modular development of manufacturing system, this paper proposes a new structure of reconfigurable manufacturing modular structure by introducing OPC technology. According to the theory hierarchical control system, we add a new OPC standard interface layer to system structure. Then, detailed design of OPC standard interface layer is presented, including design principle, the algorithm to coding OPC server/client and prototype system. Finally, the new modular system is applied in a product line of Molded-low-voltage circuit breaker. The result shows that the new module runs stably to meet reconfiguration of manufacturing system rapidly.

Keywords: OPC; Control system; Reconfigurable manufacturing system

Introduction

The need and rationale for reconfigurable manufacturing systems arises from unpredictable market changes that are occurring with increasing pace during the recent years [1]. These changes includes introduction of new products increasingly and frequently, changes in parts for existing products, fluctuations in demand and complication largely, changes in governments regulations (safety and environment requirements) and changes in industry process technology. These changes are driven by aggressive economic competition on global scale, more-educated and demanding customers, and a rapid pace of change in industry process technology [2]. These drivers by changes also need a new balance among economy, society and technology. To meet new requirements in the manufacturing development, engineers and companies must be able to react to changes rapidly, effectively and low-costly.

In order to exchange information in manufacturing system efficiently, rapidly, all control cell involved have interact as seamlessly as possible. This also requires a new standard interfaced modular development in manufacturing system. This problem is to be remedied by OPC technology supporting the standard interface of automated engineering. OLE for Process Control (OPC), that is a OLE using in process control, is common standard interfaces in the field of automation control and manufacturing system. Based on embedding (OLE) object connection, Microsoft's Component Object module (COM) and distribution -component Object module (DCOM) technology, it modules OPC server and client as interacting partners. Clients communicate with server by C/S request and response, and servers can publish notification to clients [3]. It is characterized with a language-independent, code reusability and ease of integration, etc. By OPC technology, it is access to visit all hardware and software in a uniform manner. Then, it is able to develop system in type of "plug-in"; meanwhile, engineer can be free from underlying coding so as to shorten the development cycle and reduce the difficulty. These advantages of OPC technology enable a promoting application of module in reconfiguration manufacturing system.

This paper is organized as following. The OPC technology is introduced deeply, including its family, operation module and structure. Then, the later introduces the detailed design of reconfiguration module structure, including design principle, algorithm of coding OPC server/client and prototype system. Finally, a application sample demonstrate running results this new reconfiguration module system.

Reconfiguration Module Design in Manufacturing System

Introduction of OPC technology

OPC specification is a universal interface standard, jointly developed by leading automation hardware and software company and Microsoft. Based on embedded OLE objects, COM Component Object module, and DCOM Distributed Component Object module technology development, it defines the forms and methods of data exchange between OPC server and client. Recently, the OPC family, shown in figure 1 has developed a series content, such as Data Access,

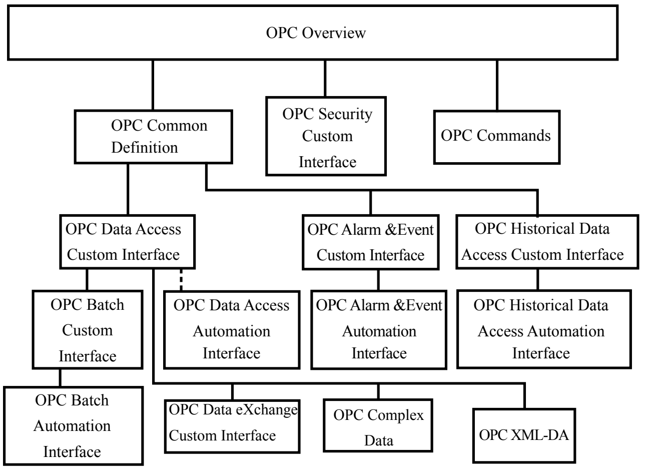
OPC Overview
OPC Common Definition
OPC Security Custom Interface
OPC Commands
OPC Data Access Custom Interface
OPC Alarm &Event Custom Interface
OPC Historical Data Access Custom Interface
OPC Batch Custom Interface
OPC Data Access Automation Interface
OPC Alarm &Event Automation Interface
OPC Historical Data Access Automation Interface
OPC Batch Automation Interface
OPC Data eXchange Custom Interface
OPC Complex Data
OPC XML-DA

Figure 1: OPC family.

***Corresponding author:** Dong lin, School of Mechanical Engineering, Shanghai University of Engineering Science, China, E-mail: knight0121@163.com

Received December 04, 2012; **Accepted** February 18, 2013; **Published** February 24, 2013

Citation: Lin D, Feng CX (2013) Reconfigurable Manufacturing Modular Structure Based on OPC Technology. J Appl Mech Eng 2: 119 doi:10.4172/2168-9873.1000119

Citation: Lin D, Feng CX (2013) Reconfigurable Manufacturing Modular Structure Based on OPC Technology. J Appl Mech Eng 2: 119 doi:10.4172/2168-9873.1000119

Bath, Security, and XML, the Complex the Data, as well as OPC Data eXChange etc. Running on client/server structure, OPC client communicates with server through OPC communication stack which is used on client-side and server-side encode and decode message requests and responses. OPC 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 implementation sends request and receives data over OPC client API. The server implementation receives OPC server API invocation and returns data from address space in which nodes are maps of real objects. In the structure, as shown in figure 2, server also can read data from various clients and vice verse. OPC specification (Data Access) defines the Automation Interface and Custom Interface. Automation Interface codes by script language, such as Visual Basic.

Delphi, PowerBuilder, Custom Interface is able to be code by high-level program language, such as C++ and .NET framework, and it is more efficient than Custom Interface but more difficult to develop. We usually take the latter mode because it is able to get the best performance of the OPC server in terms of its high efficiency. Additionally the OPC Foundation offer automation interface wrapper (wrapper DLL) to switch modules between two sets of interfaces. The typical OPC architecture is shown in figure 3.

The OPC specification consists of OPC server (server object), OPC Group (group objects) and OPC Item (item object), is shown in figure 4. The OPC server defines manageable information of server, and also can be a container for OPC Group and OPC Item; the OPC Group is responsible for maintaining itself. It also can be a container and manager for OPC Items logically; The OPC Item defines detailed item information, including item definition, value and state etc. Actually, it is not a physical data source but a logic link to the real data. The OPC Item can be connected to the real data in term of Item ID, a unique said.

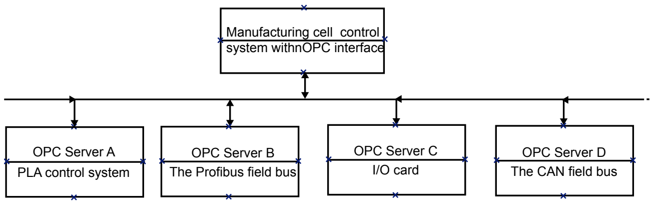

Figure 2: the manufacturing cell control system based on OPC.

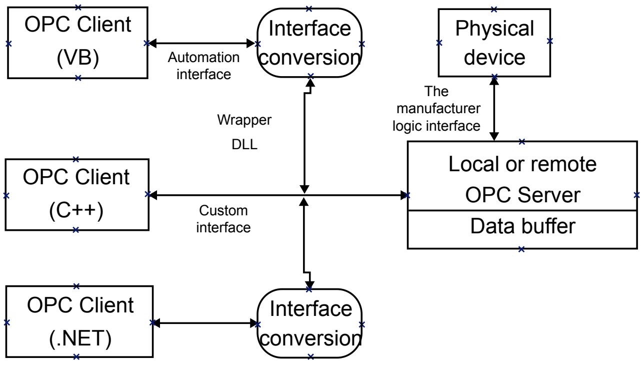

Figure 3: typical OPC architecture.

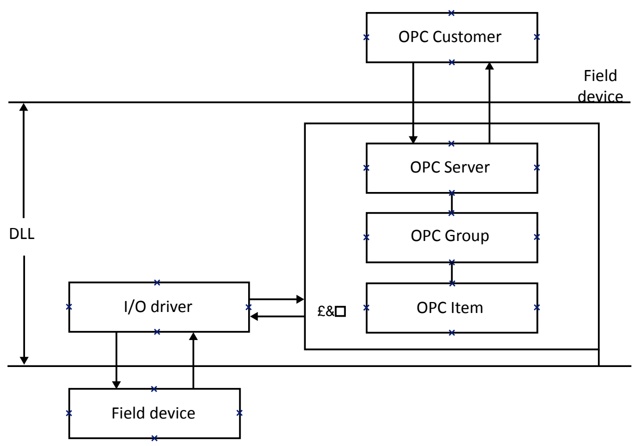

Figure 4: Data process in OPC.

As data server, OPC server contains item address space. Client browses item information through these address space in which items link to physical data source. Thus, data manager process can be concluded as following: we get data from physical device in work-site and then OPC client read and send these messages through OPC server. Next these data is stored in realtime or history database for next reading [5].

Design of reconfiguration manufacturing modules

The design of the control structure of the reconfigurable manufacturing unit needs to meet the following two requirements [6].

Modular design: module can be able to manage dramatically, such as adding, deleting and moving etc.

Structured design: multi-layer structure is design to react the complex system, and each layer has its own function and levels of precision.

According to the above requirements, we design five layers, including device module layer, standard interface layer, local control layer, global control layer and central manage layer, as is shown in figure 8. We add a new standard layer to traditional hierarchical control system to realize standard communication. Therefore, the message does not communicate between upper control layer and the underlying physical device layer any more. Instead, message communicates between standard interface layer and other layers. It is more easy to realize the modular management in terms of the switching. The detailed information of five layers is as following:

1. Device module layer: The layer is responsible to input and output message in system. Device module layer consists of physical device and logic device. Physical devices are mostly data acquisition devices, such as current sensor and voltage test device. On the other hand, the logic devices can be device logically, such as data server and OPC server.

2. Standard interface layer: the layer defines various module of standard communication, including steps to develop OPC server and client. Additionally, the layer is responsible to convert other protocol of device vendors to OPC standard protocol, as is shown in figure 2.

3. Control layer (local control and global contro l): The layer

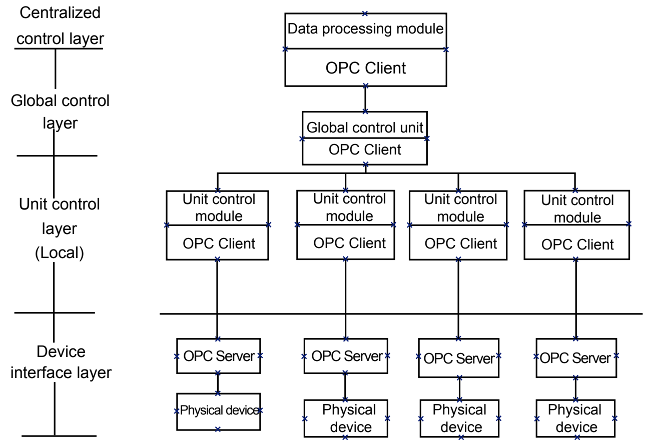

Figure 5: Reconfigurable manufacturing cell control structure.

is mainly for control function in system. In the local control layer includes the interface to client in terms of opc protocol. Meanwhile, the global layer offer safety control in system

4. Center manage layer: the layer manages realtime or history data. It offers order management, process parameter setting etc.

As is shown in figure 5, various modules combine the reconfiguration structure, and is characterized with adding or deleting modules easily. In the system, the standard interface, based on the COM/DOM components is a key layer to satisfy the requirement.

Reconfigurable Modular Manufacturing Architecture

Though discussion above the key to realize reconfigurable module is the standard layer. So we use OPC technology and component technology to develop this layer. This chapter describes theory, algorithm and an OPC software development kit (SDK) [reference] to code OPC server and OPC client. By calling function in OPC SDK to communicate with underlying communicated protocol, the OPC server and client in standard layer are able to be realized [7].

OPC server development kit

OPC_CQUPT_Svr.dll is a OPC server rapid development Toolkit, as is shown in figure 6, which defines all detailed information of COM and OPC server, the SDK offer various API function for data sets. Therefore, it is easy for engineer to pay all attention on developing the data acquisition. In the same time, the DLL is also easy to integrated into system. The SKD supports data accesses to OPC server by creating one thread for every client.

OPC Development Kit software consists of three parts:

1. OPC_CQUPT_Svr.dll dynamic link library
2. PC_CQUPT_SvrAPI.h header file
3. OPC_CQUPT_Svr.lib static libraries

Static library package defines export functions of API. All these parts are included in the OPC server project files. According to function, programmable interfaces of OPC SKD categorize into 3 types:

The OPC Toolkit software programming interface is divided into three categories: Server management, Data-property management and Real-time data management.

1. OPC Server management: It defines server architecture. It offers definition to register, start, stop and other function. It composes of OPC architecture module, initialization modules, register module and etc.
2. OPC Data property management: It organizes the address space and manages data property information and it composes of assistance function, OPC Item function etc.
3. OPC Realtime data management: It define realtime reading and writing functions, consisting of realtime message function, return message function and

According to above analysis, OPC server development toolkit (OPC_CQUPT_Svr.dll, as is shown in figure 7 have 6 function modules and detailed information is as following [8]:

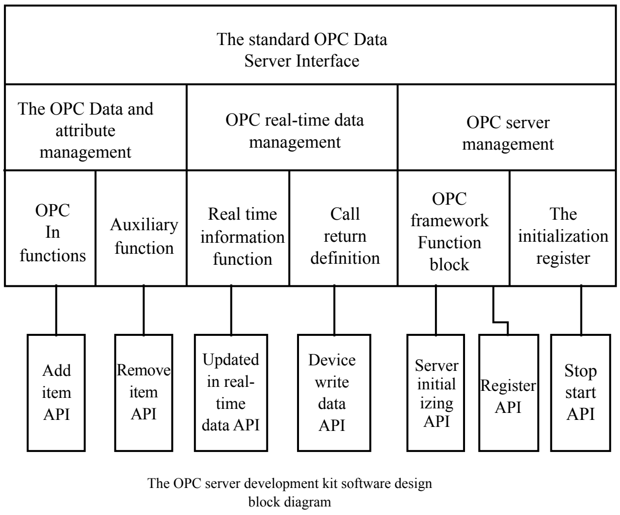
The standard OPC Data Server Interface
The OPC Data and attribute management
OPC real-time data management
OPC server management
OPC In functions
Auxiliary function
Real time information function
Call return definition
OPC framework Function block
The initialization register
Add item API
Remove item API
Updated in real-time data API
Device write data API
Server initializing API
Register API
Stop start API
The OPC server development kit software design block diagram

Figure 6: OPC server software development kit design diagram.

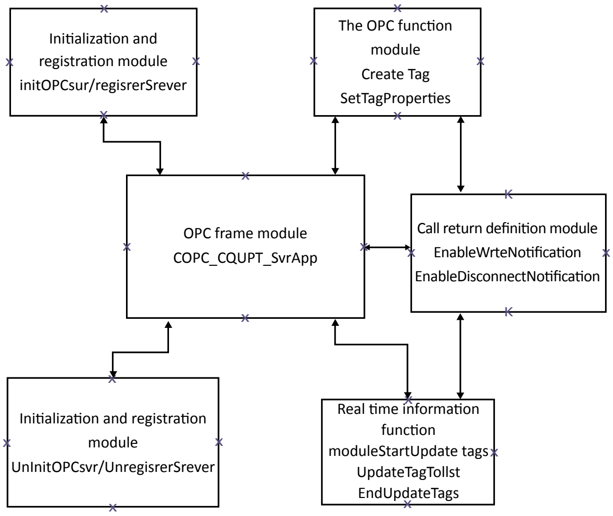
Initialization and registration module initOPCsur/regisrerSrever
The OPC function module Create Tag SetTagProperties
OPC frame module COPC_CQUPT_SvrApp
Call return definition module EnableWrteNotification EnableDisconnectNotification
Initialization and registration module UnInitOPCsvr/UnregisrerSrever
Real time information function moduleStartUpdate tags UpdateTagTollst EndUpdateTags

Figure 7: OPC development kit each module relation disgram.

1. OPC architecture module: It creates COM object to realize all functions in terms of OPC Speciation, such as OPC server interface object.
2. Return function API module: it defines detailed information of DLL parameters returning to OPC server and realizes message changing between client and server.
3. Initialization and register API function module: it registers, logs out and updates OPC server.
4. OPC Item API function: it operates creating, update and deleting items. Additionally, it is the key operation in OPC server SDK.
5. Real time message API module: it defines meths to update multi-items once.
6. Assistance API function module: it offers various assisted operation to OPC server.

Algorithm to develop OPC server

The development if OPC server includes 3 function modules. The first OPC standard interface module realizes all functions in terms of OPC Specification. Next, human machine interface (HMI) is able to add, delete and register items; finally, special communication protocol module offer driver for hardwares, such as profibus or can bus. By specific call-back function, the SDK connects to out communicated interface. These modules share same internal memory and solve safety problem by thread synchronization and mutual exclusion technology. The steps to develop server by SDK is shown in Figure 8. During the development, firstly, we call InitOPCSvr function to initial server. the call-back function is registered as finishing initialization. Secondly, by calling back CreateTag function, multi-items is added to OPC run time dramatically. At the same time, the property of item is setted by calling SetTagPropenies function. Thirdly, the server starts to send messages to client with calling UpdateTag function to update items constantly. Finally, we call UninitOPCSvr function to close server as the server stops.

Algorithm to develop OPC client

According to standard interface and OPC object mentioned in chapter, OPC Foundation packages OPC client's related information in OPCAuto.dll. The DLL realize communication between server custom interface and automation interface. Algorithm for client to visit server is as following: firstly, it connects to instance serve; Secondly, Group lists and item sets are obtained from server; Thirdly, it accesses data by synchronous communication or asynchronous communication. Finally, it stops connection and release resources. An example that a client reads a specific item value is shown as following [9]:

IOPCAutoServerPtr// define Automation Server interface;

Server.CreateInstance(_unidof(OPCServer));// create an Server instance;

Server->Connect(Server_Name.Node_Name);//create connection, Server_Name is the name of Server instancem, Node_Name is the name of Node instance;

IOPCGroupsPtr Groups=Server->OPCGroups;//read Group

IOPCGroupPtr Group=Groups->Add(Group_Name);//add specific Group;

OPCItemsPtr Items=Group->OPCItems;//read Item sets;

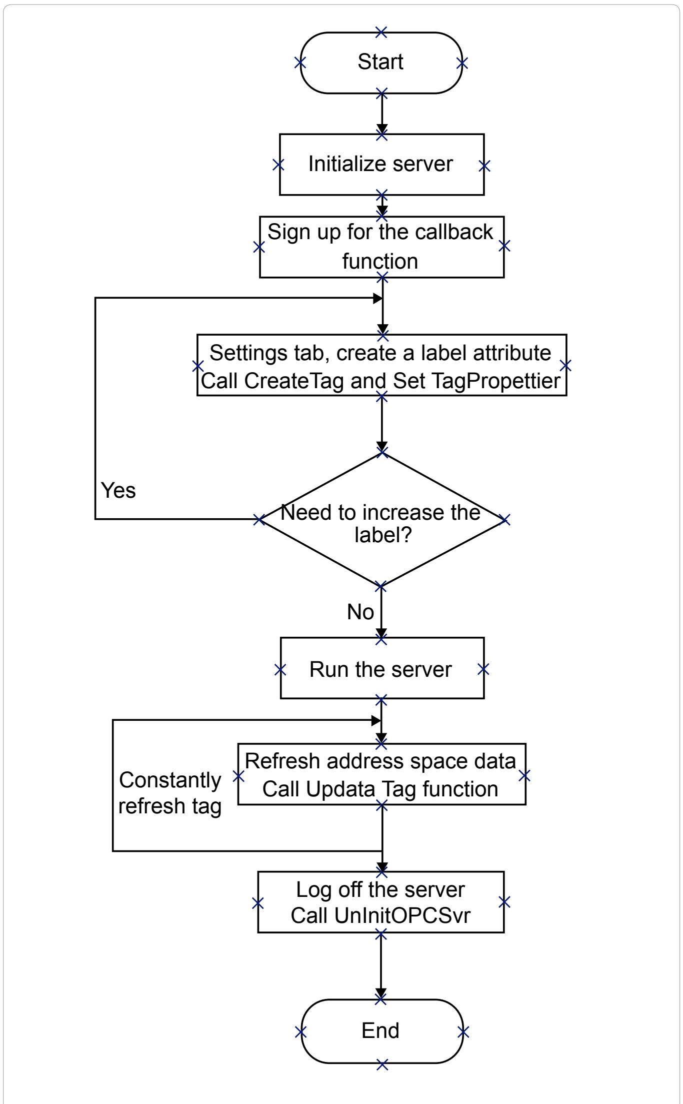
Start
Initialize server
Sign up for the callback function
Settings tab, create a label attribute
Call CreateTag and Set TagPropettier
Yes
Need to increase the label?
No
Run the server
Constantly refresh tag
Refresh address space data
Call Updata Tag function
Log off the server
Call UnInitOPCSvr
End

Figure 8: Flow chart.

OPCItemPtr Item=Item->AddItem(Tag_Name,0);//add specific Item, Tag_Name is the name of item;

_variant_t Value;//define item;

_variant_t Quality;//define Quality, 1 or 0;

_variant_t Time;//define time;

Item->Read(OPCCache,&Value,&Quality,&Time);//read item value, OPCCache is able to read data from data cache; otherwise,the parameter is OPCDevice if we read data from physical device;

Printf(“’%s’’%s’’%s\n’”,(constwchar_(*)_bstr_t(&Time),(constwchar_*)_bsr_t(&Quality),

(const wchar_t*)-bstr_t(&Value));// print out the reading result;

Server->disconnect();//stop connection;

In application, we use multi-threads to access data to take full advantages of system resource. In particular, we should use threads optimistically to achieve maximum performance because a object can be accessed only by object that creates itself in COM mechanism.

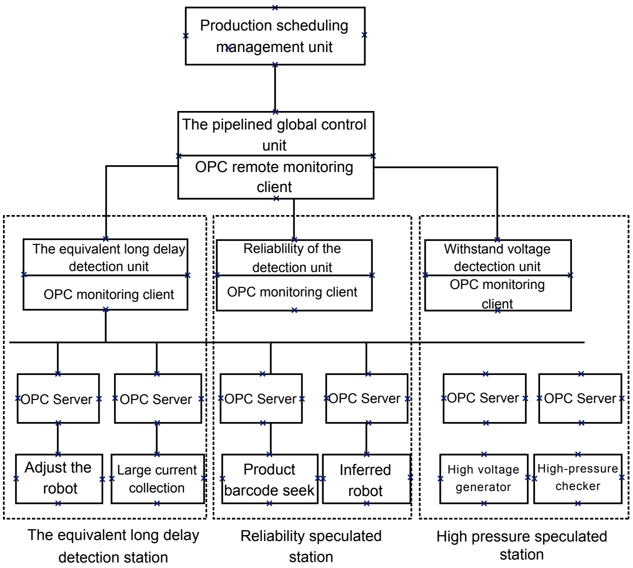
Production scheduling management unit
The pipelined global control unit
OPC remote monitoring client
The equivalent long delay detection unit
OPC monitoring client
Reliablility of the detection unit
OPC monitoring client
Withstand voltage dectection unit
OPC monitoring client
OPC Server
OPC Server
OPC Server
OPC Server
OPC Server
OPC Server
Adjust the robot
Large current collection
Product barcode seek
Inferred robot
High voltage generator
High-pressure checker
The equivalent long delay detection station
Reliability speculated station
High pressure speculated station

Figure 9: MCCB production assembly lines.

Control module layer (local control and global control)

Control layer is the core in system, consisting of local control module and global control module. According to manufacturing plan, path and state, the local control module mainly is responsible to send operated commands to control a functional cell or a physical device. Simultaneously, control modules achieve and display state information of device through device interface components. On the other hand, the global control module manage and optimize the whole system standing in the global level, especial in interlocking between control cell.

An Example of Prototype System in Application

This section describes a reconfigurable modular prototype system applied in a product and test line system of Molded-low-voltage circuit breaker (MCCB). MCCB is an important part of low voltage circuit breaker to protect the circuit [10]. The product and test system is able to test the MCCB via 7 major indicators, including running, reliability, equivalent time-lapse, overrun and opening range, withstand voltage and etc. Various testing data acquisition devices from different manufacturers apply in this system. According to this situation, we design a reconfigurable modular manufacturing system based on OPC technology which can be develop rapidly by modules and standard interfaces. The architecture is shown in figure 9. The system has applied in a MCCB factory in Shanghai. The results show that this system characters with standard interface and module developing rapidly [11].

Results

The paper introduces the theory, architecture, developing steps and other related content about reconfigurable module control system based on OPC technology. Adding a OPC based standard interface layer, the system realizes the reconfiguration and modular developing rapidly. The OPC technology, algorithm to develop OPC server/client and some core technology have described detailedly. Finally, we give an example to apply this system. Result shows that system runs well with good reconfiguration.

References

1. OPC Foundation (2009) OPC unified architecture specification part 11: Historical access.
2. OPC Foundation OPC unified architecture specification part 9: Alarms and conditions.
3. Schleipen M, Drath R, Sauer O (2008) The system independent data exchange format CAEX for supporting an automatic configuration of a production monitoring and control system. IEEE International Symposium on Indus Elec-ISIE, UK.
4. Alsmeyer F (2007) Durchgängige Nutzung von Prozessdaten im Lebenszyklus verfahrenstechnischer Anlagen VDIBerichte.
5. Hadlich T (2006) Providing device integration with OPC UA. Indus Inform IEEE International Conference, Singapore.
6. IEC 61804-2 (2002) Function Blocks for process control-Part 2: Specification of FB concept and Electronic Device Description Language (EDDL). PAS Pre-Standard.
7. Braune A, Hennig S, Hegler S (2008) Evaluation of OPC UA Secure Communication in Web Browser Applications. The IEEE International Conference on Industrial Informatics, Korea.
8. OPC Foundation (2009) OPC Unified Architecture Specification Part 9:Alarms and Conditions RC.
9. Huiming LU, Zhao Z, Changbin L (2009) Historical data processing in real-time database system. Electri Pow Auto Equip.
10. Guohua Ma (2001) The application of monitoring state software. Press of Tsing Hua University Bejing.
11. Beveridge J, Wiener R (2002) Win32 Multithreading programming. Press of Huazhong University of Science and Technology.